Universidad Central de Venezuela
Facultad de Ciencias
Escuela de Computación

Lecturas en Ciencias de la Computación
ISSN 1316-6239

Ingeniería Ontológica

Milagros Barrera
Haydemar Núñez
Esmeralda Ramos

ND 2012-01

Centro de Ingeniería de Software y Sistemas (ISYS)
Laboratorio de Inteligencia Artificial (LIA)
Caracas, enero 2012

ND 2012-01
Lecturas en Ciencias de la Computación
ISSN 1316-6239

INGENIERÍA ONTOLÓGICA

Barrera Milagros Núñez Haydemar Ramos Esmeralda
Enero 2012

Resumen

Las ontologías se caracterizan por definir un vocabulario común, que considera además la interpretación de los conceptos básicos de algún dominio específico y las relaciones entre ellos. En este sentido, una ontología no es más que una especificación de lo que existe en un dominio, convirtiéndose en una pieza fundamental de las tecnologías orientadas a la Web semántica.

Este documento tiene como propósito introducir al lector en el área de la Ingeniería Ontológica, disciplina constituida por el conjunto de actividades concernientes al proceso de desarrollo de las ontologías, a su ciclo de vida, los métodos y metodologías para construirlas y las herramientas y lenguajes que las soportan. También se presenta una amplia descripción del lenguaje Web Ontology Language -OWL- a través de sus elementos básicos; así como también una breve revisión de algunos desarrollos ontológicos de interés y una comparación entre el modelado de ontologías y el modelado de objetos.

Palabras Claves: Ontologías, Conocimiento, Métodos de construcción, OWL, Methontology.

Centro de Ingeniería de Software y Sistemas -ISYS-
Laboratorio de Inteligencia Artificial -LIA-
Escuela de Computación Facultad de Ciencias Universidad Central de Venezuela
Los Chaguaramos. Apartado 47002Caracas 1041-A Venezuela
mbarrera26@gmail.com, haydemar.nunez@ciens.ucv.ve, esmeralda.ramos@ciencs.ucv.ve

CONTENIDO

Introducción 3

1. Fundamentos Teórico de Ontologías 4

1.1. Definición de ontología 4

1.2. Tipos de ontologías 5

1.3. Componentes de las ontologías 7

1.4. Ingeniería ontológica 9

2. Métodos para la construcción de ontologías 9

2.1. Proceso de desarrollo de ontologías 9

2.2. Principios para el diseño de ontologías 10

2.3. Metodologías para la construcción de ontologías 12

2.4 Métodos de Evaluación de Ontologías 22

3. Lenguajes y herramientas para construir ontologías 25

3.1 Herramientas para el desarrollo de Ontologías 25

4. Razonamiento con ontologías 34

4.1. Vista general del Lenguaje de Ontologías Web (OWL) 35

5. Investigación en ingeniería ontológica 48

5.1. Estudio y revisión de algunos desarrollos ontológicos de interés 49

5.2. Modelado de Ontologías vs Modelado de Objetos 55

6. Conclusiones 59

7.Referencias 59

Introducción

La ingeniería ontológica refiere un conjunto de actividades relacionadas con el proceso de desarrollo de ontologías, incluyendo métodos, lenguajes y herramientas que apoyan este proceso [1]. La investigación en este campo se inició hace varias décadas y recientemente ha capturado una mayor atención de las grandes comunidades de desarrolladores de software, debido al interés cada vez mayor en áreas como la administración, negocios, medicina, Web semántica, entre otras

En este contexto, las ontologías se caracterizan por definir un vocabulario común además de incluir la interpretación de los conceptos básicos de algún dominio específico y las relaciones entre ellos. Por lo tanto, una ontología no es más que una especificación de lo que existe en dicho dominio, convirtiéndose en pieza fundamental de las tecnologías orientadas a la Web semántica.

Sin embargo, como ha sido un término que ha recibido diferentes interpretaciones, el propósito de este documento será esclarecer lo que se entiende por ontología, cuáles son sus componentes, cómo se clasifica, en qué consiste el proceso para desarrollarlas, qué lenguajes y herramientas apoyan su construcción, esquemas para valorar su calidad, entre otros. Todo esto con base en una revisión detallada de la literatura especializada en la ingeniería ontológica. El documento se ha estructurado de la siguiente manera:

En la primera sección se presentan algunas definiciones, se describen las categorizaciones proporcionadas por varios autores. Asimismo, se detallan los componentes de las ontologías, se especifican algunas razones para su creación y se presentan escenarios para aplicaciones ontológicas.

En la Sección 2 se describe el proceso de desarrollo de ontologías, detallando algunas metodologías, entre las que se tiene a Methotology, muy utilizada por la comunidad de desarrolladores en este campo, y a la Metodología NeOn, un proyecto europeo para la gestión de ontologías en la Web; además, se especifican

algunos principios para su diseño y finalmente, se presenta un esquema para valorar la calidad de una ontología.

En la Sección 3 se presentan algunos lenguajes y herramientas utilizadas para desarrollar ontologías y en la Sección 4 se exhibe una vista general del lenguaje ontológico OWL, a través de sus elementos básicos y se describe el razonamiento con ontologías construidas en este lenguaje.

Algunas investigaciones en la ingeniería ontológica así como el modelado de ontologías *vs* modelado de objetos se describen en la Sección 5.

1. Fundamentos Teórico de Ontologías

1.1. Definición de ontología

Desde hace mucho tiempo, el término ontología se ha empleado en el campo de la filosofía y hace varias décadas tomó especial interés en el campo de la Inteligencia Artificial (IA). En la actualidad, ha adquirido un nuevo impulso debido al desarrollo de la Web Semántica, proyecto cuya idea principal es transformar la red en un espacio de conocimiento además de un espacio de información. En este proyecto, convergen la IA y las tecnologías Web y se proponen nuevas técnicas y paradigmas para la representación del conocimiento que contribuyan a la localización e integración de recursos a través de la www [2]. La Web Semántica se apoya en la utilización de ontologías como vehículo para cumplir este objetivo [3].

Numerosas son las definiciones de ontologías, entre las que destacan dos referidas al campo de la filosofía y la IA:

(1) En la filosofía, disciplina que trata sobre la teoría de la existencia, el Diccionario de la Real Academia (DRAE) define ontología como: “Parte de la metafísica que trata del ser en general y de sus propiedades trascendentales”.

Desde esta perspectiva, una ontología estudia las categorías fundamentales del ser.

(2) En la Inteligencia Artificial, disciplina que trata sobre cómo lograr que las computadoras realicen tareas que, por el momento, los humanos hacen mejor, una definición de ontología ampliamente aceptada es la siguiente: "Especificación formal y explícita de una conceptualización" [3]. En este contexto, una ontología define un vocabulario común que describe en un dominio específico conceptos básicos y las relaciones entre ellos.

Se agrega expresividad a esta última definición complementándola de la siguiente manera [4]:

- Conceptualización: modelo abstracto de un fenómeno, que generalmente se expresa como un conjunto de conceptos (entidades, atributos, procesos), sus definiciones e interrelaciones.
- Formal: organización teórica de términos y relaciones usados como herramienta para el análisis de los conceptos de un dominio.
- Compartida: captura conocimiento consensual que es aceptado por una comunidad.
- Explícita: se refiere a la especificación de los conceptos y a las restricciones sobre estos.

1.2. Tipos de ontologías

Las ontologías se pueden clasificar considerando diferentes criterios. En este particular, se hace referencia a las categorizaciones proporcionadas por [5], [6] y [7]:

En [5] se ofrece tres dimensiones sobre las cuales varían los tipos de ontologías, a saber:

- **Formalidad:** referido al grado de formalismo del lenguaje usado para expresar la conceptualización, como altamente informal, informal estructurada, semi-formal y rigurosamente formal.
- **Propósito:** referido a la intención de uso de la ontología, entre las que se tienen aquellas para la comunicación entre personas, ontologías para la interoperabilidad entre sistemas y por último, las ontologías para beneficiar la ingeniería de sistemas.
- **Materia:** para expresar la naturaleza de los objetos que la ontología caracteriza. Con respecto a esta dimensión, las ontologías pueden ser: de dominio, de tarea y de representación.

Según [6], las ontologías pueden clasificarse de acuerdo a la cantidad y tipo de estructura de la conceptualización en:

- **Ontologías terminológicas**: especifican los términos que son usados para representar conocimiento en el universo del discurso. Suelen ser usadas para unificar vocabulario en un campo determinado.
- **Ontologías de información**: especifican la estructura de almacenamiento de bases de datos. Ofrecen un marco para el almacenamiento estandarizado de información.
- **Ontologías de modelado del conocimiento**: especifican conceptualizaciones del conocimiento. Contienen una rica estructura interna y suelen estar ajustadas al uso particular del conocimiento que describen.

Según su dependencia y relación con una tarea específica desde un punto de vista, en [7] se clasifican las ontologías en:

- **Ontologías de Alto nivel o Genéricas**: describen conceptos más generales.
- **Ontologías de Dominio**: describen un vocabulario relacionado con un dominio genérico.

- **Ontologías de Tareas o Técnicas básicas**: describen una tarea, actividad o artefacto.
- **Ontologías de Aplicación**: describen conceptos que dependen tanto de un dominio específico como de una tarea específica, y generalmente son una especialización de ambas.

1.3. Componentes de las ontologías

Una ontología es una herramienta conceptual que define un vocabulario común para quien necesita compartir información dentro de un determinado dominio. Esto incluye definiciones de los conceptos básicos del dominio, así como sus relaciones, que tienen que ser interpretables por máquinas [8]. En este sentido, la ontología está constituida por un vocabulario específico utilizado para describir una cierta realidad.

Producto de la revisión de la bibliografía especializada se aprecia que los componentes de una ontología varían según el dominio de interés y las necesidades de los desarrolladores. A continuación, se presentan los componentes de las ontologías que se describen en [9], [8] y [10].

- **Clase o tipo:** son la base de la descripción del conocimiento en las ontologías ya que detallan los conceptos del dominio. La clase se refiere a un conjunto de objetos (físicos, tareas, funciones, entre otros) y cada objeto en una clase es una instancia de esa clase. Desde el punto de vista de la lógica, los objetos de una clase se pueden describir especificando las propiedades que éstos deben satisfacer para pertenecer a esa clase. Una clase puede ser dividida en subclases, las cuales representarán conceptos más específicos que la clase a la que pertenecen. Una clase cuyos componentes son clases, se denomina superclase o metaclase.
- **Relaciones:** se establecen entre conceptos de una ontología para representar las interacciones entre éstos. Definidas por lo general como el

producto cartesiano de n conjuntos: R: C_1 x C_2 x …x C_n. Algunas de las relaciones mas utilizadas son:

- (a) *Instancia de:* asocian objetos a clases;
- (b) *Relaciones temporales:* implican precedencia en el tiempo y
- (c) *Relaciones topológicas:* establecen conexiones espaciales entre conceptos.

- **Instancias o individuos:** son objetos, miembros de una clase, que no pueden ser divididos sin perder su estructura y características funcionales. Pueden ser agrupados en clases.
- **Propiedades o *slots*:** los objetos se describen por medio de un conjunto de características o atributos que son almacenados en los *slots*. Éstos almacenan diferentes clases de valores. Las especificaciones, rangos y restricciones sobre estos valores se denominan características o facetas. Para una clase dada, los *slots* y las restricciones sobre ellos son heredados por las subclases y las instancias de la clase.
- **Taxonomía:** conjunto de conceptos organizados jerárquicamente. Las taxonomías definen las relaciones entre los conceptos, pero no los atributos de éstos.
- **Axioma:** elementos que permiten el modelado de verdades que se cumplen siempre en la realidad. Los axiomas pueden ser estructurales y/o no estructurales. Un axioma estructural establece condiciones relacionadas con la jerarquía de la ontología, conceptos y atributos definidos. Ejemplo: *el concepto A no es una clase de A*. Un axioma no estructural establece relaciones entre atributos de un concepto y son específicos de un dominio. Ejemplo: *la relación F=m*a, que debe cumplirse siempre entre los atributos F (fuerza), m (masa) y a (aceleración) de un determinado concepto*.
- ***Frame*:** un objeto que incluye clases, instancias y relaciones.
- **Conceptualización:** conjunto de conceptos, relaciones, objetos y restricciones que caracterizan un domino.

- **Vocabulario:** conjunto de palabras con una explicación y documentación que persigue la universalidad y el formalismo en el contexto de un dominio.

1.4. Ingeniería ontológica

Las ontologías se presentan como una buena solución para alcanzar interoperabilidad semántica, ya que permiten conceptualizar el conocimiento, generando formas comunes y compartidas de ver el mundo. Entre las ventajas proporcionadas por las ontologías destacan: (a) Reducción de la ambigüedad al disminuir las confusiones semánticas; (b) Reutilización del conocimiento por aquellas aplicaciones que lo requieran en el mismo dominio que la ontología representa; (c) Extensión de ontologías desarrolladas y su reutilización en otras ontologías; entre otras ventajas. Su importancia es tal que ha aparecido recientemente una rama de la ingeniería dedicada al conjunto de actividades concernientes al proceso de desarrollo de las ontologías, a su ciclo de vida, los métodos y metodologías para construirlas y las herramientas y lenguajes que las soportan, denominada Ingeniería Ontológica [1] y [11].

2. Métodos para la construcción de ontologías

2.1. Proceso de desarrollo de ontologías

En la comunidad de desarrolladores de ontologías, han surgido una serie de métodos de diseño y construcción de ontologías, que tienen por finalidad proporcionar un procedimiento comúnmente aceptado, validado y verificado, que garantice el logro de un producto exitoso.

A continuación, se describen las actividades que se incluyen en el proceso de diseño y construcción de una ontología según [8]:

- Definir las clases de la ontología.
- Organizar las clases en una jerarquía taxonómica (subclase-superclase).
- Definir las propiedades y describir los valores posibles para cada una.

- Dar valores a las propiedades para cada una de las instancias.

No obstante, las fases que normalmente conforman el ciclo de vida de construcción de una ontología son:

- **Especificación:** se identifica el propósito y el ámbito de la ontología.
- **Conceptualización:** se describe el modelo conceptual de la ontología.
- **Formalización***:* se transforma el modelo conceptual en un modelo formal.
- **Implementación***:* se implementa la ontología formalizada en un lenguaje de representación del conocimiento.
- **Mantenimiento:** se actualiza y corrige la ontología.

Además de las actividades que se deben realizar en las fases mencionadas, hay otras actividades que pueden llevarse a cabo durante todo el ciclo de vida como son:

- **Adquisición de conocimiento***:* se adquiere el conocimiento del dominio
- **Documentación***:* se documenta qué se ha hecho, cómo se ha hecho y por qué se ha hecho
- **Evaluación***:* técnicamente se juzga la ontología

Asimismo, existe otra actividad que depende de la metodología, ésta es la reutilización de ontologías*,* denominada a menudo integración [12]. Diseñar y construir ontologías para soportar aplicaciones de un dominio requiere de mucho tiempo y esfuerzo. Las ontologías de dominio, ofrecen posibilidades a los desarrolladores para explotar y reutilizar el conocimiento existente en el dominio, para así construir las aplicaciones con más facilidad y confianza [13].

2.2. Principios para el diseño de ontologías

El proceso de diseñar una ontología se caracteriza por ser complejo, ya que no responde a un único criterio lógico viéndose fuertemente afectado por el contexto

donde se construyen. En este sentido, las diferentes comunidades que desarrollan ontologías siguen sus propios principios, criterios, reglas o métodos, dependiendo del tipo de ontología a desarrollar o de una situación particular.

Al respecto, en [14] se establecen los siguientes principios que deben respetar las ontologías:

- **Parsimonia:** las ontologías deben ser parsimoniosas, deben contener suficientes conceptos, pero sólo aquéllos que son estrictamente necesarios.
- **Bases teóricas claras:** las ontologías deben tener bases teóricas cónsonas. Una ontología no debe ser una simple jerarquía de términos, sino un marco teórico que describe un dominio.
- **Categorías versus términos:** no se debe perseguir la especificación de los términos más comunes, sino de las categorías básicas del dominio del conocimiento.
- **Coherencia:** las categorías básicas deben ser coherentes y sus marcos deben tener sentido dentro del propio dominio de conocimiento. La representación de objetos del mundo real siempre depende del contexto en el que se usan los objetos.

Así mismo, en [8] se establecen las siguientes reglas básicas para el diseño de ontologías:

- No hay una única forma correcta de modelar un dominio. Siempre hay alternativas viables. La mejor solución siempre depende de la aplicación que se tiene en mente y de las extensiones que se puedan anticipar.
- El desarrollo de una ontología es necesariamente un proceso iterativo.
- Los conceptos en la ontología deben ser cercanos a objetos (físicos o lógicos) y a las relaciones en el dominio de interés. Éstos suelen ser nombres (objetos) o verbos (relaciones) en las frases que describen el dominio.

2.3. Metodologías para la construcción de ontologías

Existen muchas propuestas metodológicas para el diseño y construcción de ontologías. A continuación, se hará una breve revisión de alguna de ellas, a saber: Guía para crear ontologías de la Universidad de Stanford; Methontology, de la Universidad Politécnica de Madrid y Neón, un proyecto europeo.

2.3.1 Guía para crear ontologías

Metodología bastante sencilla, aplicada a la creación de ontologías [8]. La misma está compuesta por siete pasos, a saber:

1. **Determinar el dominio y el alcance o ámbito de la ontología**: con la finalidad de definir el dominio y el alcance de la ontología, se deben responder preguntas tales como: ¿cuál es el dominio que la ontología debe cubrir?, ¿para qué se va a usar la ontología?, ¿a qué preguntas debe dar respuesta la ontología?, ¿quién va a usar y a mantener la ontología?
2. **Considerar reutilizar ontologías existentes**: demostrar si es posible usar y extender fuentes de conocimientos ya existentes, y que puedan ser de utilidad para el dominio del problema.
3. **Enumerar los términos importantes en la ontología**: escribir una lista (precisa y sin ambigüedades) con todos los términos con los que se harán afirmaciones acerca del dominio o se explicará éste a un usuario.
4. **Definir las clases y la jerarquía de clases**: seleccionar de la lista creada en el paso anterior, aquellos términos independientes para constituir las clases. A partir de estas clases se establece la jerarquía.
5. **Definir las propiedades de las clases – (*slots*)**: describir los conceptos propios de la estructura interna de las clases. Por lo general los términos que no fueron seleccionados en el paso anterior pasan a considerarse propiedades de las clases.

Existen varios tipos de propiedades de los objetos que se pueden convertir en propiedades en una ontología: propiedades intrínsecas, propiedades extrínsecas, partes, si el objeto es compuesto y relaciones con otros objetos. Adicionalmente, todas las subclases heredan la propiedad de esa clase, es decir, todas las propiedades de la superclase son heredadas por sus subclases.

6. **Definir las características (facetas) de las propiedades**: las ranuras tienen diferentes propiedades que describen el tipo de valor, los valores permitidos, el número de valores (cardinalidad), así como otras características de los valores que la propiedad puede tener.

7. **Crear instancias**: definir una instancia individual de una clase requiere a) elegir una clase; b) crear una instancia individual de esa clase, y c) rellenar las propiedades con valores.

2.3.2 Methontology

Metodología desarrollada en el laboratorio de Inteligencia Artificial de la Universidad Politécnica de Madrid [11], que permite la construcción de ontologías a nivel de conocimiento incluyendo la identificación del proceso de desarrollo de la ontología, un ciclo de vida basado en la evolución de prototipos y técnicas particulares para realizar cada actividad.

La Fundación para Agentes Físicos Inteligentes (FIPA), que promueve la interoperabilidad entre las aplicaciones basadas en agentes [15], recomienda utilizar Methontology, como metodología para la construcción de ontologías.

El ciclo de vida de esta metodología se muestra en la Figura 1. Las actividades de control, aseguramiento de calidad, adquisición de conocimiento, integración, evaluación documentación y manejo de configuración se realizan simultáneamente con las actividades de desarrollo.

Las actividades de desarrollo propuestas por la metodología son las siguientes:

a) Especificación.

Realizar un documento donde se señale el alcance, objetivos, propósito, nivel de formalidad y usuarios finales de la ontología.

b) Conceptualización.

Consiste en organizar y convertir una percepción informal de un dominio en una especificación semi-formal usando un conjunto de representaciones intermedias (tablas, diagramas) que puedan ser entendidas por los expertos del dominio y los desarrolladores de ontologías.

c) Formalización.

Realizar la transformación del modelo conceptual en un modelo formal o semi-computable.

d) Implementación.

Realizar la codificación de la ontología utilizando un lenguaje formal (Ontolingua, XOL, OIL, DAML, OWL, entre otros).

e) Mantenimiento.

Esta actividad permite la actualización y corrección de la ontología.

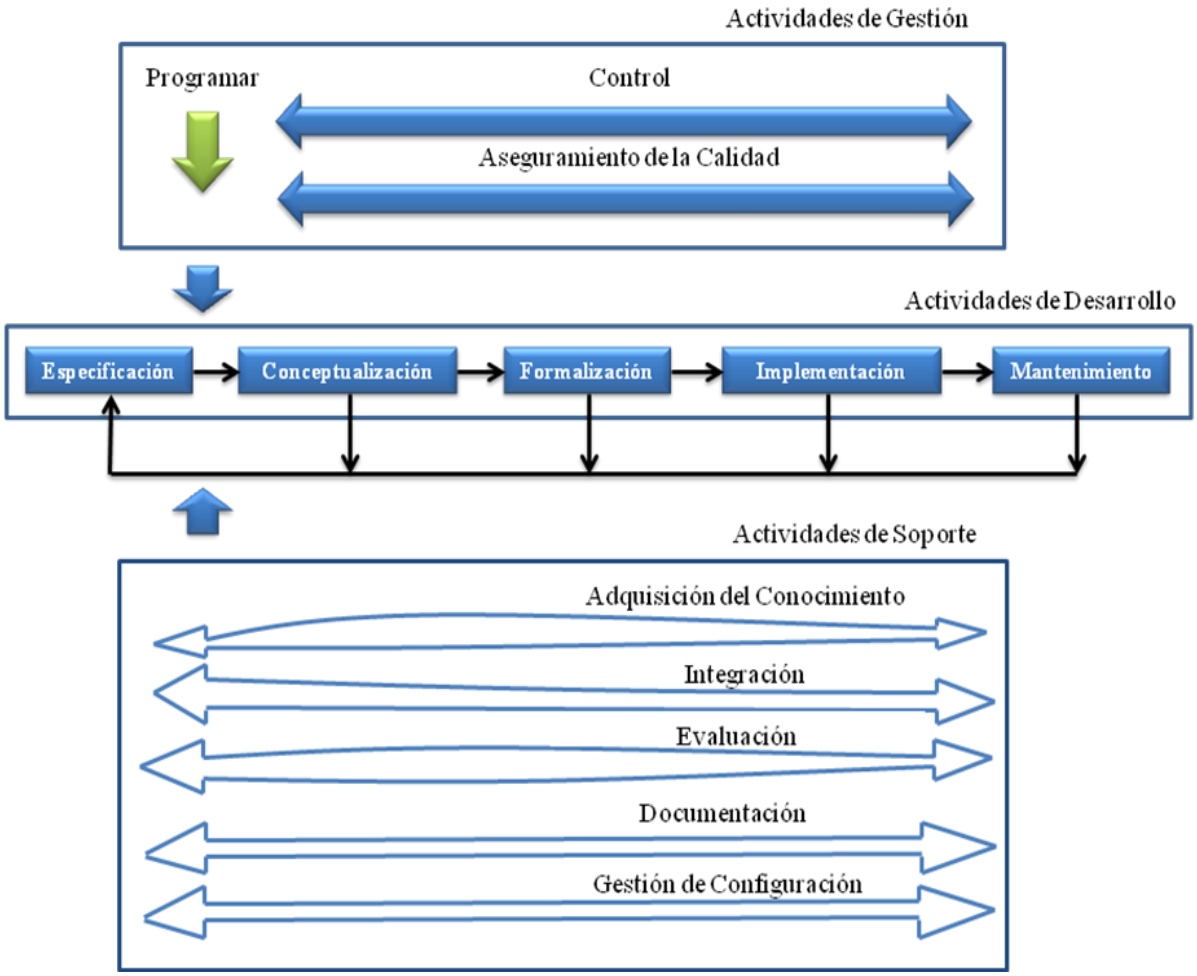

Figura 1. Ciclo de Vida de Methontology [11].

2.3.3 Metodología NeOn

La Metodología NeOn es un proyecto europeo (http://www.neon-project.org/) que inició en marzo del 2006 con la participación de líderes científicos y técnicos, en los campos de la ingeniería ontológica, tecnologías de colaboración, ingeniería del software e interacción humano-computador. Su objetivo principal es mejorar la capacidad de manipular múltiples ontologías en red, incluyendo evolución y mantenimiento de las mismas, además de orientar el proceso de desarrollo de ontologías colaborativas y la reutilización de recursos ontológicos y no-ontológicos.

En este sentido, los autores del proyecto señalan que las ontologías en la Web no son artefactos independientes. Ellas se encuentran interconectadas semánticamente mediante las siguientes relaciones:

- **Importaciones y dependencias**

Una de las relaciones más comunes entre dos ontologías es “la dependencia”. Este es el caso en que para definir su propio modelo, una ontología requiere referir las definiciones incluidas en otra ontología. Al respecto, OWL incluye una primitiva (owl:imports) que permite a un desarrollador de ontologías declarar tal relación, combinando las definiciones de ambas ontologías.

- **Control de Versiones**

Se refiere a la actividad de hacer seguimiento de las diferentes versiones de una ontología. En un entorno colaborativo de Ingeniería Ontológica, el proceso de evolución de la ontología debe ser cuidadosamente monitoreada, además de asegurar que la información sea actualizada. OWL incluye primitivas para declarar las relaciones entre versiones de ontologías.

- **Alineaciones**

En ocasiones diferentes ontologías se superponen para representar de manera desigual partes de un mismo dominio. En este sentido, la alineación consiste en hacer corresponder (mapeo) esos modelos diferentes, declarando cuales entidades deben ser consideradas como iguales o cuando una es más general que otra. El objetivo principal de las alineaciones es garantizar interoperabilidad semántica.

- **Modularización**

Se refiere a la identificación de los componentes de la ontología que pueden considerarse por separado, participando cada uno en un subdominio. El propósito es manipular, usar y mantener estos componentes más fácilmente.

Es así como la metodología, propone un marco de trabajo para la construcción de redes de ontologías que incluye nueve escenarios, a saber: (ver Figura 2) (COLOCAR LA FIGURA 2 A CONTINUACIÓN O EN LA PRÓXIMA PÁGINA, ESTÁ MUY ALEJADA DE LA CITA)

- ***Escenario 1: Desde la especificación a su implementación.*** Consiste en especificar los requisitos que debe cumplir la ontología. Para ello, se consideran aspectos claves como las necesidades, intenciones, usos, usuarios, entre otros. El producto obtenido en este escenario es el Documento de Especificación de Requisitos de la Ontología (DERO) que contiene el propósito, alcance, lenguaje de implementación, usuarios finales, requerimientos funcionales y no funcionales de la ontología, y un glosario previo de términos.

Asimismo, otras actividades como la búsqueda de los recursos en el mismo dominio que pueden ser reutilizados en la construcción de la ontología y la planificación del proceso de desarrollo son llevados a cabo. En el caso de la búsqueda de recursos, existen repositorios y motores de búsqueda en línea que pueden apoyar, por ejemplo, Watson (http://watson.kmi.open.ac.uk) para buscar ontologías, Cupboard (http://cupboard.open.ac.uk) para publicar ontologías y Oyster, un sistema para compartir ontología *peer-to-peer*.

- ***Escenario 2: Reutilización y Re-ingeniería de los recursos no-ontológica (NOR).*** Con el propósito de acelerar la construcción de las ontologías, los desarrolladores pueden llevar a cabo el proceso de reutilización y re-ingeniería de recursos no-ontológicos. En este sentido, los desarrolladores basados en el DERO, deciden cuáles de estos recursos pueden ser reutilizados para construir la red de ontologías. Seguidamente, los NORs seleccionados debe ser rediseñados. Estos recursos son fuentes de conocimientos por lo general heterogéneos, como por ejemplo, glosarios, vocabularios, diccionario de sinónimos, esquemas de clasificación, entre otros, cuya semántica aún no ha sido formalizada en una ontología.

Respecto a la reutilización de NORs, NeOn lleva a cabo las siguientes actividades: (a) Búsqueda de recursos no-ontológicos; (b) Evaluación de un conjunto de recursos no-ontológicos que son candidatos y, (c) Selección del recurso no-ontológico más apropiado.

Por otra parte, respecto a la reingeniería de NORs, NeOn también define un patrón con una secuencia bien definida de actividades para transformar el contenido y esquema de los recursos. Estas actividades son: (a) Ingeniería inversa de recursos no-ontológicos; (b) Transformación de los NORs; (c) Ingeniería de la Ontología (formalización e implementación).

- ***Escenario 3: La reutilización de recursos ontológicos.*** Los desarrolladores utilizan recursos ontológicos, tales como ontologías generales que no están delimitadas por dominios específicos y pueden ser reutilizadas por dominios diferentes. El proceso que se lleva a cabo abarca las siguientes actividades: (a) Identificación del tipo de ontología general a ser reutilizada; (b) Identificación de las definiciones y axiomas más importantes de las teorías de apoyo; (c) Estudio comparativo; (d) Selección de una ontología general, a través de las siguientes tareas: análisis de grupos locales de preguntas de competencias, identificación de las características de las ontologías generales y determinación de la ontología general que mejor se adapte; (e) Personalización de la ontología general seleccionada, a través de las siguientes tareas: poda y enriquecimiento de la ontología, traducción al lenguaje local y finalmente, la evaluación; (f) Integración de la ontología general.

Otro recurso ontológico que puede ser reutilizado son las ontologías de dominio, a través de las siguientes actividades: (a) Búsqueda de candidatos de ontologías de dominio en librerías, repositorios y registros y (b) Evaluación. El objetivo de esta actividad es descubrir si el conjunto de ontologías que resultaron candidatas son útiles para el desarrollo de la red de ontologías, comprobando si los alcances, propósitos, requisitos, entre otros, son similares a los establecidos en el DERO; (c) Selección de la ontología más adecuada del conjunto de ontologías que

resultaron útiles en la actividad anterior; e (d) Integración. Esta actividad se lleva a cabo tomando uno de los siguientes modos de integración: reutilización sin cambios en la ontología, reutilización con cambios significativos y combinación de varias ontologías de dominio para obtener una nueva.

Asimismo, en ocasiones resulta apropiado recuperar sólo trozos de conocimientos en vez de grandes ontologías. En este sentido, otro recurso ontológico que puede ser reutilizado son las declaraciones y las actividades del proceso son las mismas que se llevan a cabo en la reutilización de ontologías de dominio; sin embargo, una última actividad es añadida: chequeo de inconsistencias locales.

- ***Escenario 4: La reutilización y re ingeniería de recursos ontológicos.*** Los desarrolladores de ontologías reutilizan y rediseñan recursos ontológicos. NEON incluye una librería de patrones de diseño que ofrecen soluciones de modelado para ser aplicadas en la solución de problemas recurrentes en el diseño de ontologías. Entre los tipos de patrones se encuentran: los estructurales, de correspondencia, de razonamiento, de léxico-sintaxis y de contenido.

- ***Escenario 5: La reutilización de los recursos y la fusión ontológica.*** Este escenario se produce cuando varios recursos ontológicos en el mismo dominio se seleccionan para su reutilización, y los desarrolladores desean crear un nuevo recurso ontológico con los seleccionados.

- ***Escenario 6: La reutilización, la fusión y re-ingeniería de recursos ontológicos***. Desarrolladores reutilizan, fusionan y reestructuran los recursos ontológicos. Este escenario es similar al Escenario 5, pero en este caso los desarrolladores deciden volver a diseñar el conjunto de recursos combinados.

- ***Escenario 7: La reutilización de patrones para el diseño de ontologías (PDOs).*** En la construcción de ontologías, los desarrolladores pueden acceder a

los repositorios para reutilizar PDOs, con la finalidad de ofrecer soluciones a problemas recurrentes en el diseño de ontologías

- ***Escenario 8: Reestructuración de los recursos ontológicos.*** Los desarrolladores pueden reestructurar (modularización, poda, extensión y/o especialización) recursos ontológicos para ser integrados en la red de ontologías. Específicamente, la modularización ayuda a mejorar el rendimiento, facilita el desarrollo y mantenimiento de la ontología a través de sus componentes autónomos débilmente acoplados y/o facilita la reutilización de partes de la ontología.

- ***Escenario 9. La localización de los recursos ontológicos.*** Consiste en adaptar una ontología a otros lenguajes y culturas, obteniendo así una ontología multilingüe.

NeOn también incluye un glosario de procesos y actividades, producto de un consenso entre ingenieros, editores y usuarios dentro del proyecto, este glosario identifica y define los procesos y actividades que participan en la construcción de redes de ontologías. Actualmente, cuenta con 59 definiciones escritas en el idioma inglés y ordenadas alfabéticamente. También contiene notas para aclarar procesos y actividades que son similares y/o sinónimos. Se publica en la página Web de NeOn: http://www.neon-project.org/web-content/images/Publications/neonglossaryofactivities.pdf.

Por otra parte, NeOn propone dos modelos para organizar los procesos y actividades que se llevan a cabo durante el proceso de desarrollo de una ontología: a) Cascada, (b) Incremental-iterativo.

Asimismo, NeOn plantea un conjunto de orientaciones metodológicas que describen como llevar a cabo los diferentes procesos, a través de: (a) Metas, Entradas, Salidas y Limitaciones; (b) Flujos de Trabajo; y (c) Ejemplos.

La Metodología NeOn no establece un flujo de trabajo rígido, por el contrario, sugiere una variedad de caminos para la construcción de redes de ontologías. Además, ofrece un *kit* de herramientas que incluye varios *plug-ing* que apoyan las tareas del proceso de desarrollo.

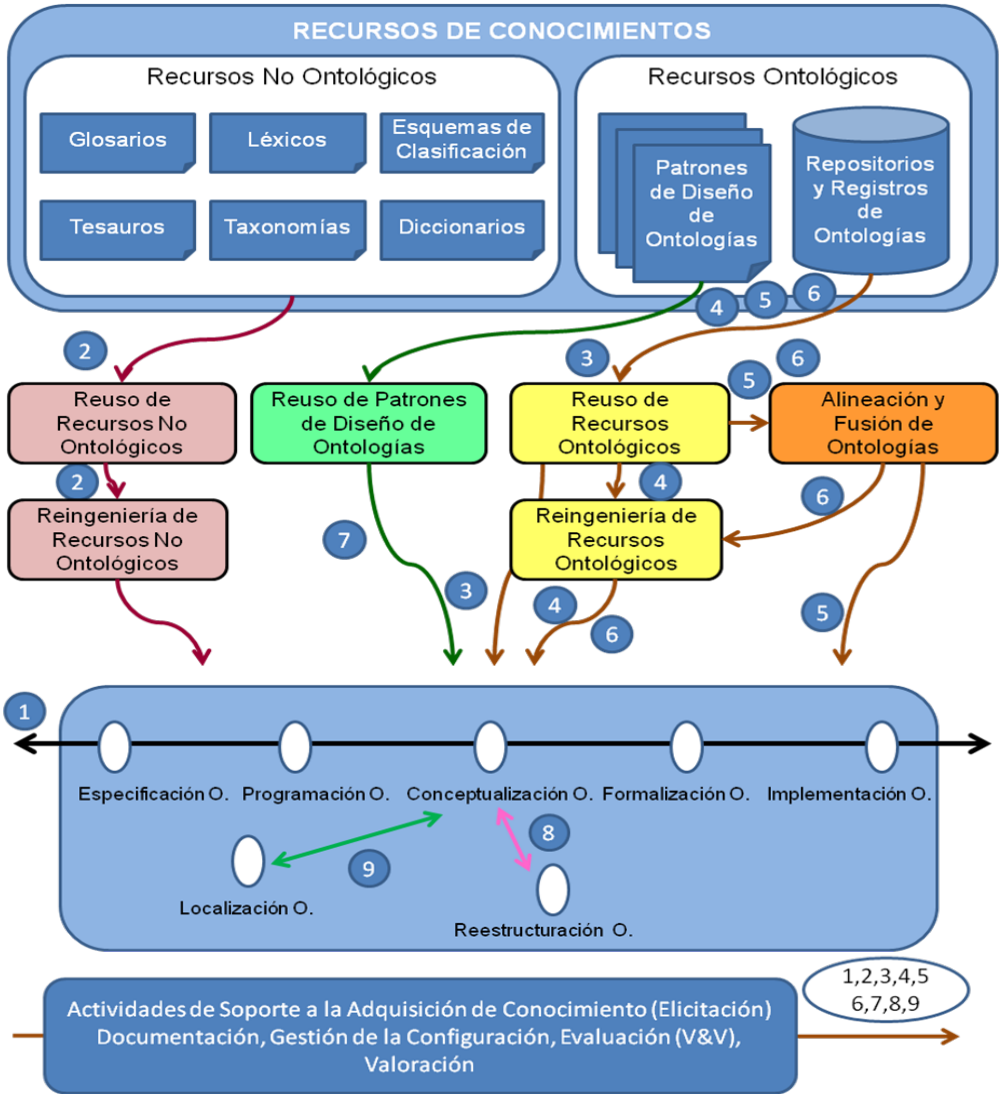

Figura 2. Escenarios de la Metodología NeOn

2.4 Métodos de Evaluación de Ontologías

Como cualquier aplicación de software, el contenido de las ontologías (conceptos, taxonomía y axiomas) debe ser evaluado antes de ser usado o reutilizado en otras ontologías o aplicaciones. En [11] se señala, que la evaluación de la ontología es un juicio técnico de su contenido con respecto a un marco de referencia (requerimientos, preguntas de competencia, entre otros) durante todas las fases de su ciclo de vida. Este proceso incluye la verificación y la validación de la ontología y debe llevarse a cabo en las definiciones y axiomas establecidos explícitamente en ella y en las que pueden ser inferidas desde otras aplicaciones.

La verificación se refiere a construir la ontología correctamente, es decir, asegurar que sus definiciones implementan correctamente los requerimientos de la ontología. Por su parte, la validación se refiere a garantizar que las definiciones ontológicas modelan realmente el mundo real para el que la ontología fue creada.

En el proceso de evaluación se establecen y ejecutan un conjunto de pruebas y se analizan los resultados de éstas [16], partiendo de tres posibles estados de las ontologías: a) Premodelado, donde se consideran la revisión y evaluación de los materiales disponibles para su construcción b) Modelado, donde se comprueba la calidad de los significados y la consistencia y redundancia de los conceptos, utilizando otras ontologías y preguntas de competencia, así mismo, se evalúan los posibles errores sintácticos cometidos durante la codificación y c) Entregadas (delivery), donde se determina su calidad, comparando con otras ontologías diferentes pero equivalentes.

Existen numerosas propuestas de evaluación de ontologías, tales como las planteadas en: [17], [18], [19], [20], [21], entre otros. Según [22], éstas propuestas coinciden en la evaluación de los siguientes criterios: a) la rigurosidad taxonómica, b) el lenguaje utilizado para la codificación, c) el rendimiento de las aplicaciones o tareas que utilizan las ontologías y d) el vocabulario utilizado para representar los

conceptos y relaciones del dominio modelado. En [23], destacan que muchos de estos criterios se evalúan sobre la base de la comparación con otras ontologías disponibles las cuales se usan como referencia o estándar de oro.

Sin embargo, cuando ocurre que la ontología que se está desarrollando es la única descripción que se conoce del dominio, los desarrolladores se ven en la necesidad de utilizar de manera parcial los métodos disponibles no garantizando una evaluación suficientemente confiable y completa.

En este sentido, en [22] se indica la posibilidad de valorar la calidad de una ontología sin la necesidad de recurrir a referencias previas, examinando un conjunto mínimo de criterios como son: que el vocabulario utilizado para representar el conocimiento tenga cobertura suficiente del corpus, que la ontología esté escrita de manera correcta, sin errores y conforme a las reglas del lenguaje utilizado, que la estructura taxonómica que organiza los conceptos y términos del dominio sea completa, sin redundancias y consistente y que satisfaga los requerimientos para los cuales fue creada y, de manera particular, que las preguntas de competencia sean respondidas adecuadamente.

A continuación, se presenta el esquema de evaluación propuesto en [22], el cual está constituido por cuatro fases, una para cada criterio a evaluar: (1) Uso correcto del lenguaje; (2) Exactitud de la estructura taxonómica; (3) Validez del vocabulario y (4) Adecuación a requerimientos. La Tabla 1 describe para cada fase: las actividades, los insumos y el producto obtenido.

Tabla 1. Esquema para evaluar ontologías únicas en un dominio de conocimiento

DEL ESQUEMA PARA LA EVALUACION DE ONTOLOGIAS UNICAS EN UN DOMINIO			
NOMBRE/DESCRIPCION	**ACTIVIDADES**	**INSUMOS**	**PRODUCTO**
Uso correcto del lenguaje: Se evalúa la codificación, basado en las características y reglas de construcción del lenguaje usado.	✓ Seleccionar un lenguaje de codificación sólido y completo, que cumpla con estándares de desarrollo ontológico ✓ Evaluar sintácticamente la ontología en cada fase, para corregir inconsistencias sintácticas	✓ Ontología ✓ Herramientas para la evaluación (Editor Protegé-OWL, Analizador sintáctico DAML, entre otros)	Código libre de errores
Exactitud de la estructura taxonómica: Se examina la taxonomía. Para ello se considera la consistencia, completitud y no redundancia de los conceptos y términos.	✓ Identificar inconsistencias (conceptos que no pertenecen a una clase en particular, clases definidas como generalizaciones o especializaciones de sí mismas, entre otros) ✓ Evaluar la completitud de los conceptos codificados ✓ Evaluar la existencia de redundancias en clases, instancias y relaciones ✓ Identificar la existencia de clases e instancias con diferentes nombres y definiciones similares, clases que tienen más de una relación de subclase. ✓ Identificar la omisión de conocimiento disjunto entre clases de la estructura.	✓ Taxonomía	Taxonomía completa y libre de inconsistencias
Validez del vocabulario: Se evalúa el significado de los términos y conceptos a partir del conocimiento de expertos, recopilaciones de textos u otra fuente disponible del dominio.	✓ Identificar, extraer y organizar los términos significativos del dominio. ✓ Evaluar el vocabulario considerando medidas de calidad de resultados usadas en escenarios de recuperación de información.	✓ Conocimiento de expertos, textos y otras fuentes.	Tabla de términos válidos en el dominio
Adecuación a requerimientos: Se valida si la ontología implanta los requerimientos preestablecidos y si responde a las preguntas de competencia.	✓ Verificar que las especificaciones del documento de requerimientos se cumplan. ✓ Verificar que las respuestas proporcionadas por la ontología a las preguntas de competencia sean correctas y pertinentes.	✓ Documento de requerimientos de la ontología	Una Ontología que satisface los requerimientos identificados inicialmente

3. Lenguajes y herramientas para construir ontologías

Como se mencionó anteriormente, las ontologías capturan conocimiento consensuado de un modo general y formal, de tal manera que pueda ser compartido y reutilizado por distintos grupos de personas y aplicaciones de software. En este sentido, las ontologías conforman uno de los pilares básicos de las aplicaciones actuales y de la Web Semántica y requieren de un lenguaje lógico y formal para ser expresadas. En la inteligencia artificial se han desarrollado numerosos lenguajes para este fin, algunos basados en la lógica de predicados, otros basados en *frames* (taxonomías de clases y atributos) y otros orientados al razonamiento.

Con los lenguajes ontológicos se pretende alcanzar un alto grado de expresividad y uso. Dentro de los principales lenguajes destacan: XML, XML Schema, RDF, RDF Schema y OWL. En la Tabla 2 se describen brevemente, sin embargo, en la sección 4 se hará énfasis en la recomendación de la W3C: OWL.

3.1 Herramientas para el desarrollo de Ontologías

Los editores de ontologías son herramientas especializadas que apoyan la construcción de estas en base a un determinado lenguaje, por ejemplo el OWL. Entre las diversas herramientas informáticas empleadas en el desarrollo de ontologías destacan: Protegé, KAON, WebODE, Swoop, WebOnto, Ontolingua y Chimaera. Más adelante se mencionan algunas de sus características.

3.2.1. Protegé: editor de ontologías de código abierto, desarrollado por el grupo SMI (Stanford Medical Informatics) de la Universidad de Stanford (http://protege.stanford.edu/). Esta herramienta permite crear y editar ontologías sobre RDFS, OWL y XML *Schema*, dentro de un entorno gráfico, usable e intuitivo. Utiliza la representación basada en marcos e implementada en CLIPS.

Tabla 2. Algunos lenguajes ontológicos

Lenguaje	¿Qué es?	Principales características
XML (eXtensible Markup Languaje)	Metalenguaje que define una serie de normas básicas que permiten diseñar cualquier otro vocabulario de etiquetado para la Web	✓ Utiliza un estándar abierto e independiente de la plataforma ✓ Proporciona un mecanismo simple y estandarizado para representar conocimiento ✓ Permite reutilizar contenido ✓ Es legible en los navegadores web ✓ Utiliza UNICODE, sistema de codificación multilingüe y universal ✓ No garantiza la interoperabilidad semántica y tampoco entre sistemas heterogéneos
XML Schema	Lenguaje que define la estructura de los documentos XML que estén asignados a tal esquema y los tipos de datos válidos para cada elemento y atributo.	✓ Enfoque modular que facilita la reutilización del código ✓ Permite especificar y jerarquizar un gran número de tipos de datos (simples, complejos y definidos por el usuario) ✓ Es extensible ✓ Expresa vocabularios compartidos que permiten a las máquinas extraer las reglas hechas por las personas
RDF (Resource Description Framework)	Lenguaje de propósito general para representar información en la web.	✓ Provee un conjunto de declaraciones que permiten describir clases y propiedades. ✓ No tiene capacidad para expresar la semántica del vocabulario
RDF Schema	Extensión semántica de RDF	✓ Permite modelar metadatos con una representación explícita de su semántica ✓ Permite especificar restricciones de tipos de datos para los sujetos y objetos de las *tripletas* de RDF, introduciendo unas primitivas de modelado orientado a objetos: **rdfs:Class**, **rdfs:Property**, **rdfs:subClassOf** ✓ Expresividad limitada. No soporta conceptos ontológicos básicos como equivalencia, disyunción, negación, relaciones inversas y limitaciones de cardinalidad.
OWL (Web Ontology Languaje)	Vocabulario propuesto como estándar por la W3C para definir ontologías	✓ Especifica de manera precisa la semántica formal de un ámbito determinado de la realidad ✓ Hace posible el razonamiento con las ontologías para inferir conocimiento ✓ Proporciona tres lenguajes (OWL Lite, OWL DL y OWL Full)

Protégé en su núcleo, implementa un rico conjunto de estructuras de modelado de conocimiento y acciones que apoyan la creación, visualización y manipulación de ontologías en varios formatos de representación (ver Figuras 3 y 4). Se presenta como una única aplicación Java de código libre que se ejecuta localmente en la máquina del cliente; su arquitectura se basa en *plug-ins* como Jambalaya, que permite representar y editar gráficamente los archivos que está creando el usuario. Se conecta fácilmente con razonadores de lógicas descriptivas y genera automáticamente documentación para ontologías y bases de conocimiento en formato OWLDoc.

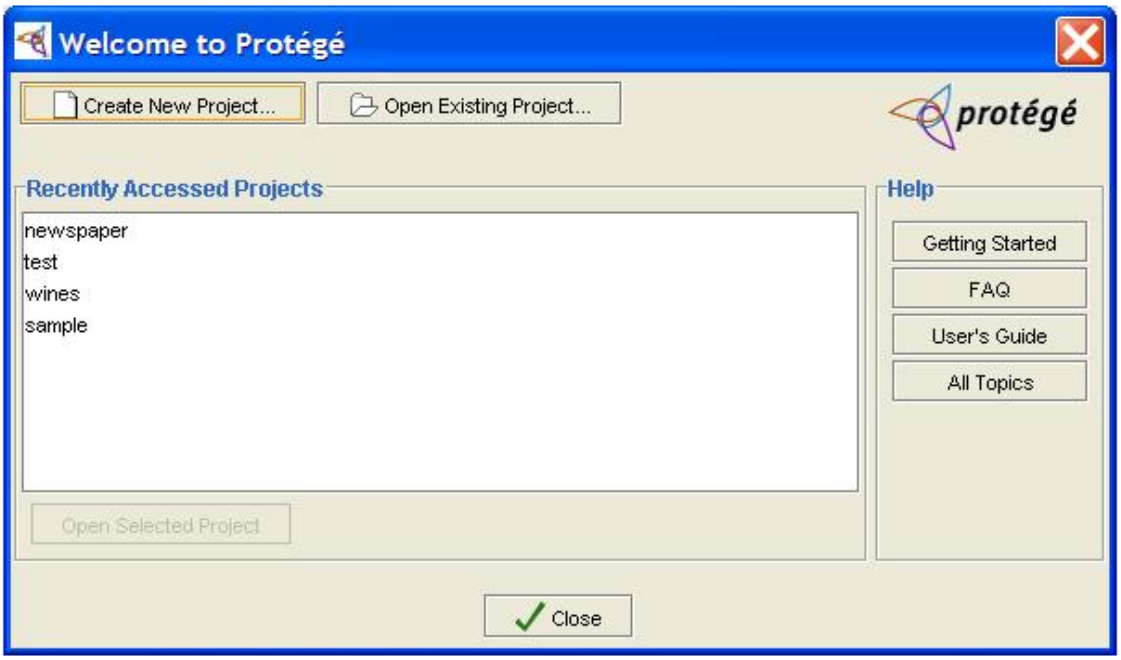

Figura 3. Protegé (Pantalla de Bienvenida)

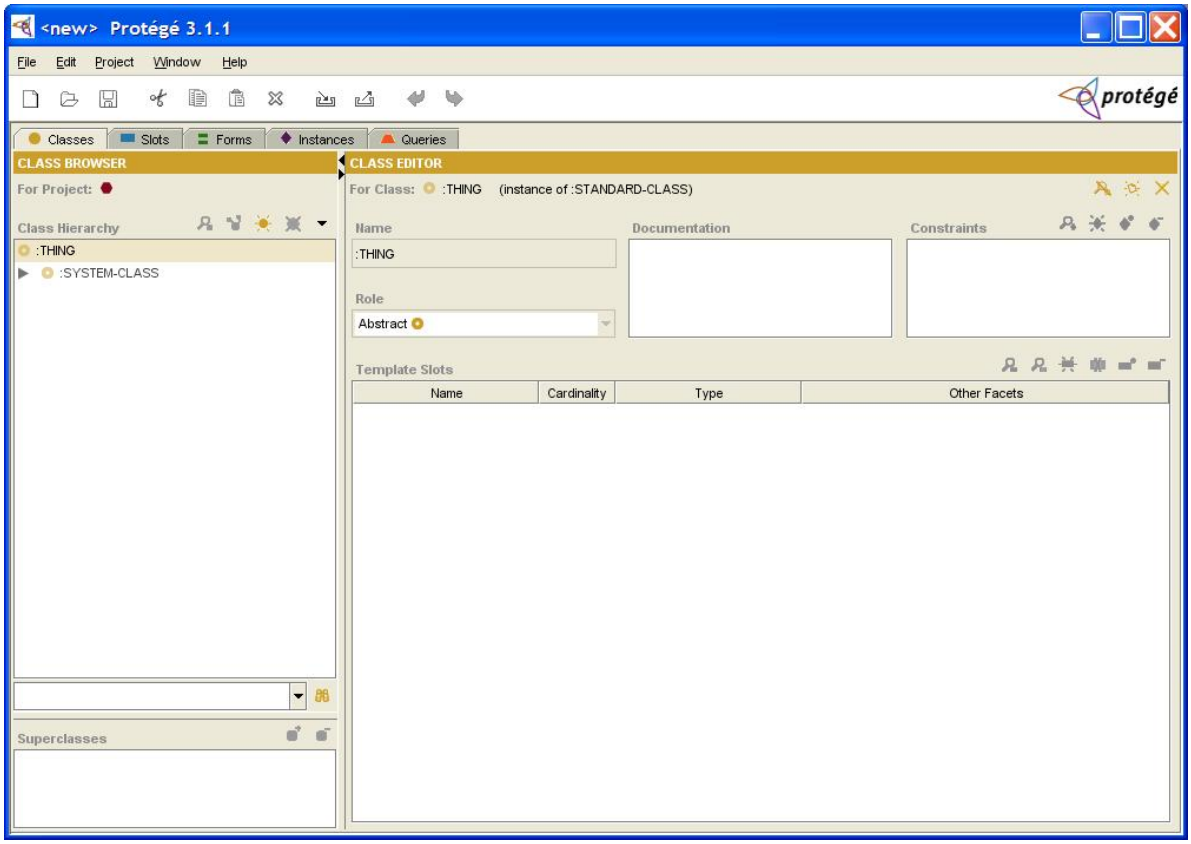

Figura 4. Protegé (Nuevo proyecto)

3.2.2. KAON: desarrollada por la Universidad de Karlsruhe (el nombre de KAON procede de KArlsruhe Ontology). Es una herramienta de libre acceso que permite la creación y gestión de ontologías con facilidad, así como la creación de aplicaciones comerciales basadas en ontologías. Un aspecto fundamental es su enfoque es la integración de las tecnologías tradicionales para la gestión de tales aplicaciones. La escalabilidad y la eficiencia son características fundamentales de KAON.

3.2.3. WebODE: es una herramienta para modelar el conocimiento usando ontologías. Fue desarrollada en la Escuela Técnica de Informática (FI) en Madrid y es el equivalente Web de ODE (Ontology Desing Environment). Ofrece soporte para múltiples usuarios. Su interfaz basada en formularios y editores gráficos permite guiar la conceptualización y editar taxonomías. Su arquitectura ofrece

chequeo completo de la consistencia e importación avanzada de términos. Posee API para el acceso desde cualquier aplicación usando RMI o CORBA. (ver Figuras 5 y 6)

Figura 5. WebODE (Pantalla de bienvenida)

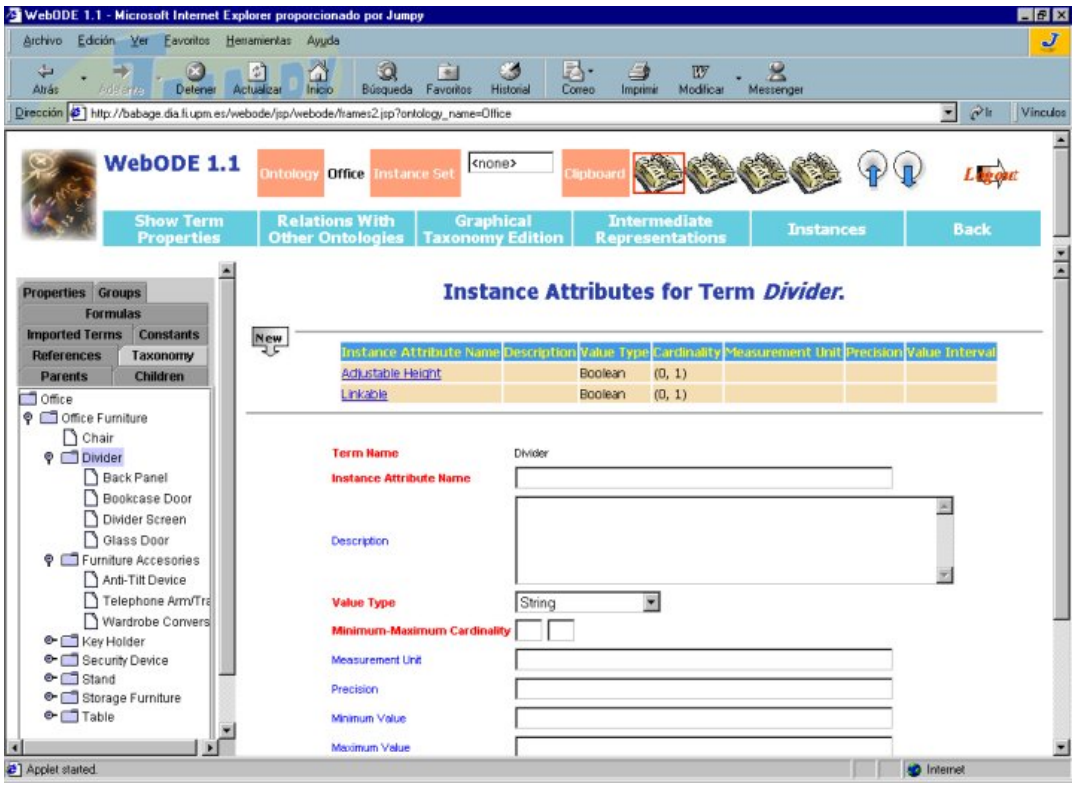

Figura 6. WebODE (Entorno de la versión 1.1)

3.2.4. Swoop: proyecto de código abierto desarrollado en el laboratorio de Mindswap de la Universidad de Meryland. Se trata de una herramienta para la creación, edición y depuración de ontologías OWL, basado en conceptos de diseño y navegación hipermedia. Su aspecto es similar a un navegador Web, pudiendo recorrer los distintos elementos de las diferentes ontologías cargadas a base de activar hipervínculos. Esta herramienta admite varias sintaxis de representación, desde OWL a la sintaxis habitual RDF/XML. Permite la búsqueda semántica. Utiliza anotación colaborativa mediante el entorno de trabajo Annotea. Es fácilmente extensible por herramientas de anotación semántica tipo SMORE; además incluye el razonador Pellet integrado, con la posibilidad de depurar la ontología al tiempo que se está creando. (ver Figura 7)

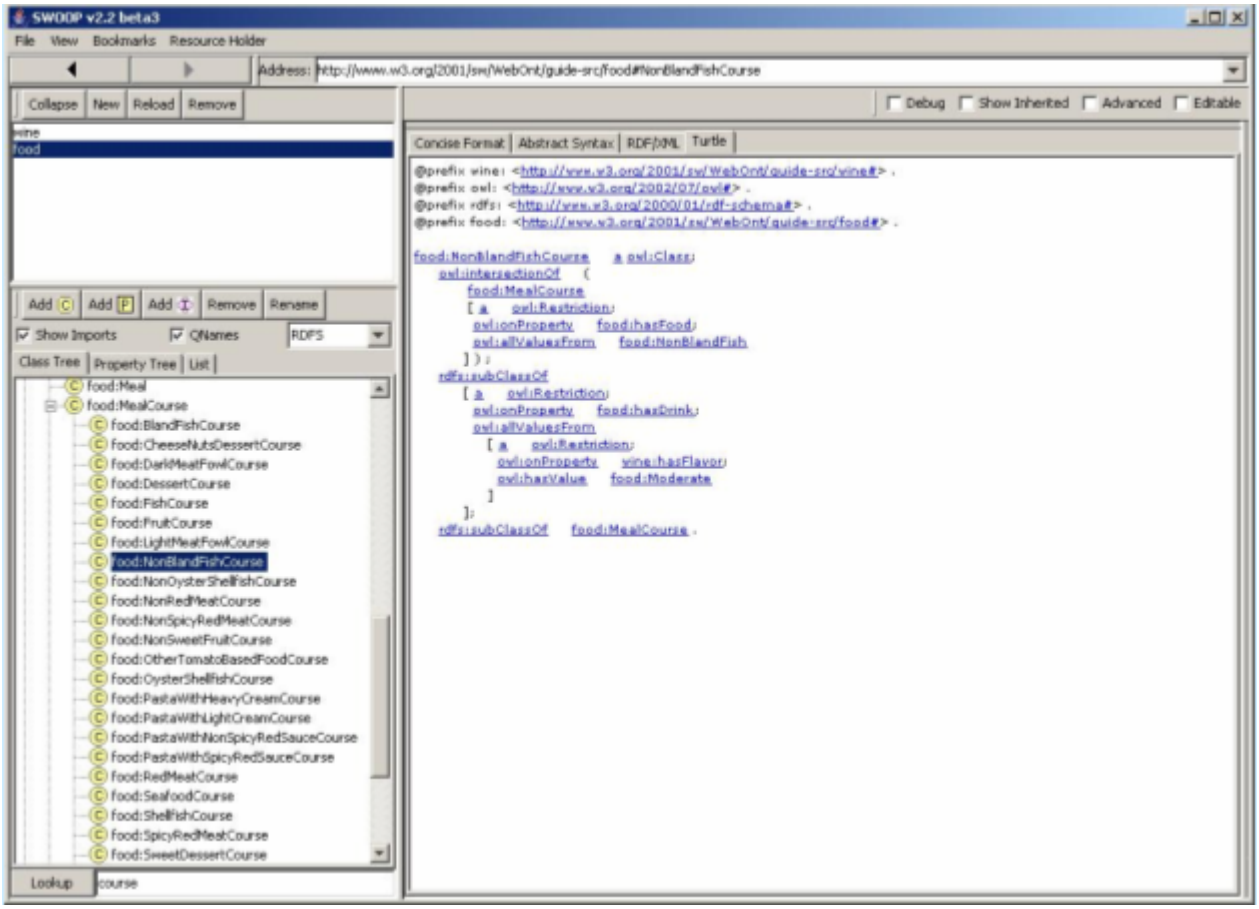

Figura 7. Swoop (Entorno de la versión 2.2 beta)

3.2.5. WebOnto: Desarrollado por el Knowledge Media Institute (KMI) de la Open University (Reino Unido). Aunque es de libre acceso, para acceder y disfrutar de sus potencialidades de edición, los usuarios deben solicitar una cuenta a los administradores. Es un applet (pequeña aplicación que permite obtener una gran variedad de efectos en las páginas web) de Java, trabaja junto a un servidor Web personalizado que permite a los usuarios navegar, crear y editar ontologías sin problemas de interfaz. WebOnto ofrece la gestión gráfica de ontologías e inspección de elementos, teniendo en cuenta la herencia de propiedades y el chequeo de consistencia; además, permite la generación automática de instancias a partir de definiciones de clases.

3.2.6. Ontolingua: desarrollada por el Knowledge Systems Laboratory (KSL) de la Universidad de Stanford. Facilita el desarrollo colaborativo de ontologías y proporciona un repositorio de las mismas. Proporciona un entorno distribuido y colaborativo para la creación, edición, modificación, navegación y utilización de ontologías mediante la Web. Incluye una API para integrar las ontologías del servidor con agentes preparados para Internet y su diseño modular, ofrece un conjunto de librerías que permiten a los usuarios ensamblar rápidamente una nueva ontología.

3.2.7. Chimaera: esta herramienta permite crear y mantener ontologías en la Web. Proporciona un ambiente distribuido para navegar, crear, editar, modificar y usar ontologías. Inicialmente fue pensado su integración con Ontolingua. Utiliza diferentes formatos para la carga de las bases de conocimientos. Reorganiza taxonomías y resuelve conflictos de nombres y edición de términos. Facilita la combinación permitiendo al usuario subir ontologías a su espacio de trabajo.

A continuación, se exhiben las principales características y los WebSite para cada herramienta mencionada anteriormente (ver Tabla 3):

Tabla 3. Principales características y Website de algunos editores

Herramienta	Principales características	WebSite
Protegé	Código abierto, *plug-in*, edición gráfica de taxonomías de conceptos y axiomas formales, chequeo de consistencia, interoperable	http://protege.stanford.edu/
KAON	Código abierto, servidor de aplicaciones, frames como formalismo de representación del conocimiento, edición colaborativa, chequeo de consistencia, escalable, eficiente e interoperable	http://kaon.semanticweb.org/
WebODE	Libre acceso, servidor de aplicaciones, interoperable, edición gráfica de taxonomías de conceptos y axiomas formales, chequeo de consistencia, edición colaborativa, utiliza motor de inferencias, chequeo de consistencia	http://webode.dia.fi.upm.es/WebODEWeb/
Swoop	Código abierto, extensible, edición colaborativa, utiliza motor de inferencias	http://www.mindswap.org/2004/SWOOP/
WebOnto	Libre acceso, arquitectura cliente/servidor, edición gráfica de taxonomías de conceptos, interoperable, edición colaborativa, utiliza motor de inferencias, chequeo de consistencias	http://kmi.open.ac.uk./projects/webonto
Ontolingua	Libre acceso, edición colaborativa, utiliza motor de inferencias, interoperable	http://www.ksl.stanford.edu/software/ontolingua/
Chimaera	Flexible, robusto, ambiente distribuido, interoperable	http://www.ksl.tanford.du/software/chimaera.

4. Razonamiento con ontologías

Para extraer más conocimiento que los conceptos almacenados en una ontología, son necesarias otras herramientas como un razonador, los cuales permiten deducir nuevas relaciones o conceptos no explícitos en el modelo original de la ontología, permitiendo además realizar consultas sobre ella. Otro de los servicios ofrecidos por un razonador es probar si una clase es o no subclase de otra. También, permite comprobar la consistencia de una ontología, entre otros servicios. Existen distintos razonadores que permiten inferir a partir del contexto, destacan particularmente (ver Tabla 4): Pellet (http://clarkparsia.com/pellet), FaCT++ (http://owl.man.ac.uk/factplusplus/) y Racer (http://www.sts.tu-harburg.de/~r.f.moeller/racer/).

En el caso de las ontologías construidas en OWL, los razonadores permiten extraer información basándose en lógica descriptiva (DL) que es el lenguaje formal utilizado para la construcción de las mismas. Al respecto, en [24] señalan que: dado que la base de OWL es la lógica descriptiva y cuando se desarrolla una ontología OWL en definitiva se está realizando una representación formal de un dominio de aplicación en forma de sentencias lógicas (hechos y reglas), se hace necesario el uso de estas herramientas: razonadores.

En este sentido, los razonadores DL deben proporcionar los siguientes servicios de inferencia [25]:

(a) Chequear la consistencia de una ontología, es decir, debe ser capaz de asegurar que la ontología no contiene hechos contradictorios;

(b) Satisfacibilidad de los conceptos de la ontología, es decir, el razonador determina si es posible que una clase tenga instancias;

(c) Clasificación de la ontología. El razonador computa a partir de los axiomas declarados en la ontología las relaciones de subclase entre todos los conceptos declarados explícitamente para construir la jerarquía de clases.

(d) Instanciación de los conceptos de la jerarquía. Un razonador DL puede inferir cuáles son las clases a las que directamente pertenece. Si además se utiliza la jerarquía inferida mediante el servicio de clasificación anterior, es posible obtener todas las clases a las que indirectamente pertenece una instancia dentro de la ontología.

Tabla 4. Principales características de algunos razonadores

Razonador	Algunas características
Pellet	Razonador *Open source* para OWL-DL construido en JAVA, basado en los algoritmos *Tableau* desarrollados para Lógicas Expresivas potentes; soporta las nuevas características de la OWL 1.1.
FaCT++	Tiene licencia GPL y trabaja eficientemente con *TBox* de ontologías de tamaño grande y mediano; no tiene soporte para otros tipos de dato que no sean *string* o *integer* y tampoco para el razonamiento con la *A-Box* de la ontología.
Racer	Su nombre comercial es RacerPro. No tiene licencias libres. Razonador DL para la lógica descriptiva *SHIQ*.

4.1. Vista general del Lenguaje de Ontologías Web (OWL)

OWL (http://www.w3.org/TR/2004/REC-owl-features-20040210) es un lenguaje propuesto por la W3C en el año 2004, como estándar para definir ontologías. Es una extensión semántica de RDF del que aprovecha la semántica definida para clases y propiedades, pero al que añade nuevos constructores más potentes y expresivos que permiten superar algunas de sus limitaciones expresivas.

OWL ofrece, por lo tanto, medios para especificar de una manera más precisa la semántica formal de un ámbito determinado de la realidad, y hace posible el

razonamiento con las ontologías para inferir conocimiento sobre diferentes aspectos de los elementos descritos.

OWL proporciona tres lenguajes, cada uno con nivel de expresividad mayor que el anterior, diseñados para ser usados por comunidades específicas de desarrolladores y usuarios:

- ***OWL Lite**:* dirigido a usuarios que necesitan principalmente una clasificación jerárquica y restricciones simples.
- ***OWL DL***: diseñado para aquellos usuarios que necesitan la máxima expresividad conservando completitud computacional y resolubilidad. Incluye todas las construcciones del lenguaje de OWL, pero sólo pueden ser usados bajo ciertas restricciones.
- ***OWL Full:*** dirigido a usuarios que necesitan máxima expresividad y libertad sintáctica sin garantías computacionales.

En la siguiente sección se hará referencia a los elementos básicos del lenguaje OWL y se presentará como ejemplo la ontología Universidad construida con *Protégé*. Es importante señalar que para aquellos términos presentes en RDF o en RDF Schema se usarán los prefijos "rdf:" o "rdfs:" respectivamente y para aquellos que pertenecen a OWL se usará el prefijo owl.

4.1.1. Elementos básicos:

Entro los elementos básicos de una ontología OWL se encuentran las clases y sus propiedades, características e instancias de clases y las relaciones entre estas instancias. A continuación, se describen estos elementos y se muestra para cada uno su correspondiente ejemplo en la ontología Universidad diseñada en Protégé 3.4.4.

Clases e Individuos:

- ***owl:Class****:* una clase define un grupo de individuos que pertenecen a la misma porque comparten algunas propiedades. En OWL existen dos clases predefinidas: una llamada Thing y otra Nothing, la primera es una clase de todos los individuos y superclase de todas las clases de OWL, y la segunda, una clase vacía, no tiene instancias y es una subclase de todas las clases de OWL (ver Figura 8).

  Una definición de clase incluye un nombre y una lista de restricciones. En este sentido, las instancias de la clase pertenecen a la intersección de las restricciones.

- ***rdfs:subClassOf****:* las jerarquías de clase deben crearse indicando que una clase es subclase de otra. Por ejemplo, las clases “Contratado” y “Ordinario” podrían definirse como subclase de la clase “ProfesorDepartamento” (ver Figura 8).

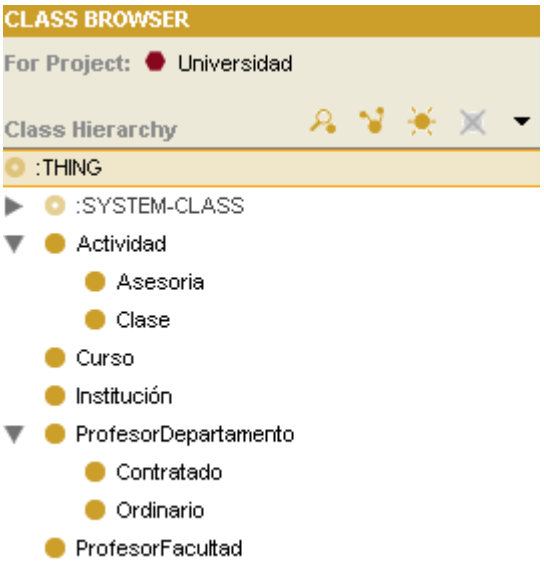

Figura 8. Ontología Universidad

- ***owl:individual:*** los individuos son instancias de clases y las propiedades pueden ser usadas para relacionar un individuo con otro. Por ejemplo, un individuo llamado “María” puede ser definido como una instancia de la clase “ProfesorDepartamento” y la propiedad “TrabajaEn” puede ser usada para relacionar el individuo “María” con el individuo “UCV”, siendo ésta última, instancia de una clase denominada “Institución”.

  OWL marca la diferencia entre una clase y un individuo. Una clase es simplemente un nombre y una colección de propiedades que describen un conjunto de individuos. Y los individuos son entidades reales, miembros de estos conjuntos (ver Figura 9).

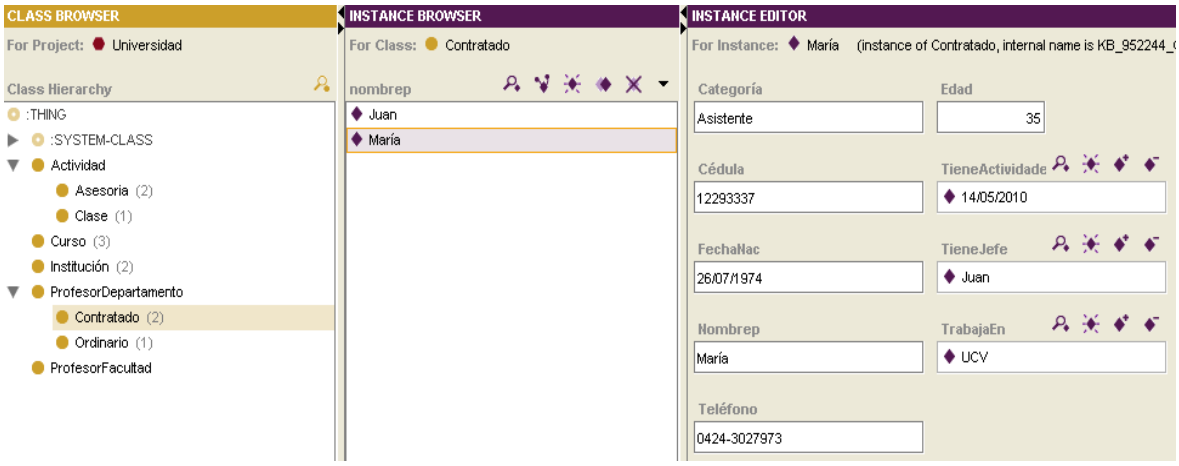

Figura 9. Instancias de la subclase Contratado

- ***owl:disjoinWith:*** esta característica denota clases que no tienen elementos comunes. Por ejemplo, la clase “Contratado” es disjunta de la clase “Ordinario”, entre estas clases no hay elementos (instancias) comunes (ver Figura 9).

- ***owl:equivalentClass**: dos clases se pueden definir como equivalentes si tienen las mismas instancias. El valor de igualdad puede ser utilizado para crear clases sinónimas. Para el caso de la ontología Universidad, si las clases ProfesorDepartamento y ProfesorFacultad tuvieran las mismas instancias pudieran definirse como equivalentes.*

Propiedades Simples:

Las propiedades son relaciones binarias que permiten afirmar hechos generales sobre las clases y hechos específicos sobre los individuos que le pertenecen. En OWL existen dos tipos de propiedades:

- ***owl:ObjectProperty:*** se usa para establecer relaciones entre instancias de dos clases. Por ejemplo, se tiene la clase "Curso" y se establece que una instancia de ella se relaciona con una instancia de la clase "ProfesorDepartamento" a través de la propiedad "EsImpartidoPor" (ver Figura 10)

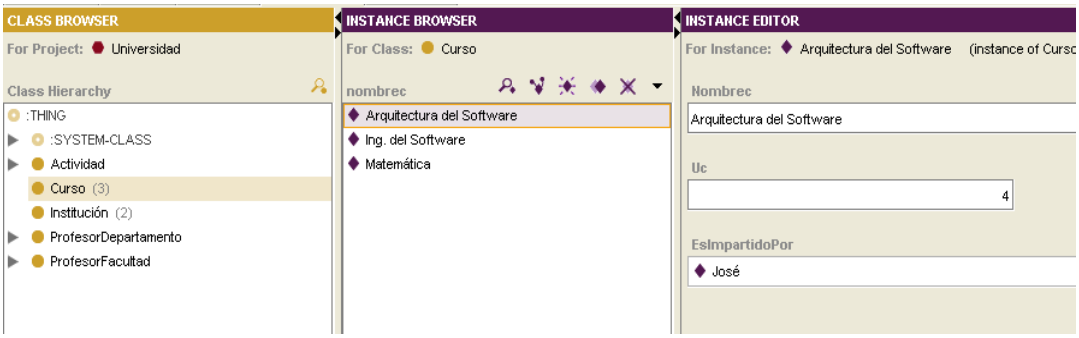

Figura 10. Propiedades entre instancias

- ***owl:DatatypeProperty:*** se usa para relacionar objetos con valores de tipo de datos. Algunos ejemplos pueden ser teléfono, edad, fecha de nacimiento, entre otros (ver Figura 11)

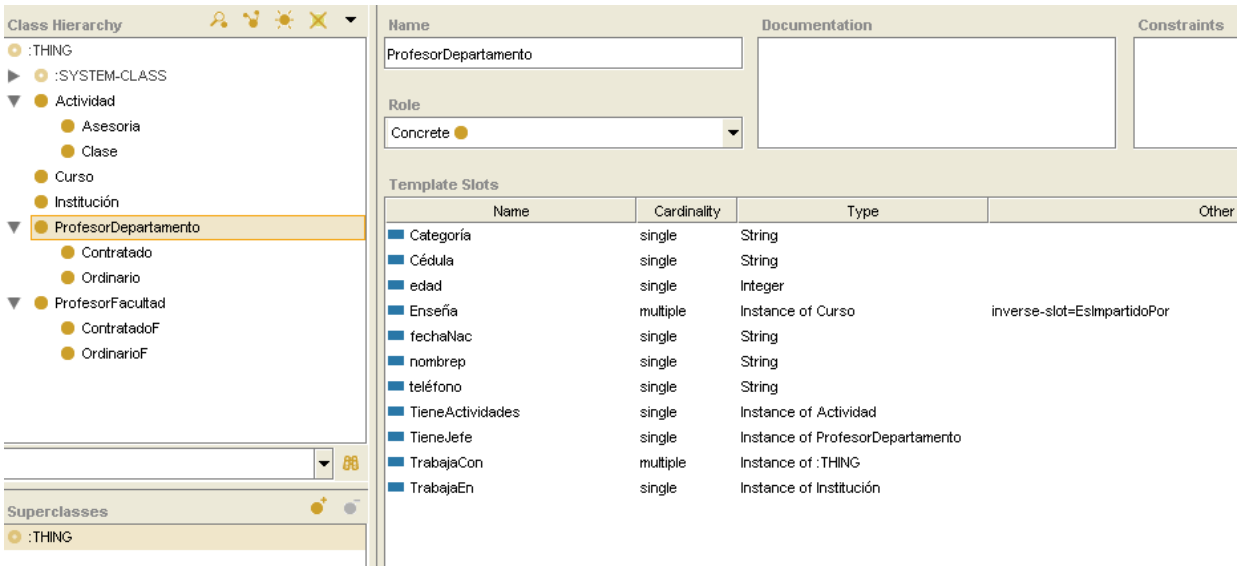

Figura 11. Propiedades de la clase ProfesorDepartamento

OWL utiliza gran parte de los tipos de datos definidos en XML Schema. Los siguientes son los tipos de datos recomendados por OWL: *string*, decimal, *integer, long, time, date, boolean, double, short, byte, gDay, gMonthDay, gYearMonth, gMonth*, entre otros.

Cuando se define una propiedad, deben especificarse el dominio y el rango, a fin de restringir la relación:

- ***rdfs:domain:*** el dominio indica a que individuos puede aplicarse la propiedad. Si una propiedad relaciona un individuo con otro individuo, y la propiedad tiene una clase como uno de sus dominios, entonces el individuo debe pertenecer a esa clase. Para la propiedad TieneActividades entre las clases ProfesorDepartamento y Actividad de la ontología Universidad, el dominio es la clase ProfesorDepartamento (ver Figura 12).

- ***rdfs:range****:* el rango indica los individuos que una propiedad puede tener como su valor. Si una propiedad relaciona a un individuo con otro individuo, y ésta como rango a una clase, entonces el otro individuo debe pertenecer a dicha clase. Para la propiedad TieneActividades entre las clases ProfesorDepartamento y Actividad de la ontología Universidad, el rango es la clase Actividad (ver Figura 12)

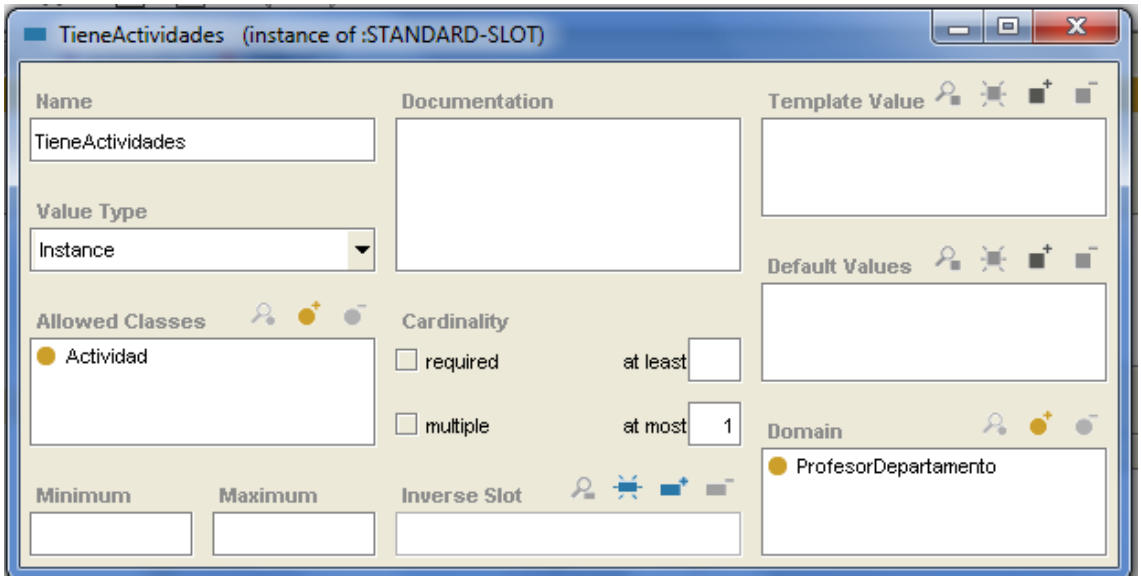

Figura 12. Dominio y Rango (Allowed Classes) de la propiedad TieneActividades

Una propiedad puede definirse como una especialización (subpropiedad) de otra propiedad existente:

- ***rdfs:subPropertyOf****:* permite establecer jerarquías de propiedades, esto se hace indicando que una propiedad es a su vez subpropiedad de una o más propiedades. Por ejemplo, “TieneAdministrativas” puede ser una subpropiedad de “TieneActividades”.

Las siguientes características están relacionadas con los valores de igualdad y desigualdad:

- ***owl:equivalentProperty****:* dos propiedades se definen como equivalentes si se relaciona a un individuo con el conjunto de otros individuos similares.
- ***owl:sameAs****:* dos individuos pueden definirse como iguales. Estas construcciones pueden utilizarse para crear nombres diferentes que se refieren al mismo individuo.
- ***owl:differentFrom****:* un individuo puede definirse como diferente de otros individuos. Establecer explícitamente que los individuos son diferentes entre sí permite que un razonador pueda deducir que ambos hacen referencia a individuos distintos.
- ***owl:allDifferent****:* permite definir que un número de individuos son mutuamente distintos. Esta construcción es particularmente útil cuando hay conjuntos de objetos distintos. Asimismo, se usa distinctMembers para establecer que todos los miembros de una lista son distintos y disjuntos en pares.

Características de las propiedades:

Es posible especificar las características de una propiedad, a través de un poderoso mecanismo para mejorar el razonamiento sobre la misma. A continuación, se describen algunos identificadores especiales que se utilizan para proporcionar información referente a las propiedades y a sus valores:

- ***inverseOf****:* se puede definir una propiedad P1 como la inversa de otra propiedad P2. En este sentido, si se establece la relación X con Y mediante la propiedad P2, entonces Y estaría relacionado con X mediante la propiedad P1. Por ejemplo, si la propiedad “EsImpartidoPor” es la propiedad opuesta de “Enseña” y “María Enseña Matemática”, entonces un

razonador puede deducir que “Matemática” “EsImpartidoPor” “María” (ver Figura 13).

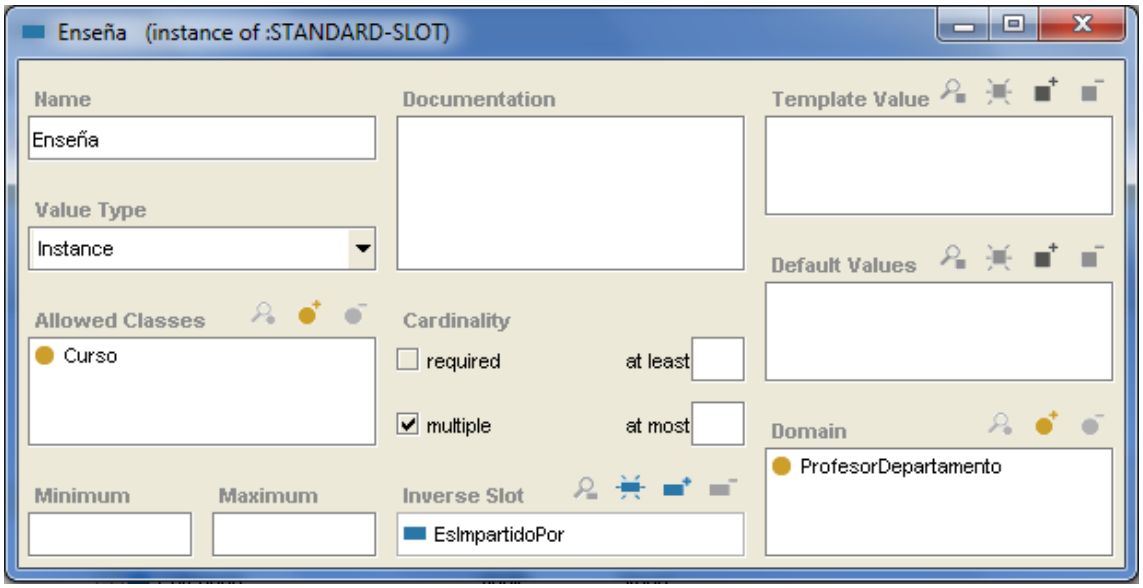

Figura 13. Propiedad Enseña y su inversa EsImpartidoPor

- ***TransitiveProperty***: se pueden definir propiedades como transitivas si el par (X, Y) es una instancia de la propiedad transitiva P, y el par (Y, Z) es otra instancia de la propiedad transitiva P, entonces el par (X, Z) también es una instancia de P. Por ejemplo, se tiene el par (María, Juan) instancia de la propiedad “TieneJefe”, y el par (Juan, José) otra instancia de la misma propiedad, entonces el par (María, José) también es una instancia de la propiedad definida. Al respecto, un razonador puede deducir que si “María TieneJefe Juan” y “Juan TieneJefe José” entonces “María TieneJefe José”.

- ***SymmetricProperty***: si una propiedad P es simétrica, y el par (x, y) es una instancia de esa propiedad, entonces el par (y, x) es también una instancia de la propiedad simétrica P. Por ejemplo, la propiedad “TrabajaCon” puede

considerarse simétrica. Al respecto, si se indica a un razonador que “María” TrabajaCon “José”, éste puede deducir que “José” TrabajaCon “María”.

- ***FunctionalProperty***: permite definir propiedades para que tengan un valor único, es decir, no tendrán más de un valor para cada individuo e indica que la cardinalidad mínima de la propiedad es 0 y la máxima es 1. Por ejemplo, “TieneJefe” puede establecerse como FunctionalProperty. En este contexto, un razonador puede deducir que ningún individuo pueda tener más de un Jefe, sin embargo, esto no implica que todas las instancias de “ProfesorDepartamento” deban tener al menos un Jefe.

- ***InverseFunctionalProperty***: es posible definir propiedades para que sean funcional inversa. Si una propiedad es funcional inversa, entonces la inversa de la propiedad será funcional. Por tanto, la inversa de la propiedad tiene como máximo un valor para cada individuo. Esta característica también se denomina propiedad inequívoca.

  Por ejemplo, “Cédula” (un identificador único) puede establecerse como funcional inversa. La inversa de esta propiedad (al que se puede llamar “EsCédulaDe”) tiene como máximo un valor para cada individuo dentro de la clase de los números de Cédula. De esta manera, el número de la cédula de cualquier “ProfesorDepartamento” es el único valor para su propiedad “EsCédulaDe”. Así un razonador puede deducir que no existen dos individuos diferentes, instancias de “ProfesorDepartamento”, que tengan el mismo número de Cédula. Además, un razonador puede deducir que si dos instancias de “ProfesorDepartamento” tienen el mismo número de Cédula, entonces dichas instancias se refieren al mismo individuo.

Restricciones de las propiedades:

Además de definir las características de una propiedad, es posible delimitar aún más el rango de una propiedad en contextos específicos. OWL Lite permite establecer restricciones sobre la forma en que las propiedades son utilizadas por las instancias de una clase. Estos elementos se usan dentro del contexto de una restricción de esta forma owl:Restriction y el elemento owl:onProperty indica la propiedad restringida. A continuación se describen las dos restricciones:

- ***allValuesFrom***: restricción de rango local asociada a una clase y se establece sobre una propiedad con respecto a dicha clase. Se utiliza para especificar la clase de valores posibles que puede tomar la propiedad.

- ***someValuesFrom***: esta restricción también se establece sobre una propiedad con respecto a una clase. Una clase particular puede tener una restricción sobre una propiedad que haga que al menos un valor para esa propiedad sea de un tipo concreto. A diferencia de *allValuesFrom*, *someValuesFrom* no restringe que todos los valores de una propiedad sean instancias de una misma clase.

Restricciones de Cardinalidad

Las restricciones de cardinalidad se refieren a restricciones locales, ya que se definen en las propiedades con respecto a una clase específica. Son limitadas, permiten realizar indicaciones referentes a cardinalidades de valor 0 ó 1:

- ***minCardinality***: si se establece cardinalidad mínima de 0 o 1 sobre una propiedad con respecto a una clase, entonces cualquier instancia de esa clase estará relacionada con cero o al menos con un individuo mediante esta propiedad.

- ***maxCardinality***: si se establece cardinalidad máxima de 1 sobre una propiedad con respecto a una clase, entonces cualquier instancia de esa clase estará relacionada como máximo con un individuo mediante dicha propiedad.

Estas expresiones permiten al usuario indicar “al menos uno”, “no más de uno” y “exactamente uno”; owl:maxCardinality se puede utilizar para especificar un límite superior y owl:minCardinality para especificar un límite inferior. La combinación de ambas pueden ser utilizadas para limitar la cardinalidad de la propiedad a un intervalo numérico. Es útil establecer que una propiedad tiene sobre una clase *minCardinality* 0 y *maxCardinality* 0, o ambos *minCardinality* 1 y *maxCardinality* 1.

OWL además permite especificar clases basadas en la existencia de valores de propiedades particulares. Por lo tanto, un individuo será un miembro de esa clase cada vez que al menos uno de los valores de su propiedad sea igual al recurso hasValue.

- ***hasValue:*** establece que una propiedad necesariamente tenga un determinado individuo como un valor. Por ejemplo, instancias de la clase ProfesorDepartamento pueden ser caracterizadas como los profesores que tiene Informática como valor de su ubicación. (El valor de la ubicación Informática, es una instancia de la clase Departamentos).

Combinaciones booleanas

En OWL, es posible realizar las siguientes combinaciones booleanas: unión (*unionOf*), intersección (*intersectionOf*) y complemento *(complementOf)* de las clases. Estos operadores básicos permiten manipular los conjuntos constituidos por los individuos que son miembros de las clases (extensiones de las clases).

Enumeraciones

OWL proporciona los medios para especificar una clase a través de una enumeración directa de sus miembros. Esto se hace utilizando la construcción oneOf y especifica completamente la extensión de clase.

- ***oneOf:*** permite describir las clases mediante la enumeración de los individuos que la componen, pues los miembros de la clase son exactamente el grupo de los individuos enumerados. Por ejemplo, la clase horarioSemana puede describirse simplemente enumerando los individuos Lunes, Martes, Miércoles, Jueves, Viernes, Sábado, Domingo. De esta forma, un razonador puede deducir la cardinalidad máxima (7) de cualquier propiedad que tenga horarioSemana como restricción de allValuesFrom.

4.1.2. Anotaciones

OWL DL permite anotaciones en las clases, las propiedades, los individuos y los encabezados de la ontología. Por su parte, OWL Full no tiene ninguna restricción sobre las anotaciones en una ontología OWL.

Los conjuntos de propiedades de los objetos, las propiedades de tipo de datos, las propiedades de anotación y las propiedades de la ontología deben ser mutuamente disjuntos.

Asimismo, el objeto de una propiedad debe ser un campo literal, una referencia URL o un individuo. Tiene una sintaxis explícita de la forma:

AnnotationPropertyID rdf:type owl:AnnotationProperty

OWL define las siguientes cinco propiedades de anotación:

- ***owl:versionInfo:*** permite declarar información general sobre la versión de la ontología.
- ***rdfs:label:*** permite declarar etiquetas en una ontología.
- ***rdfs:comment***: permite incluir comentarios en la ontología.

- ***rdfs:seeAlso*** (véase también)
- ***rdfs:isDefinedBy*** (definido por)

4.1.3. Cabeceras

Un documento que describe una ontología general, contiene información acerca de la propia ontología. La ontología puede describirse mediante las propiedades de anotaciones definidas en OWL y otros elementos como el encabezado que se encuentra típicamente al comienzo del documento de la ontología.

A continuación, se muestra un ejemplo de encabezado de un documento de ontología usando Protégé:

```
<owl:Ontology rdf:about="">
  <owl:versionInfo> ... </owl:versionInfo>
  <rdfs:comment>...</rdfs:comment>
  <owl:imports rdf:resource="..."/>
</owl:Ontology>
```

La línea <owl:Ontology rdf:about=""> describe la ontología actual, mientras que la declaración <owl:imports rdf:resource="..."/> referencia las definiciones de otra ontología, los cuales son considerados parte de la ontología importada. Cada referencia consiste de un URL que especifica desde donde la ontología se va a importar.

5. Investigación en ingeniería ontológica

Las ontologías han sido tradicionalmente usadas como modelo de representación del conocimiento consensuado y compartido en un campo de aplicación determinado. Durante los últimos años, investigadores han dirigido su atención en la creación, integración y reutilización de las mismas.Hasta la fecha, se han realizado numerosos desarrollos ontológicos, algunos de ellos son descritos en la sección 5.1. Seguidamente, se presenta una comparación entre el modelado de una ontología y el modelado de objetos.

5.1. Estudio y revisión de algunos desarrollos ontológicos de interés

5.1.1. Servidor de Conocimiento Cyc

El Servidor de Conocimientos Cyc (http://www.cyc.com/) fue desarrollado por la Cycorps. Este proyecto se inició en 1984 y actualmente es la base de conocimiento más grande del mundo, es multi-contexto y cubre conocimiento general de sentido común, es considerada una ontología de nivel superior. El conocimiento está a disposición de otros programas. La tecnología Cyc incluye los siguientes componentes:

(a) La Base de Conocimientos (BC Cyc): es una representación formal de una gran cantidad de conocimiento fundamental humano: unos cinco millones de hechos, reglas básicas y heurísticas para razonar acerca de los objetos y los acontecimientos de la vida cotidiana.
Actualmente, consta de miles de "micro teorías", cada una es esencialmente un conjunto de afirmaciones que comparten un conjunto común de supuestos, algunas se centran en un dominio particular del conocimiento, otras con un determinado nivel de detalle, otros se refieren a intervalos particulares en el tiempo, etc. El mecanismo de “micro teorías” permite que la BC mantenga de forma independiente afirmaciones que son contradictorias a primera vista.

(b) El Motor de inferencia: realiza deducción lógica general empleando mecanismos de inferencia como la herencia, clasificación automática, entre otros. Realiza la búsqueda sobre un espacio de pruebas empleando el algoritmo *Primero el Mejor* con un conjunto de propiedades heurísticas y micro teorías para optimizar el proceso de inferencia mediante restricciones en los dominios de búsqueda.

(c) El Lenguaje de Representación (CyCL): se basa esencialmente en el cálculo de predicados de primer orden, con extensiones para manejar la igualdad, el razonamiento por defecto y algunas características de segundo orden.

(d) El Subsistema de Procesamiento de Lenguaje Natural (CyC-NL): está formado por tres componentes: el léxico, el analizador sintáctico y el intérprete semántico. Actualmente, Cyc-NL puede analizar correctamente muchas frases diferentes, incluyendo entradas sintácticamente ambiguas y complejas. Cyc es capaz de manejar la negación, modales, y cuantificadores anidados. Asimismo, se están desarrollando interfaces que permitan a las personas hacer afirmaciones y consultas.

(e) El Bus de Integración Semántica: se encarga de convertir información en conocimiento útil. Esta información puede estar almacenada en el computador de muchas formas: datos estructurados como las bases de datos, semiestructurados como las hojas de cálculos y páginas web, y no estructurados como archivos de texto y campos de texto. Así por ejemplo, Cyc (SUBIR ACA, PERO NO ME DEJA) trata cada registro de base de datos como una afirmación implícita en la BC encontrándose disponible durante la inferencia. Asimismo, los campos de texto pueden ser leídos usando el procesador de lenguaje natural con el propósito de verificar si implícitamente contienen alguna afirmación útil.

(f) Conjuntos de herramientas para desarrolladores: el sistema Cyc también incluye una variedad de herramientas que permiten al usuario navegar, editar y ampliar la BC Cyc, plantear consultas al motor de inferencia, interactuar empleando lenguaje natural y módulos de integración de base de datos.

OpenCyc es la versión de código abierto de la tecnología Cyc, cuenta con unos 6.000 conceptos y 60.000 afirmaciones. Cycorp ofrece esta ontología sin costo alguno e invita a hacer uso de ella según las necesidades. La siguiente URL http://www.cyc.com/cycdoc/vocab/upperont-diagram.html muestra un diagrama de la ontología. (ver Figura 14)

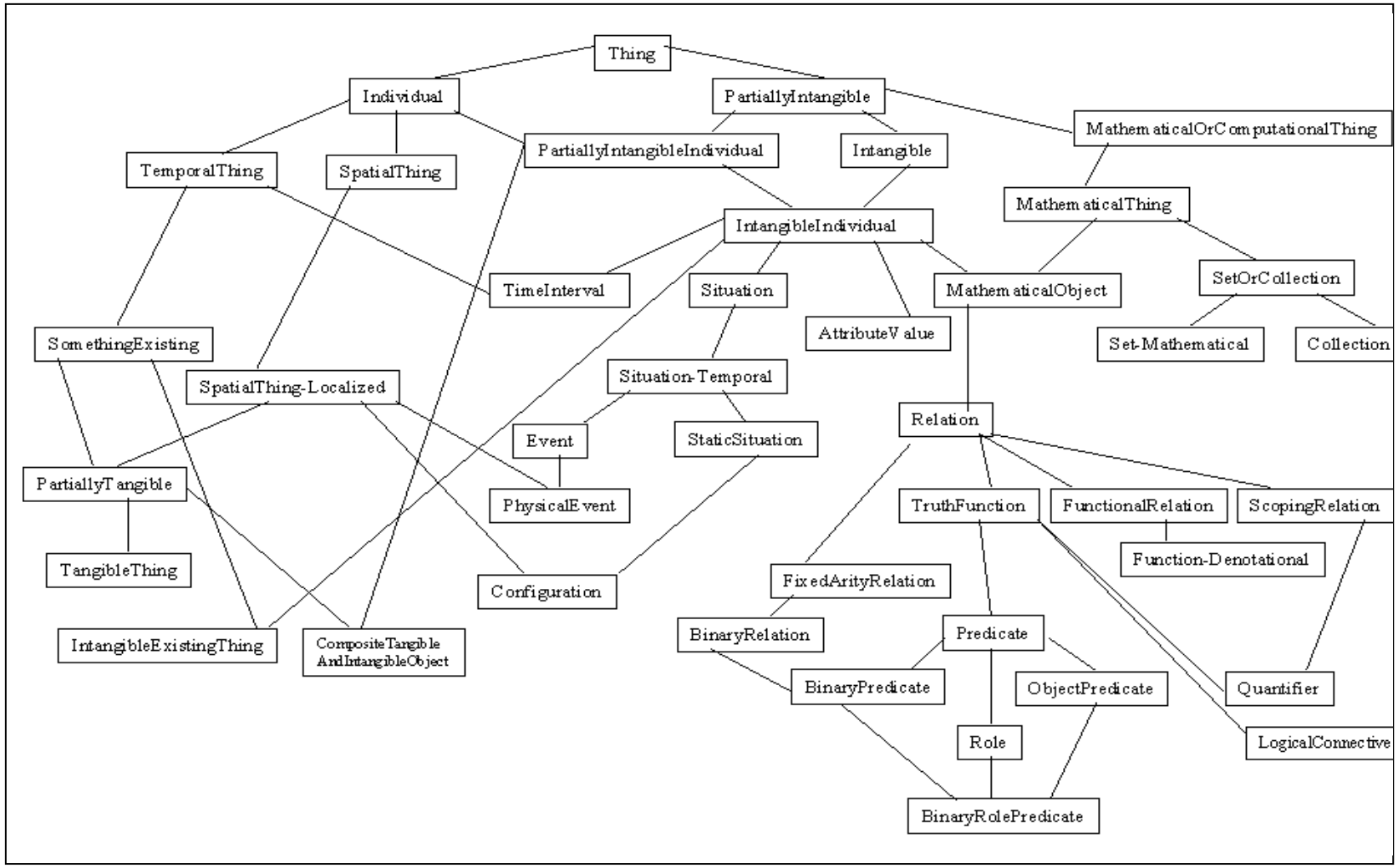

Figura 14. Diagrama de la Ontología Cyc.

5.2.2. Wornet

WordNet fue desarrollado por el *Cognitive Science Laboratory* de la Universidad de Princeton bajo la dirección del profesor de psicología George A. Miller. El desarrollo comenzó en 1985. Es un sistema de referencia léxica y se considera una ontología de nivel superior.

El diseño de WordNet está en consonancia con teorías psicolingüísticas relativas a la organización de la información léxica en la mente del hablante [25]. Sus objetivos fundamentales son:

(a) La validación de las teorías psicolingüísticas sobre organización léxica;

(b) Su utilización en diversas aplicaciones que requieran acceso a información léxica.

Siguiendo un modelo de redes semánticas, la herramienta permite al usuario moverse por la estructura de un diccionario conceptualmente, además de utilizar la estructura del alfabeto. Divide el lexicón en cinco categorías sintácticas: nombres, verbos, adjetivos, adverbios y elementos funcionales. Este tipo de organización, facilita el análisis de las diferencias de organización semántica que existen entre estas categorías.

WordNet es extraordinario en cuanto a la cantidad de información que contiene en formato electrónico, sin embargo no puede considerarse como un repositorio de conocimiento léxico detallado, sino como una representación de las diferentes relaciones semánticas que existen entre elementos léxicos.

La versión 2 de WordNet está disponible en http://www.cogsci.princeton.edu/wn/wn2.0.shtml

5.2.3. Enterprise Ontology (EO)

EO es una colección de términos y definiciones importantes para las empresas comerciales. Fue desarrollado en 1996 por Mike Uschold y el grupo de Inteligencia Artificial de la Universidad de Edinburg [5]. El propósito de EO se resume como sigue a continuación:

(1) Garantizar una comunicación fluida entre los participantes facilitando la cooperación en el entendimiento consensuado sobre el modelo de empresa al proporcionar vocabulario suficiente y necesario.
(2) Proveer una infraestructura estable y al mismo tiempo adaptable a los cambios del modelo de empresa.
(3) Aumentar la interoperabilidad de los distintos programas de aplicación de un modelo de empresa para el intercambio de la información.

Conceptualmente, la ontología de empresa se divide en varias secciones, los cuales se resumen a continuación:

(a) Actividades y procesos: permite capturar la noción de todo lo que implica el hacer, incluyendo la acción. El concepto de actividad está estrechamente vinculada con la idea del hacedor, que puede ser una persona, organización o máquina. También se relaciona con recursos (algo que se usa o consume) y tiempo (intervalo). Una actividad también puede tener resultados o efectos. Asimismo, una actividad muy grande y compleja puede ser representada como la composición de muchas sub-actividades.
Una Actividad puede haber ocurrido (en el pasado) o estar ocurriendo (en el presente). También puede referirse a una actividad hipotética (en el futuro). En cualquiera de las tres ocurrencias, hay una necesidad de referirse explícitamente a datos específicos o proyectos para actividades, imperando la necesidad de hacer referencia de manera explícita a las especificaciones o planes de actividades (una o más actividades con un propósito).

(b) Organización: esta sección comprende conceptos jurídicos de las entidades y unidades organizativas (UO). Se diferencian en que una entidad jurídica se reconoce como titular de derechos y deberes en el mundo en general, mientras que las UO sólo tienen el pleno reconocimiento de una organización.

(c) Estrategia: el concepto central de esta sección es el objetivo. El objetivo captura la idea de algo que un plan puede ayudar a alcanzar o que una UO puede ser responsable de alcanzarlo. En este contexto, la estrategia es definida como un plan para alcanzar un objetivo de alto nivel. Basado en el concepto de plan de la sección de actividades, los términos claves para la planificación estratégica se puede representar con los vocablos decisión, asunción, riesgo, y algunos tipos de factores.

(d)Marketing: el concepto central de esta sección es la venta. Una venta es un acuerdo entre dos entidades legales para el intercambio de un producto (puede ser un bien o servicio) y el precio de la misma tiene un valor monetario, sin embargo pueden incluirse otras posibilidades. Las entidades jurídicas desempeñan los roles (por lo general distintos) del vendedor y el cliente. Asimismo, el mercado incluye ventas y ventas potenciales dentro de un ámbito de interés. También puede contemplar ventas en un escenario de competidores. Se habla además de segmentos de mercados, así como de su análisis que implica una comprensión de las características de los productos, necesidades de los clientes, e imágenes de marcas, productos o proveedores. De la misma forma, se consideran las promociones como actividades cuyos fines se relacionan con la imagen en un mercado.

La URL http://www.aiai.ed.ac.uk/project/enterprise/enterprise/ontology.html muestra una lista completa de los términos definidos en la ontología.

5.2.4. Gene Ontology

A diario se produce información en diversos temas del campo de la biología, tales como: genoma celular, estructura, fenotipo, entre otros y cuando se expresa ésta información las diferencias sintácticas y semánticas se hacen evidentes, lo que impide a los investigadores la recuperación y utilización pertinente de la información para facilitar su actividad investigadora. A fin de proporcionar un vocabulario común, la ontología juega un papel importante en este campo.

Particularmente, Gene Ontology (GO) es producto de un esfuerzo colaborativo para abordar la necesidad de descripciones coherentes sobre productos genéticos en diferentes bases de datos. El proyecto comenzó en 1998 como una colaboración entre tres bases de datos de organismos modelos: FlyBase (Drosophila), la Saccharomyces Genome Database (SGD) y Mouse Genome Database (MGD). Desde entonces, el Consorcio GO ha crecido hasta incluir a muchas bases de datos, entre ellas varios de los depósitos más importantes del mundo de plantas, animales y genomas microbianos.

El proyecto GO (http://www.geneontology.org/GO.doc.shtml) ha desarrollado tres vocabularios controlados estructurados (ontologías) que describen los productos genéticos en términos de sus procesos biológicos asociados, los componentes celulares y funciones moleculares de las especies de manera independiente. Hay tres aspectos distintos a este esfuerzo: en primer lugar, el desarrollo y mantenimiento de las propias ontologías; en segundo lugar, la anotación de los productos de los genes, lo que implica establecer relaciones entre las ontologías y los genes y productos génicos en las bases de datos que colaboran, y el tercero, el desarrollo de herramientas que faciliten la creación, mantenimiento y uso de ontologías.

5.2. Modelado de Ontologías vs Modelado de Objetos

Una ontología hace referencia a una formalización de los conocimientos en un dominio específico. Por otra parte, un modelo de objetos representa conceptualmente a los datos requeridos por una base de datos. Existen algunas similitudes y diferencias entre ambos modelos, por lo que esta sección tiene como propósito establecer una comparación entre los elementos que los describen. Esta comparación fue propuesta por [27].

- **Conceptos, Clases y Objetos**

Los conceptos en una ontología pueden ser objetos y relaciones en un dominio de interés. Los conceptos que son objetos se representan mediante clases, gramaticalmente son sustantivos; para el caso de las relaciones son verbos. Asimismo, en el modelado de datos los conceptos refieren objetos y relaciones para una aplicación particular, y a la colección de objetos también se les atribuye el término clase.

Es importante destacar, que en ambos casos, la clase reúne un conjunto de objetos (físicos ó lógicos) que poseen características comunes y cada objeto es una instancia de la clase misma. Específicamente en una ontología, las clases pueden contener individuos (instancias), otras clases más específicas (subclases) o una combinación de ambos. Por lo general se les aplica mecanismos de herencia.

Para el modelado de datos, las clases se definen mediante su estructura interna y la especificación de su comportamiento. En un diagrama de clases, estas son representadas por una caja que contiene tres elementos: el nombre de la clase, sus atributos (nombre, tipo y visibilidad) y las operaciones.

- **Slot/Facets y Atributos/Propiedades**

En el modelado de una ontología, los slots son atributos que describen a los conceptos y las facetas pueden ser especificaciones, rangos o restricciones sobre los valores de los atributos. Asimismo, estos atributos y sus restricciones son heredados por las subclases y las instancias de la clase. Al respecto, un slot puede ser usado para describir una relación entre dos conceptos, uno finge como dominio y el otro como rango.

Del mismo modo, en el modelado de dato los atributos describen a las clases. Existe un atributo clave que identifica unívocamente a la clase y el resto se

exhiben como no claves. Algunos atributos son de tipo primitivo (enteros, booleanos, etc), otros son referencias a objetos. En este sentido, las propiedades de una clase son sus atributos y operaciones.

- **Instancias**

Para ambos modelos, un objeto específico de un concepto es una instancia. En una ontología, también se hace referencia al término individuo.

- **Relaciones**

Para ambos casos, las relaciones se establecen entre conceptos para representar las interacciones entre éstos. En el modelado de la ontología, una relación entre dos conceptos se llama slot o relación binaria (Ejemplo: “subclase de”, “conectado a”, “es un”, “es parte de”). Asimismo, una relación entre el concepto n y los conceptos n-1 se denomina función. Otra relación muy utilizada es aquella que permite la asociación entre objetos y clases (instancia de). Las relaciones exhiben una estructura jerárquica compleja y bien diseñada.

De la misma forma, en el modelado de datos a través de las definiciones de herencia se utiliza la relación “es un” y otras relaciones de pertenencia a una clase. Además, permite construir una jerarquía de clases entre las cuales se encuentra el supertipo (superclase) con atributos comunes y los subtipos (subclase) con atributos exclusivos para cada uno de ellos. Cada subtipo hereda los atributos del supertipo.

- **Axiomas**

Este elemento sólo es usado en el modelado de ontologías, describen sentencias que son siempre ciertas y permiten junto con la herencia de conceptos, inferir conocimiento que no aparece explícitamente en la taxonomía de conceptos.

- **Funciones/Métodos/Operaciones**

Como ya se mencionó, una función en el modelado de una ontología es un tipo especial de relación. En el modelado de datos, forma parte de las clases o de las propiedades y contienen meta información, como comentarios, limitaciones y valores por defecto.

- **Estándares y lenguajes**

Las tecnologías de apoyo para el modelado ontológico descansan en XML, RDF y OWL. Estos tres lenguajes describen los metadatos asociados a los objetos y son usados para comprender el significado de la información modelada.
En otro contexto, UML (Lenguaje Unificado de Modelado) y ODMG (Estándar del Grupo de Desarrollo de Modelos de Datos) se han desarrollado para los modelos orientados a objetos en la ingeniería del software.

Los modelos de ontologías y los modelos de datos se usan para describir un dominio específico. Sin embargo, una ontología representa tipos de cosas que existen y las reglas que las rigen, mientras, un modelo de objetos define los registros de las cosas y es el fundamento para el diseño de una base de datos. No toda la información de una ontología es necesaria para un modelo de objetos, y en algunos casos pueden utilizarse solo como referencia para su construcción. En este sentido, la ontología es más rica en información que un modelo de datos y ofrece un vocabulario controlado de conceptos.
En este sentido, una ontología se basa en términos relacionados solamente, mientras que un modelo de objetos debe contener de forma explícita las propiedades, relaciones y operaciones de los objetos del dominio modelado, por lo que la ontología pudiera ser una extensión de este modelo. Un modelo de objetos puede servir como punto de partida para el modelado de una ontología.

6. Conclusiones

Las ontologías son especificaciones formales que proporcionan una vía para representar conocimiento en un área de interés, con el propósito de facilitar la comunicación, reusar y compartir información entre personas, organizaciones y sistemas de computadores. La notable presencia alcanzada por las ontologías puede deberse a su consideración como recurso fundamental para la Web semántica [2]. Por tal razón, en las últimas décadas las investigaciones sobre este tema han cobrado gran relevancia.

Apoyado en la literatura especializada sobre ingeniería ontológica y artículos de referencia, este documento consiguió reunir información acerca de los aspectos fundamentales de las ontologías. Se revisaron algunas metodologías para su construcción, entre ellas Methontology y NeOn. Un aspecto de interés fue la presentación de una vista general del lenguaje OWL, el razonamiento con ontologías construidas en este lenguaje y el uso de razonadores, entre otros.

7. Referencias

[1] Mizoguchi R. Tutorial on ontological engineering - Part 1: Introduction to Ontological Engineering. New Generation Computing, OhmSha&Springer, Vol.21, No.4, pp.365-384. 2003.

[2] Berners-Lee Tim, Hendler James y Lassila Ora. The Semantic Web. Scientific American, Mayo 2001. pp. 28-37. 2001. Disponible en fecha enero de 2012 en: http://www.jeckle.de/files/tblSW.pdf

[3] Gruber T. Toward Principles for the Design of Ontologies Used for Knowledge Sharing. Available as Technical Report KSL 93-04, Knowledge Systems Laboratory, Stanford University. 1993. Disponible en fecha diciembre de 2011 en: http://citeseer.ist.psu.edu/viewdoc/summary?doi=10.1.1.122.3207l

[4] Ramos E. y Núñez H. Ontologías: componentes, metodologías, lenguajes y aplicaciones. RT 2007-12. Centro de Ingeniería de Software y Sistemas – YSYS- Laboratorio de Inteligencia Artificial –LIA- Universidad Central de Venezuela. ISSN 1316-6239. 2007. Disponible en fecha Octubre de 2011 en: www.ciens.ucv.ve/escueladecomputacion/documentos/archivo/51

[5] Uschold, M. Building Ontologies: Towards a Unified Methodology. Artificial Intelligence Application Institute. University of Edinburgh. Reino Unido. 1996. Disponible en fecha Diciembre 2011 en: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.39.9075

[6] Van Heijst, G., Schereiber, A. y Wielinga, B. Using Explicit Ontologies in KBS Development. International Journal of Human and Computer Studies, 46(2/3), pp. 293–310. 1997. Disponible en fecha enero de 20129 en: http://www.cs.vu.nl/~guus/papers/Heijst97a.pdf

[7] Guarino, N. Formal Ontology and Information Systems. Proceedings of FOIS ´98. National Research Council, LADSEB–CNR. 1998. Disponible en fecha setiembre de 2011 en: http://www.loa.istc.cnr.it/Papers/FOIS98.pdf

[8] Noy N. y McGuinness D. Ontology development 101: A Guide to creating your first ontology. Stanford University. Stanford knowledge Systems Laboratory. Technical Report KSL-01-05. 2001.

[9] Sowa, J. *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Pacific Grove, CA: Brooks Cole Publishing Co. 2000.

[10] Farquhar, A. Ontolingua Tutorial. University of Stanford. Knowledge Systems Lab Stanford University. California, Estados Unidos. 1997. Disponible en fecha Octubre de 2009 en: http://www-ksl.stanford.edu/people/axf/tutorial.pdf

[11] Gómez-Pérez, A., Fernández-López, M. y Corcho, M. Ontological Engineering. Springer Verlag, London. 2004.

[12] Beck, H. y Pinto, H. S. *Overview of Approach, Methodologies, Standards, and Tools for Ontologies*. The Agricultural Ontology Service. 2002. Disponible en fecha diciembre 2011 en: ftp://ftp.fao.org/gi/gil/gilws/aims/publications/workshops/AOS_3/ppt/BackgroundPaper.pdf

[13] Annamalai, M. y Sterling, L. Guidelines for Constructing Reusable Domain Ontologies. *AAMAS03 Workshop on Ontologies in Agent Systems Proceedings.* pp: 71-74. 2003. Disponible en fecha Diciembre 2009 en: http://oas.otago.ac.nz/OAS2003/papers/oas03-annamalai2.pdf

[14] Valente, A. y Breuker, J. Towards Principled Core Ontologies. P*roceedings of the Knowledge Acquisition Workshop - KAW'96*. 1996.

[15] Corcho, O., Fernández, M., Gómez-Pérez, A., y López, A. Building legal ontologies with METHONTOLOGY and WebODE. Law and the Semantic Web. Legal Ontologies, Methodologies, Legal Information Retrieval, and Applications. Springer-Verlag, LNAI 3369. 2005. Disponible en fecha diciembre 2011:http://www.cs.man.ac.uk/~ocorcho/documents/LawSemWeb2004_CorchoEtAl.pdf

[16] Hartman, J., Spyns, P., Giboin, A., Maynard, D., Cuel, R., Suárez-Figueroa, M. y Sure, Y. D1.2.3 Methods for Ontology Evaluation. Knowledge Web Consortium. Project Number IST-2004-507-507-482. 2005.

[17] Brewster, C., Alani, H., Dasmahapatra, S. y Wilks, Y. Data driven ontology evaluation. Un Proceeding Resources of International Conference on Language Resources and Evaluation. Lisbon, Portugal. 2004. Disponible en fecha noviembre de 2011 en: http://eprints.ecs.soton.ac.uk/9062/

[18] Obrst, L., Ashpole, B., Ceuters, W., Mani, I., Ray, S. y Smith, B. The evaluation on ontologies: Toward improved semantic interoperability. En Baker, C. y Cheung, KH. (eds) Semantic Web Revolutionizing Knowledge in the Life Science. Springer US. pp:139-158. 2007.

[19] Porzel, R. y Malaka, R. A Task-based Approach for Ontoly Evaluation. ECAI Workshop on Ontology Learning and Population. España. 2004. Disponible en fecha Diciembre de 2009 en: http://olp.dfki.de/ecaio4/final-porzel.pdf

[20] Burton, A., Story, V., Suguraman, V. y Ahluwalia, P. A Semiotic Metrics for Assessing the Quality of Ontologues. Data & Knowledge Engineering. Elsevier. (55). pp: 84-102. 2005.

[21] Brank, J. Grobelnik, M. y Mladenic, D. A Survey of Ontology Evaluation. In International Conference on Language Resources and Evaluation, Technique.

SIKDD 2005 Multiconference IS, Ljubljana. Slovenia. 2005. Disponible en fecha Diciembre de 2009 en: http://kt.ijs.si/dunja/sikdd2005/Papers/BrankEvaluationSiKDD2005.pdf

[22] Ramos, E., Núñez, H. y Casañas, R. Esquema para evaluar ontologías únicas para un dominio de conocimiento. *Enl@ce: Revista Venezolana de Información, Tecnología y Conocimiento*, 6 (1), pp: 57-71. ISSN: 1690-7515. 2009. Disponible en fecha enero de 2012 en: http://dialnet.unirioja.es/servlet/articulo?codigo=2932226

[23] Maedche, E. y Staab, S. Measuring Similarity between Ontologies. En Gómez-Pérez, A., y Benjamins, VR. (eds) 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW'02). Singüenza, Spain. Lecture Notes in Artificial Intelligence LNAI 2473. Springer-Verlag, Berlin, Germany. pp: 251-263. 2002.

[24] Navarro, J. y Samos, J. Una panorámica actual de software para trabajar con ontologías. Departamento de Lenguajes y sistemas Informáticos, Escuela Técnica de Ingenierías Informáticas y de Telecomunicaciones, Universidad de Granada. Granada, España. 2007. Disponible en fecha Octubre de 2010 en: http://www.salmer.info/docu/sds07.pdf

[25] Polo, L., Berrueta, D., Rubiera, E. y Fernández, S. D 2.4 Experimento semántico para definir contextos y recursos: 1.0. Morfeo Proyect. 2007. Disponible en fecha Abril de 2010 en: http://forge.morfeo-project.org/wiki/index.php/D_2.4_Experimento_sem%C3%A1ntico_para_definir_contextos_y_recursos:1.0

[26] Miller, E. An introduction to the Resource Description Framework [documento www]. *D-Lib Magazine*, May 1998. Disponible en fecha Diciembre 2009 en: http://www.dlib.org/dlib/may98/miller/05miller.html

[27] Siricharoen, W. Ontology Modeling and Object Modeling in Software Engineering. International Journal of Software Engineering and Its Applications Vol. 3, No. 1. 2009. Disponible en fecha Febrero 2010 en: http://www.sersc.org/journals/IJSEIA/vol3_no1_2009/5.pdf