

UNIVERSIDAD DE ALCALÁ
Departamento de Automática
Grado en Ingeniería Informática
Grado en Ingeniería de Computadores
Grado en Sistemas de Información

Práctica 3: Desarrollo y manejo de ontologías OWL

Índice

1. Competencias asociadas a la práctica	3
2. Introducción	3
3. Ontologías	4
3.1. OWL	4
4. Protégé	5
5. Construcción de ontologías OWL con Protégé	5
5.1. Clases	6
5.2. Propiedades	6
5.2.1. Propiedades inversas	6
5.2.2. Características de las propiedades OWL	7
5.2.3. Restricciones de propiedades	7
6. Herramientas gráficas de Protégé	8
6.1. OWLViz	8
6.2. OntoGraf	8
7. Ejercicios	8
8. Duración de la práctica	12

1. Competencias asociadas a la práctica

1. Comprender la utilidad de la Ingeniería Ontológica en el desarrollo de aplicaciones web.
2. Ser capaz de instalar diversas herramientas necesarias para desarrollar ontologías.
3. Familiarizarse con el editor de ontologías *Protégé* y sus principales funcionalidades para el desarrollo de ontologías en lenguaje OWL.
4. Ser capaz de aplicar el framework Jena para la gestión de ontologías.
5. Ser capaz de comprender la utilidad de realizar inferencias sobre ontologías.
6. Ser capaz de realizar consultas sobre una ontología usando el lenguaje *SPARQL*.

2. Introducción

La integración de la semántica en la Web ha posibilitado que se puedan agregar notaciones semánticas que hacen posible la interacción automática de los sistemas y los consiguientes beneficios para los usuarios. Tim Berners-Lee ¹ pensó en una red de recursos que nos permitiera programar agentes que navegasen a través de la infinidad de sitios web para obtener la información necesaria sin tener que indicarle de dónde obtenerla o qué significado debe tener cada recurso, y que finalmente se transformara esa información a un formato fácilmente entendible por los usuarios. Esta web, aún en fase de desarrollo, se conoce como Web Semántica².

La descripción del significado que tienen los contenidos en la web es uno de los conceptos fundamentales en los que se basa la Web Semántica. Esta descripción requiere conceptos ligados a:

Semántica significado procesable por máquinas.

Metadatos contenedores de información semántica sobre los datos.

Ontologías conjunto de términos y relaciones entre ellos que describen un dominio de aplicación específico.

Las ontologías tienen un papel esencial en las aplicaciones de Web Semántica. Por tanto, uno de los objetivos claves de esta práctica es introducir al alumno en el mundo de las Ontologías, es decir, que sea capaz de construir una ontología con un editor de ontologías ampliamente utilizado como es Protégé y realizar las operaciones más habituales sobre ella, utilizando para ello un framework de amplia difusión como es Jena, que se verá en sesiones de laboratorio.

Asimismo, un sub-objetivo importante dentro de la práctica es acostumbrar al alumno a trabajar en equipo, una cualidad esencial de cara al mundo laboral.

¹<http://www.w3.org/People/Berners-Lee/1996/ppf.html>

²<http://www.med.nyu.edu/research/pdf/mainim01-1484312.pdf>

3. Ontologías

Una ontología se usa para capturar conocimiento acerca de un cierto dominio de interés. Describe los conceptos del dominio y la interrelación entre ellos para proporcionar una comprensión compartida y consensuada del conocimiento de un dominio que puede ser comunicada entre sistemas heterogéneos y entre personas. Fueron desarrolladas en el área de la Inteligencia Artificial para facilitar el intercambio y reutilización del conocimiento (Guarino, 1998)³.

3.1. OWL

Existen diferentes lenguajes de ontologías con diferentes características y posibilidades. El más reciente estándar proporcionado por la W3C es el lenguaje OWL (Ontology Web Language), (Guarino, 1998).

OWL es un lenguaje usado cuando la información contenida en los documentos necesita ser procesada por las aplicaciones. Provee un mayor conjunto de primitivas para representar el significado de los elementos y sus relaciones con otros elementos en una ontología.

OWL se puede formular en RDF y, por lo tanto, es considerado como una extensión de este último. Incorpora toda la capacidad expresiva de RDF y la extiende para utilizar expresiones lógicas. Permite la creación de clases y asociar ciertas propiedades a las relaciones, como cardinalidad, simetría, transitividad, o relaciones inversas (Knoblauch, 2005).

OWL tiene los siguientes niveles:

- OWL Lite. alguna de sus características principales es que tiene restricciones simples (cardinalidad sólo 0 ó 1), facilita el desarrollo de herramientas, es el menos expresivo y se usa en situaciones donde sólo existe una jerarquía simple de clases, con pocas restricciones.
- OWL DL (Description Logics). Tiene una expresividad que, sin ser muy alta, es mayor que la de OWL Lite, basado en lógica descriptiva (lógica basada en lógica de primer orden). OWL Lite está contenido en OWL DL y éste está contenido en OWL Full.
- OWL Full. Es el nivel más expresivo. Se usa en situaciones donde es fundamental una extrema expresividad, más que la capacidad de razonamiento. Realiza la unión de sintaxis OWL y RDF (sin restricciones).

OWL se expresa con primitivas RDF (todo documento válido OWL es un documento válido RDF). El lenguaje RDF está contenido en OWL Full -extensión de RDF- (todo documento válido RDF es un documento válido OWL Full). RDF no está contenido ni en OWL DL ni en OWL Lite (son en realidad una extensión de una versión restringida de RDF). Por tanto, no todo documento válido RDF es un documento válido en OWL DL o en OWL Lite.

³<http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.29.1776>

4. Protégé

Protégé es una herramienta open source desarrollada en *Stanford Medical Informatics*. Es una herramienta de modelado que, como otras, se divide en una parte de modelo y una parte de vista:

- El modelo de Protégé es el mecanismo de representación interna de ontologías y bases de conocimiento.
- Los componentes de la vista de Protégé proporcionan al usuario una interfaz para visualizar y manipular el modelo (*Getting Started With Protégé*, 2003)⁴.

El modelo de Protégé está basado en un metamodelo sencillo similar a sistemas orientados a objetos y sistemas basados en marcos. Estos pueden representar ontologías consistentes en clases, propiedades, características de propiedades e instancias. La ventaja de Protégé es que su metamodelo es, en sí, una ontología Protégé. La estructura del metamodelo facilita la extensión y adaptación a otras representaciones.

Por otro lado, las vistas de la interfaz de usuario de Protégé permite a los diseñadores de ontologías, entre otras tareas, crear clases y asignarles propiedades y creación de instancias. En cada clase de la ontología, el sistema crea un formulario con componentes de edición para cada propiedad de la clase. En la actualidad, Protégé se puede usar para cargar, editar y salvar ontologías en varios formatos como CLISP, RDF, XML, UML, y OWL. Además, Protégé proporciona *plugins*. Con el plugin OWL de Protégé es una extensión compleja que puede ser utilizada para editar archivos OWL y bases de datos y, entre otras características, proporciona las siguientes:

- Acceso a servicios basado en clasificación, chequeo consistente y pruebas.
- Permite la descripción de conceptos y tiene un amplio conjunto de operadores (and, or, y negación).
- Permite que mucha gente edite la misma ontología al mismo tiempo.
- Está basado en un modelo lógico que garantiza que los conceptos (incluso complejos) estén bien descritos y que permite utilizar un razonador que chequea la consistencia de la ontología garantizando que la jerarquía descrita en la ontología sea correcta.

5. Construcción de ontologías OWL con Protégé

Para crear una ontología OWL con el editor Protégé, se requiere poseer el plugin OWL⁵. Al arrancar Protégé, se debe crear una ontología OWL, tal y como se verá en la realización, paso a paso de la guía práctica de Protégé en el primer paso de esta práctica. En este apartado sólo se van a describir los elementos de una ontología OWL para que el alumno vaya familiarizándose con la terminología asociada.

Una ontología OWL está formada por los siguientes elementos:

⁴<http://protegewiki.stanford.edu/wiki/Protege4GettingStarted>

⁵En las últimas versiones de Protégé ya está incluido.

- **Clases OWL.** Conjuntos que contienen individuos y se pueden organizar en una jerarquía de clases y subclases conocida como taxonomía. Se conocen también como conceptos al ser una representación de ellos.
- **Individuos.** Representan objetos del dominio de interés. Se conocen también como instancias.
- **Propiedades.** Relaciones binarias sobre los individuos. Pueden ser de diversos tipos: inversas, transitivas, o simétricas.

5.1. Clases

Al iniciar la creación de una ontología OWL, existe una clase llamada "owl:Thing" que contendrá la ontología y que representa el conjunto de todos los individuos. Todas las clases que se creen serán subclases de *Thing*.

Después de crear todas las clases en la jerarquía de la ontología, se deben establecer las clases disjuntas, es decir, las clases que indican que un objeto o individuo no puede ser instancia de más de una de estas clases que se establecieron disjuntas.

5.2. Propiedades

Las propiedades OWL representan relaciones entre dos objetos o individuos. Existen los siguientes tipos de propiedades:

- **ObjectProperties** (Propiedades Objeto). Permite relacionar un individuo con otro.
- **DatatypeProperties** (Propiedades de tipo Datos). Representan datos que no son objetos. Relacionan un individuo con un valor XML Schema Datatype o un literal RDF.

Los elementos que debe tener una propiedad Objeto son los siguientes:

- **Nombre** de la propiedad.
- **Dominio.** Se refiere a la clase o clases iniciales de la propiedad.
- **Rango.** Se refiere a la clase o clases finales que relaciona la propiedad.

Toda propiedad de tipo *ObjectProperty* (propiedad Objeto) debe tener un dominio y un rango especificados. Las propiedades enlazan objetos del dominio con objetos del rango; dominio y rango son clases definidas en la ontología.

5.2.1. Propiedades inversas

Cada propiedad Objeto debe tener su correspondiente propiedad inversa. Si una propiedad relaciona o enlaza un objeto A con otro B, la propiedad inversa enlaza el objeto B con el A.

5.2.2. Características de las propiedades OWL

OWL permite que el significado de las propiedades se vea enriquecido con las siguientes características:

- *Propiedades funcionales.* Si una propiedad es funcional, puede estar a lo sumo un objeto relacionado con otro mediante esta propiedad. Por ejemplo, si se tienen tres objetos que son X, Y y Z y se tiene una propiedad funcional *tienePadre*, entonces se podrán asociar los objetos X e Y por medio de la propiedad y resultaría que X *tienePadre* Y. Del mismo modo, se podrían asociar los objetos X y Z por medio de la propiedad y resultaría que X *tienePadre* Z. Como la propiedad *tienePadre* es funcional, se concluye que Y y Z son el mismo objeto.
- *Propiedades funcionales inversas.* Si una propiedad es funcional inversa significa que puede estar a lo sumo un objeto relacionado con otro mediante esta propiedad de forma inversa a la propiedad funcional original. Por ejemplo, si se tienen tres objetos que son X, Y, y Z, y se tiene una propiedad funcional *esPadreDe* (inversa de *tienePadre*), entonces se podría asociar el objeto Y y X por medio de la propiedad y daría como resultado Y *esPadreDe* X. Del mismo modo, se podrían asociar los objetos Z y A por medio de la propiedad y resultaría que Z *esPadreDe* X. *esPadreDe* es propiedad funcional inversa, luego se concluye que Y y Z son el mismo objeto.
- *Propiedades transitivas.* Si una propiedad es transitiva y relaciona dos objetos X e Y, y además la propiedad relaciona el objeto Y con el Z, entonces se puede inferir que el objeto X está relacionado con el objeto Z mediante la propiedad transitiva.
- *Propiedades simétricas.* Si una propiedad R es simétrica y la propiedad relaciona los objetos X e Y, entonces el objeto Y es relacionado mediante la propiedad R con el objeto X.

5.2.3. Restricciones de propiedades

Las propiedades ya creadas pueden utilizarse para definir y describir el comportamiento de las clases (*Getting Started with Protégé*, 2003) creando restricciones en las clases de una ontología OWL.

En general, el nombre de la propiedad debería sugerir las restricciones impuestas a los objetos de la clase.

Las restricciones OWL se clasifican en los siguientes tres tipos:

- Restricciones de cuantificación. Se componen de los siguientes elementos:
 - Cuantificador existencial (\exists). Permite indicar la existencia de, al menos, un objeto.
 - Cuantificador universal (\forall). Permite indicar la existencia de todos los objetos.
- Restricciones de cardinalidad.
- Restricciones de valor.

En Protégé, debe seleccionarse previamente la clase para la que se establezcan las restricciones. En el desarrollo de la guía práctica de Protégé (ejercicio 1 de la práctica), se aprenderá a definir restricciones de propiedades de cada tipo.

6. Herramientas gráficas de Protégé

6.1. OWLViz

OWLViz es una herramienta de Protégé que se instala como un plugin. Para hacer uso de esta herramienta se requiere instalar previamente *GraphViz*, una herramienta que permite realizar componentes gráficos de la ontología. Una vez instaladas *GraphViz* y *OWLViz*, se genera una pestaña en el entorno de Protégé denominada como la herramienta, *OWLViz*. Al seleccionar la pestaña, aparece una barra de herramientas que proporciona diferentes funcionalidades al plugin. En esta barra se encuentra un icono denominado *Options* en el que se indica la ruta de la aplicación *GraphViz* y ciertas características con las que se quiere visualizar la ontología en formato gráfico con la jerarquía de aserción o la jerarquía de inferencia. Para razonar con una ontología se requiere una herramienta que actúe como razonador para Protégé. Existe una herramienta llamada *Racer* para el análisis y razonamiento de las ontologías que se puede conectar con Protégé.

6.2. OntoGraf

Con la herramienta *Ontograf* Protégé ofrece soporte para navegar interactivamente entre las relaciones de sus ontologías OWL. Soporta diversos diseños para organizar automática la estructura de la ontología. Se soportan diferentes relaciones: subclase, individuos, dominio/rango de propiedades objeto, y equivalencia. Las relaciones y los tipos de nodos se pueden filtrar.

7. Ejercicios

Como aplicación de la Web Semántica y, en concreto, la Ingeniería Ontológica, el alumno deberá realizar varios ejercicios que le permitan familiarizarse con las Ontologías y aprender a manejarlas tanto desde el punto de vista de su diseño como desde el punto de vista de su programación, percibiendo así de forma más directa las ventajas de la Ingeniería Ontológica en el desarrollo de una aplicación web como son los sistemas de supervisión y entrenamiento a través de la web.

La práctica se ha dividido en varios bloques, de carácter obligatorio u optativo, que se describen a continuación:

- BLOQUE OBLIGATORIO I: *Creación de ontologías guiada con un editor (Protégé).*

1. Instalar el editor de Ontologías *Protégé*.

2. Instalar el *plugin* para Protégé OWLViz (<http://www.co-ode.org/downloads/owlviz/OWLVizGuide.pdf>). Necesitará instalar previamente la herramienta gráfica *Graphviz* (`sudo apt-get install graphviz`). En las últimas versiones, se instalan ambas automáticamente con Protégé.
3. Como primer contacto con el editor Protégé, realice los pasos de la guía de Protégé 4: *A practical Guide To Building OWL Ontologies* (sólo **hasta** el apartado 4.9) mediante los ejercicios (muy sencillos) que se van proponiendo en el documento. Introduzca los distintos elementos que se proponen en los ejercicios y de la forma que especifica en la guía pero traduzca todos los elementos a añadir en la jerarquía de la ontología EN ESPAÑOL⁶. Una vez construida la ontología, deberá entregarse el archivo *.owl* asociado a la ontología.

■ BLOQUE OPTATIVO I: *Creación de ontologías e inferencia*.

Realizada la ontología previa, de forma completamente guiada, construya ahora sin guía, la ontología *parentesco.owl*. La taxonomía y relaciones asociadas se proporcionarán de forma textual al alumno.⁷

■ BLOQUE OBLIGATORIO II: *Manejo básico de ontologías usando las APIs de Jena*

1. Instale el *framework Jena*. Debería estar instalado desde la realización de la práctica 2.
2. Abra en Protégé la ontología *movieontology.owl* que se ha proporcionado como material de apoyo y analícela. Utilizando esta ontología como ejemplo y tomando como archivo de partida el modelo *Modelo.java* que se proporcionó en la práctica 2, realice una clase *ManejoOntologia* en la que incluirá algunos métodos relacionados con el manejo de ontologías que pueden ser útiles para ser aplicados en diferentes contextos que involucren operaciones con ontologías. Los métodos que se añadirán a la clase son los siguientes:
 - Método *CargarOntología()* que cargue la ontología *movieontology.owl*. Sugerencia: utilice el modelo que posee de la práctica 2 para esta parte.⁸. Por simplicidad, este método no se aplicará de forma genérica para cualquier ontología, sino que será específico para la ontología dada. Debe añadir **NECESARIAMENTE**, con comentarios en el propio código, una explicación clara de las diferentes APIs de Jena que utilice. Esto le ayudará a ir comprendiendo poco a poco la semántica de las más frecuentemente

⁶Aunque la guía parece muy extensa, hay muchas páginas con capturas de pantalla para mejorar la comprensión de las explicaciones y los ejercicios son muy sencillos y guiados a través del documento.

⁷En realidad, para desarrollar una ontología de forma rigurosa para aplicaciones web es útil aplicar una buena metodología de desarrollo de ontologías como NeOn <http://mayor2.dia.fi.upm.es/oeg-upm/index.php/es/methodologies/59-neon-methodology>. Las metodologías de construcción de ontologías quedan fuera del ámbito de la asignatura pero es interesante hacer referencia a ellas, a pesar de que, para desarrollar ontologías, sobre todo *light-weight*, no se siga con frecuencia una metodología concreta.

⁸Tenga en cuenta al analizar esta ontología en Protégé si importa otra(s) ontologías y sus URIs.

utilizadas⁹. Además, tenga cuidado con los paquetes que se vayan necesitando, de acuerdo siempre a la interfaz utilizada.

- Añada a la clase ManejoOntología un método, EscribirOntología(), que se encargue de almacenar la ontología en un archivo. Utilice de nuevo como modelo el archivo Modelo.java en el que se añadía esta funcionalidad dentro de su código. Incluya como interfaz del método (parámetros de entrada y salida) los que considere adecuados (es sencillo).
- Introduzca el método que a continuación se proporciona:

```
1      public void imprimirPropiedadesObjeto(OntModel m) {
2          ArrayList<String> propiedades = new ArrayList<String>();
3          Iterator i = m.listObjectProperties().filterDrop(new Filter() {
4              public boolean accept(Object o) {
5                  return ((Resource) o).isAnon();
6              }
7          });
8          while (i.hasNext()) {
9              ObjectProperty pro = ((ObjectProperty) i.next());
10             propiedades.add(pro.getLocalName());
11         }
12         for (int j = 0; j < propiedades.size(); j++) {
13             System.out.println(propiedades.get(j));
14         }
15     }
16 }
```

El método anterior imprime las propiedades de tipo objeto de una ontología. Realice un método similar, imprimirPropiedadesDatos(), encargado de imprimir las propiedades de datos (*data property*) de la ontología.

■ BLOQUE OPTATIVO II: *Manejo más avanzado de ontologías usando las APIs de Jena.*

1. Añada a la clase ManejoOntología un método que se encargue de invocar a imprimirPropiedadesObjeto() y a imprimirPropiedadesDatos() para imprimir todas las propiedades definidas en una ontología.
2. Se valorará que el alumno incluya otros métodos típicos para: obtener las instancias de una ontología (o mostrarlas en pantalla), añadir una propiedad objeto a la ontología, añadir una propiedad de tipo dato (*data property*) a la ontología, obtener las instancias de una clase (o mostrarlas en pantalla), etc.¹⁰

■ BLOQUE OBLIGATORIO III: *Consulta sobre ontologías utilizando el estándar SPARQL.*

1. Añada el valor de la propiedad tieneSexo y fechaNacimiento a todas las instancias que se añadieron a la ontología parentesco.owl en sesiones previas.
2. Utilizando el *framework Jena* y el lenguaje de consultas de *SPARQL*, realice un programa con la siguiente funcionalidad:

⁹Para ello, consulte el javadoc de Jena: <http://jena.apache.org/documentation/javadoc/jena/>

¹⁰Si el alumno decide finalmente realizar este apartado optativo, se le proporcionará una guía para su realización.

- Cargue la ontología `parentesco.owl` (ya visto en sesiones previas).
- Incluya el siguiente código correspondiente a una consulta sencilla y ejecutelo. De esta forma, se familiarizará con las APIs y con la sintaxis de SPARQL para realizar las posteriores consultas.

```

/* incluya funcionalidad de las APIs mediante comentarios */
/*query String: personas cuya madre es Maria */
String queryString = "PREFIX familia:<http://www.semanticweb.org/.../parentesco.owl#>" +
    "SELECT ?descendientes" +
    "WHERE {?descendientes familia:tieneProgenitor familia:Maria}";

Query query = QueryFactory.create(queryString);
QueryExecution qe = QueryExecutionFactory.create(query, moduloCarga);
ResultSet results = qe.execSelect();
ResultSetFormatter.out(System.out, results, query);
qe.close();

```

- Realice las consultas que se enumeran a continuación utilizando los elementos de la sintaxis de *SPARQL* que considere adecuados (ya vistos en clase), adaptando el fragmento de código previo a cada consulta¹¹.
 - Obtenga los abuelos y nietos pero SIN USAR la propiedad `tieneAbuelo` definida en `parentesco.owl`.
 - Obtenga toda la información referente a *Ana* almacenada en la ontología¹².
 - Obtenga los hijos de Alvaro y cuya fecha de nacimiento sea mayor que 1999-12-23 y menor que 2003-11-24.
 - Obtenga los hijos de Alvaro y los hijos de Sara.
- BLOQUE OPTATIVO III: *Consulta sobre ontologías más avanzadas utilizando el estándar SPARQL*.
 - Obtenga, siguiendo la idea de la consulta 2 del bloque anterior, sobre qué datos se tiene información de la instancia *Ana*, es decir, de qué propiedades se tiene información referente a ella. Tenga en cuenta que el resultado de la consulta 2 incluye información sobre alguna propiedad varias veces puesto que para ellas hay varios datos almacenados. **Sólo** se desea con esta consulta que aparezcan las propiedades y **no repetidas** de las que se sabe información para Ana (**sin los valores** correspondientes).
 - Obtenga los nombres y apellidos de todas las personas que tienen hermanos y sus fechas de nacimiento. La información debe estar ordenada por nombre y apellidos.
 - Obtenga los nombres y apellidos de todas las personas que tienen hermanos y sus fechas de nacimiento pero sólo debe aparecer la información de las tres personas más jóvenes.

Todos los pasos se llevarán a cabo de forma guiada. Sobre ellos se irá proporcionando al alumno la información necesaria para su adecuada realización.

¹¹ Tenga cuidado con los paquetes que se requieren para realizar consultas de tal forma que la compilación sea correcta en el entorno de programación que utiliza (Eclipse, etc.).

¹² La información sobre Ana son sus propiedades junto con los valores añadidos a cada una de ellas para la instancia asociada: *Ana*.

8. Duración de la práctica

La práctica se desarrollará en aproximadamente 10 horas. De 2 a 4 horas de trabajo en casa, y de 6 a 8 horas en el grupo pequeño correspondiente. Debe realizarse necesariamente en grupos de dos alumnos.

La entrega se realizará a través del enlace correspondiente que se habilitará en la página web de la asignatura.