
UNIVERSIDAD TECNOLÓGICA DE LA MIXTECA

# "Construcción coordinada de ontologías para representar clasificaciones jerárquicas de documentos"

T E S I S

que para obtener el título de

Ingeniero en Computación

presenta

Ricardo Omar Chávez García

Directora de tesis:

M. en C. María Auxilio Medina Nieto

Huajuapan de León, Oaxaca, a 8 de enero de 2008

Aunque a veces me sentí solo, siempre supe que estaban ... y estarán ahí.
Gracias:
Mamá, Hermanitos, Gaby, y cómo no, mis amigos.
Terminé.

# Índice general

# Capítulo 1

# Introducción

---

> Si el conocimiento puede crear problemas, no será a través de la ignorancia que podamos resolverlos.
>
> *Isaac Asimov*

En la literatura de bibliotecas digitales, son comunes las referencias hacia la Iniciativa de Archivos Abiertos (*Open Archives Initiative OAI*), la cual está formada por proveedores de datos, (organismos que comparten colecciones de información), y proveedores de servicios, (sistemas que emplean los datos para ofrecer servicios como recuperación de documentos).

Usuarios y aplicaciones consideran a los proveedores de datos fuentes valiosas de información, los cuales hacen uso de registros de metadatos, datos acerca de los datos, para describir documentos [VdS02]. Algunos proveedores cuentan con mecanismos de recuperación de registros. Sin embargo, cuando se desea consultar varios proveedores de datos, los usuarios necesitan acceder, seleccionar y organizar los documentos por proveedor para determinar su relevancia.

El proveedor de servicios OntoSIR, propuesto en [Med05] es una herramienta que intenta apoyar esta tarea a través de la generación y uso de ontologías de registros, las cuales son estructuras jerárquicas que representan la organización de un conjunto de documentos.

Las ontologías de registros se generan de forma automática y se representan en lenguaje XML, lo cual facilita su manejo. Sin embargo, los editores comunes de ontologías no reconocen este lenguaje, lo cual dificulta la revisión y extensibilidad de las ontologías propuestas. Por ello, la justificación de esta tesis es la implementación de un sistema que permita a los usuarios transformar una ontología de registros en una ontología en OWL (*Web Ontology Language*) a través de un proceso coordinado de revisión.

## 1.1. Hipótesis

Es posible construir una ontología de registros en OWL a partir de una representación en XML y un proceso de revisión.

## 1.2. Objetivo general

Implementar un sistema de revisión que permita la construcción coordinada de una ontología de registros.

## 1.3. Objetivos específicos

- Describir el escenario de aplicación de la ontología de registros.
- Identificar las limitaciones principales del lenguaje XML para representar ontologías.
- Comparar la expresividad de los lenguajes RDF, RDF-Schema, DAML-OIL y OWL.
- Implementar la fusión de grupos de documentos.
- Describir los elementos de OWL necesarios para construir la ontología.
- Proponer un mecanismo de revisión que facilite la construcción de la ontología.
- Verificar que las ontologías generadas sean reconocidas por editores de OWL.

## 1.4. Propuesta de solución

Se propone el desarrollo de un sistema accesible vía web que implementará las tareas siguientes:

- *Reconstrucción*. Esta tarea permitirá a los usuarios reconstruir la ontología a través de la unión de los documentos de un nodo hijo con los documentos de su nodo padre.
- *Revisión*. Esta tarea consistirá en implementar un mecanismo coordinado de revisión mediante la asignación de un grupo de revisores con diferentes niveles de prioridad.
- *Construcción*. Esta tarea transformará una ontología de registros en XML, a una ontología equivalente en OWL.

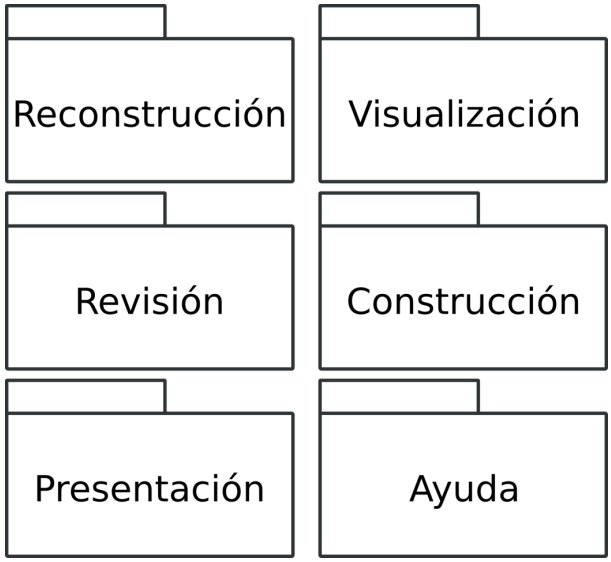


**Figura 1.1.** Arquitectura del sistema.

- *Visualización.* Esta tarea permitirá a los usuarios visualizar la ontología de registros para revisar la organización de los documentos.

Los módulos principales del sistema se muestran en la Figura 1. Éstos corresponden a las tareas descritas, a excepción de los módulos de presentación (encargado de la interfaz gráfica para el usuario), y el módulo de ayuda.

## 1.5. Trabajo relacionado

La literatura describe varias formas de representar clasificaciones jerárquicas de documentos; algunas de las más comunes son gráficas o basadas en texto, una antología de registros, en particular, es una clasificación de este tipo. Estas representaciones se aplican exitosamente a jerarquías pequeñas. Sin embargo, cuando el número de documentos es grande, una representación no ambigua y legible para la computadora ofrecería mayores ventajas.

La construcción de una representación con estas características constituye la verificación de la hipótesis. Dos fuentes relacionadas con la hipótesis son las siguientes. [Far05] presentan una comparación entre los lenguajes XML, RDF y OWL, la cual analiza los aspectos de apoyo a la seguridad que éstos ofrecen. Este documento sirve para identificar los elementos a comparar para hacer la traducción en esta tesis. Por otro lado, la herramienta *Contorsion XPath*[1] realiza una traducción de XML a OWL con base en un enfoque basado en modelo. Esta herramienta se utiliza para analizar la manera en que realiza la traducción.

[1] http://dmag.upf.edu/contorsion/

## 1.6. Metodología

La implementación de este sistema seguirá la metodología de desarrollo basada en componentes. Este enfoque se basa en la existencia de un número significante de componentes reutilizables. El proceso de desarrollo del sistema se enfoca en integrar estos componentes.

La característica principal de esta metodología es la reutilización, la cual provee beneficios al reducir el tiempo y costo de un desarrollo de software.

## 1.7. Alcances y limitaciones

- La revisión será un proceso secuencial. Se realizará antes de la traducción.
- La traducción se realizará sólo si los documentos fuentes satisfacen el DTD[2] definido para una ontología de registros.
- La factibilidad para editar la ontología incluirá el análisis comparativo de dos editores de OWL.

El orden de la tesis es el siguiente: el Capítulo 2 muestra el marco teórico, seguido del diseño del sistema en el Capítulo 3. El Capítulo 4 muestra la representación de las ontologías de registros en los lenguajes XML, RDF y OWL. La implementación del sistema de revisión y traducción se muestran en el Capítulo 5. En el Capítulo 6 se presentan las conclusiones del trabajo de tesis.

[2]Siglas de Document Type Definition

# Capítulo 2

# Marco teórico

---

> La investigación se asemeja a los largos meses de gestación, y la solución del problema, al día de nacimiento. Investigar un problema es resolverlo.
>
> *Mao Tse Tung*

## 2.1. Ontología de registros

Una ontología es la especificación explícita de una conceptualización compartida [Gru93]. Informalmente, las ontologías definen los conceptos de un vocabulario que representan un dominio de interés para una comunidad. Por ejemplo, existen ontologías para médicos, químicos o ingenieros. En esta tesis se denominan *ontologías de registros* a las estructuras jerárquicas que representan colecciones de documentos de los proveedores de datos en la OAI (*Open Archives Initiative*). OntoSIR es un software que procesa estas estructuras y provee un servicio asíncrono para la recuperación de documentos de múltiples colecciones. El componente de OntoSIR que se encarga de la construcción del documento que representa la estructura jerárquica de los documentos recuperados de los distintos proveedores de datos se llama OntoSER, este componente está basado en el formato de metadatos Dublin Core y en el protocolo de interoperabilidad OAI-PMH (*Open Archives Initiative Protocol for Metadata Harvesting*). Las tareas que realiza son las siguientes:

- Recolección.
- Representación.
- Agrupamiento.
- Transformación.

La descripción detalla de cada tarea se presenta en [Med05].

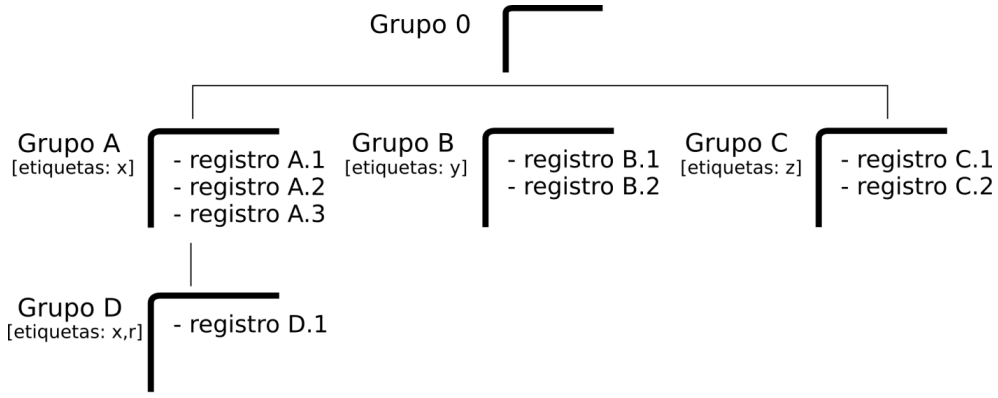


**Figura 2.1.** Representación gráfica de una ontología de registros.

Las ontologías de registros están formadas por grupos, los cuales representan niveles en la estructura jerárquica. Estos grupos están caracterizados por etiquetas que representan a los documentos o registros que están contenidos en ellos. Los grupos a su vez contienen registros con metadatos sobre documentos reales, tales como: título, tema, descripción, identificador, url, proveedor de datos, formato de los metadatos y una estampa de tiempo [Med06]. La Figura 2.1 muestra de forma gráfica lo dicho en este párrafo.

La estructura jerárquica que se obtiene con las ontologías de registros permite tener una clasificación de los documentos comunes dentro de un dominio, es por ello que los investigadores de ese dominio encuentran útil la representación, la cual se puede utilizar como una ontología base a la que se le pueden aplicar operaciones para obtener una ontología que concuerde con sus necesidades, o también la pueden utilizar como una opción de método de clasificación o un punto de referencia para otras clasificaciones.

Dentro del dominio de la Iniciativa de Archivos Abiertos podemos encontrar otro escenario de aplicación, en el cual OntoSIR basa su clasificación para obtener una ontología de registros, con lo cual se facilita el compartir la información a través de internet, y precisamente con sistemas que implementen OAI-PMH.

La aplicación más clara surge con la necesidad de la visualización, resultado de recolectar y clasificar registros de varios proveedores de datos. Una ontología de registros muestra la clasificación de todos los registros, agrupándolos de acuerdo al número de etiquetas que tienen en común, por tal motivo, personas como un bibliotecario resultarían beneficiadas.

## 2.2. La operación de fusión

Las ontologías de registros generadas por OntoSIR, son el resultado de agrupar jerárquicamente documentos comunes según sus metadatos, en algunas ocasiones la

ontología generada no contiene la clasificación esperada por el especialista humano, por ello es necesario realizar una modificación a la ontología de registros. La forma de modificar la ontología es mediante la operación de fusión de grupos.

En la estructura jerárquica cada grupo almacena a los registros comunes, a las etiquetas que lo describen, y cada grupo de menor jerarquía está descrito por las etiquetas del grupo inmediato de mayor jerarquía más una etiqueta extra, de esta forma, los registros en los grupos con la última jerarquía están identificados de manera más específica.

La operación de fusión de grupos se realiza para eliminar niveles jerárquicos que se creen innecesarios en la estructura, para que de esta forma, se mejore la clasificación. Básicamente se puede decir que la fusión, como su nombre lo dice, une dos grupos de la estructura jerárquica sin romper las reglas de la misma. Para realizar la operación de fusión se toman en cuenta las siguientes reglas:

1. Sólo se pueden fusionar 2 grupos a la vez.
2. Sólo se puede fusionar grupos de nivel mayor o igual a 1.
3. Sólo se puede realizar la fusión entre el grupo de nivel n y su hijo de nivel n+1.
4. Al fusionar el grupo A y el grupo B, los registros del grupo B así como sus grupos descendientes, pasan a formar parte de la estructura del grupo A, es decir, los registros de B serán ahora los registros de A, y los grupos descendientes de B serán ahora los grupos descendientes de A. Además, la etiqueta extra que representaba al grupo B desaparece, por lo tanto el grupo B desaparece, quedando representados los registros de B por las etiquetas de A. Los descendientes de B tampoco serán representados por la etiqueta de B.

La Figura 2.2 muestra gráficamente la operación de fusión entre los grupos A y D, con la que el grupo D desaparece y el grupo E se convierte en el hijo de A.

## 2.3. XML

### 2.3.1. Introducción a XML

XML son las siglas en inglés de *Extensible Markup Language* (Lenguaje de Marcado Extensible), el cual es un lenguaje para la representación de datos en forma de cadenas de texto, además incluye "marcado" intercalado para describir propiedades de los datos.

El marcado ocurre como *etiquetas*, la cuales se distinguen de los datos porque están encerradas entre paréntesis angulares <, >, por ejemplo: <*universidad*> y <*salón*>. Un documento XML está formado por etiquetas y por datos, los cuales al combinarse forman *elementos*. Un elemento está definido por una etiqueta de inicio

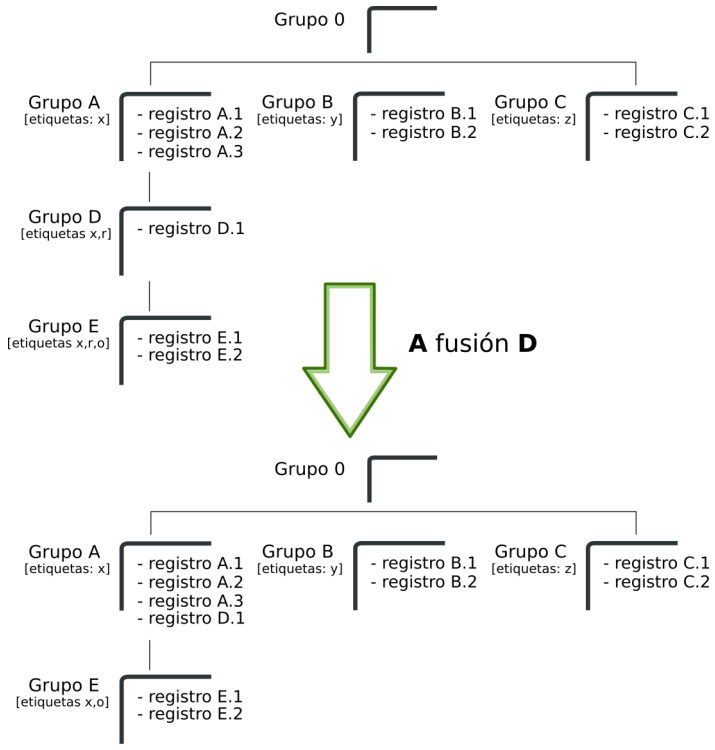


**Figura 2.2.** Operación de fusión entre los grupos A y D.

y una etiqueta final. Una etiqueta final tiene además de los paréntesis angulares una diagonal / después de < para distinguirse de la etiqueta de inicio. En XML, el elemento es el bloque básico de construcción. A continuación se muestran dos ejemplos de elementos en XML:

**Ejemplo 2.1.**

```
<definicion> Una ontología es la especificación explícita de una
conceptualización compartida </definicion>
```

**Ejemplo 2.2.**

```
<tipo> Ontologia de registros </tipo>
```

Del ejemplo anterior, <definicion> y <tipo> son las etiquetas de inicio, </definicion> y </tipo> son las etiquetas finales y el texto entre ellas forman los datos de los elementos. Es importante resaltar que por convención los nombres de las etiquetas se escriben en minúsculas.

De forma específica, el marcado provee un mecanismo para agregar *meta-contenido* e información de la estructura del documento. Los elementos pueden contener dentro de ellos otros elementos anidados los cuales se conocen como *sub-elementos*. En general podemos decir que un documento XML consiste de un elemento de nivel superior (*top-level element*), el cual contiene otros elementos y/o datos y cada sub-elemento contiene otros sub-elementos que a su vez pueden contener más datos y más elementos, y así sucesivamente [Gra02]. Por ejemplo:

**Ejemplo 2.3.**

```
<alumno>
  <nombre>
     <primernombre> Mario </primernombre>
     <apellidopaterno> Torres </apellidopaterno>
     <apellidomaterno> Mora </apellidomaterno>
  </nombre>
  <edad> 20 </edad>
  <carrera> Ing. Industrial </carrera>
</alumno>
```

Donde *alumno* es el elemento de nivel superior que contiene tres sub-elementos: *nombre, edad y carrera*, de los cuales sólo el sub-elemento nombre tiene sub-elementos: *primernombre, apellidopaterno y apellidomaterno*.

Otra característica de los elementos es que pueden tener atributos, que se representan como parte de la etiqueta de inicio, tal como < alumno sexo=“masculino”> o <definicion lenguaje=“ingles”palabraclave=“ontologias”>. Un atributo consiste de un par nombre-valor.

A diferencia de HTML (*HyperText Markup Language*) que es un lenguaje muy utilizado para el diseño de páginas web, el cual mezcla el contenido, la presentación y el proceso de visualización en el mismo documento, en el lenguaje XML y los estándares con los que se relaciona, fomentan la separación del contenido como tipos de elementos abstractos, la presentación como objetos para el formato de los datos y el proceso de visualización como hojas de estilo tales como XSLT[1] y CSS[2] [Gra02].

En HTML es más difícil trabajar con los documentos y darles mantenimiento. En cambio XML ofrece ventajas como las siguientes:

- Es una recomendación del *World Wide Web Consortium* (W3C).
- Es un mecanismo para el intercambio de la información entre personas y aplicaciones.
- Permite el intercambio de contenido de una forma flexible y extendible.
- Permite integrar datos en documentos existentes.
- Puede representar estructuras de datos complejas.
- Tiene una sintaxis consistente que permite el uso de parsers y su creación.
- Existe una gran cantidad de herramientas para el manejo de XML.
- Los documentos XML pueden ser presentados en una gran variedad de formatos: HTML, PDF, Postscript, entre otros.
- Se utiliza para representar y gestionar bases de datos y aplicaciones Web.
- Es una forma sencilla para definir metadatos.

Algunas de las desventajas de usar XML son:

- No permite la especificación de tipos de datos ni la definición de tipos definidos por el usuario.
- Sólo existen relaciones del tipo "*es un*" y sub-elementos "parte-de".
- No permite definir restricciones en la estructura del documento por sí solo.
- Debido a la simplicidad de sus relaciones, no se puede asegurar que un documento escrito en XML representa claramente la realidad.

Estas ventajas y desventajas permiten ver la capacidad que posee XML para la definición de un vocabulario y las restricciones para la definición y uso de relaciones entre elementos.

[1] *Extensible Stylesheet Language Transformations*
[2] *Cascade Style Sheet*

### 2.3.2. Documentos XML válidos

Para que un documento XML sea leído por un parser o alguna herramienta, éste debe cumplir con la característica de ser *bien formado*, es decir, que debe cumplir con las reglas establecidas por la W3C. Algunas de las reglas más importantes son las siguientes:

- Debe tener un prólogo, el cual va en la primera línea del documento e indica la versión de XML y el conjunto de caracteres que se va a utilizar, por ejemplo: <?xml version="1.0" encoding="ISO-8859-1"?>.
- Sólo debe existir un *elemento raíz*.
- Todo elemento debe tener su etiqueta de inicio y de término o bien pueden definirse etiquetas vacías.
- Los nombres de las etiquetas no deben contener el prefijo *xml* ni ninguna combinación de estas letras en mayúsculas o minúsculas.

Un documento bien formado es *válido* si cumple con algunas restricciones de estructura, es decir, si tiene un documento de declaraciones de tipos asociado y si el documento cumple con las restricciones establecidas en él.

Un documento de declaraciones de tipos contiene las declaraciones de marcado que proveen una gramática para una clase de documentos XML. Esta gramática es conocida como documento de declaraciones de tipos (*Document Type Definition*) o DTD [W3C06a]. Un DTD define los bloques de construcción (elementos) permitidos para un documento XML.

El DTD puede declararse interno en el documento XML o como una referencia a un documento externo [W3S06a].

Una alternativa del DTD es el XML-Schema, el cual también es un lenguaje para describir la estructura de un documento XML. XML Schema está basado en XML y debe cumplir con todas las reglas anteriores. Con el lenguaje XML-Schema se pueden definir:

- Elementos y atributos que pueden aparecer en un documento.
- La jerarquía de los elementos.
- El orden y número de los elementos hijo.
- Cardinalidad de los elementos.
- Tipos de datos para los elementos y atributos.
- Valores por defecto o fijos de los elementos y atributos.

Otras razones por las que se prefiere usar XML-Schema en vez de DTDs son [W3S06b]:

- XML-Schema no requieren de una sintaxis especial como los DTDs.
- XML-Schema son extensibles para futuras modificaciones.
- XML-Schema soportan espacios de nombres[3] [W3C06b].

### 2.3.3. URI y URL

Al hablar de XML, es necesario también hablar sobre URI (*Uniform Resource Identifier*) y URL (*Uniform Resource Locator*). Un URI, identificador uniforme de recursos (servicio, página, documento, dirección de correo electrónico,...), es una cadena de caracteres que identifica un recurso físico o abstracto accesible en una red. Una vez que se tiene identificado un recurso, es posible realizar una variedad de operaciones con él, tales como: accederlo (*access*), actualizarlo(*update*), reemplazarlo (*replace*) o localizar sus atributos (*find attributes*).

Un URI consta de dos partes:

1. Identificador del método de acceso (protocolo) al recurso, por ejemplo *http:, mailto:, ftp:.*
2. Nombre del recurso, por ejemplo *//utm.mx.*

Un ejemplo de URI es el siguiente:

```
ftp://ftp.is.co.za/universidad/universodadinfo.txt
```

Un URI puede ser considerado un localizador, un nombre o ambos. El término localizador uniforme de recursos se refiere al subconjunto de URI que identifica recursos vía una interpretación de sus mecanismos de acceso primario (por ejemplo su localización) más que por un nombre o algún otro atributo del recurso [W3C98]. Un ejemplo de URL es el siguiente:

```
http://universidad.mx/definiciones/URL\#inicio
```

## 2.4. RDF y RDF-Schema

XML es un metalenguaje para el marcado que provee una forma para describir lenguajes, además de un conjunto de herramientas para el intercambio de datos y metadatos entre aplicaciones. Pero existe algo que XML no provee: el significado de los datos accesibles para una computadora, la semántica ( "machine-accesible"). Por ejemplo, no existe algún significado implícito en la anidación de las etiquetas, cada aplicación o persona interpreta de la forma más conveniente esta estructura.

El Marco de Descripción de Recursos (RDF - *Resource Description Framework*) es una infraestructura que permite la codificación, el intercambio y la reutilización

[3]Namespaces

de estructuras de metadatos. RDF es una aplicación de XML que impone restricciones estructurales necesarias para proporcionar métodos no-ambiguos de expresión semántica. Adicionalmete RDF provee los medios para publicar vocabularios (legibles) para los humanos y procesables por las computadoras diseñados para impulsar la reutilización y la extensión de la semántica de los metadatos entre comunidades con información distinta.

Aunque se le conoce como un lenguaje, RDF es esencialmente un modelo de datos, en el cual su bloque básico de construcción es una terna objeto-atributo-valor, llamada *declaración* [Ant04].

Un modelo de datos necesita de una sintaxis concreta para ser representado y transmitido. En RDF la sintaxis puede emplear XML, por lo tanto hereda todos los beneficios de este lenguaje [W3C06c].

RDF es independiente del dominio, (no hay suposiciones acerca de un domino particular), sin embargo, es necesario que los usuarios definan su propia terminología en un lenguaje de esquemas llamado RDF-Schema (RDFS).

RDFS surge de la necesidad en las comunidades de descripción de recursos de tener la habilidad para decir ciertas cosas acerca de cierto tipos de recursos. Por ejemplo, para la descripción de recursos bibliográficos, se tienen atributos descriptivos como: autor, título, y tema, los cuales son comunes a todos estos recursos. La declaración de esas propiedades (atributos) y su correspondiente semántica se definen en el contexto RDF con un RDF-Schema. Un RDF-Schema no sólo define las propiedades de un recurso, sino también los tipos de recursos que serán descritos.

RDF-Schema está especificado en términos del modelo de información básico de RDF, el cual es una estructura de grafo que describe recursos y sus propiedades. Todos los vocabularios RDF comparten una estructura básica: describen clases de recursos y tipos de relaciones entre recursos, esta caracterísica común permite que las máquinas procesen de una manera mixta y clara los vocabularios.

RDF-Schema tiene una relación similar con RDF tal como XML-Schema con XML, sólo que con XML-Schema se restringe la estructura de los documentos XML, mientras que RDF-Schema define el vocabulario usado en los modelos de datos RDF. Además de definir el vocabulario, con RDFS se puede especificar qué propiedades se aplican a qué tipo de objetos y qué valores pueden tomar éstos y describir las relaciones entre objetos.

Los conceptos fundamentales de RDF son los recursos, las propiedades y las declaraciones. Los recursos son objetos o cosas de las que se quiere hablar. Las propiedades son un tipo especial de recursos, describen relaciones entre recursos. A continuación se muestra un ejemplo de una declaración:

Declaración: Ontología de registros es una subclase de ontología.

donde:

**Objeto** = Ontología de registros

**Atributo** = subclase

**Valor** = ontología

Esta declaración significa que todas las ontologías de registros son una subclase

de una ontología. Utilizando una terna y URLs para el objeto como para el atributo, la declaración anterior se representa como sigue:

**Ejemplo 2.4.**

```
("Ontologia de registros",
  http://www.midominio.org/subclass-of,
  "ontologia")
```

Del ejemplo anterior, se puede notar que la propiedad "subclase de" es representada por una URL, lo cual es válido ya que las propiedades pueden ser un objeto.

El ejemplo anterior al traducirse a sintaxis XML de RDF se presenta en el Ejemplo 2.5.

**Ejemplo 2.5.**

```
<?xml version="1.0" encoding="UTF-16"?>

<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:mydomain="http://www.mydomain.org/my-rdf-ns">
<rdf:Description
rdf:about="http://www.mydefinitions.au/ontologyofrecords">
  <mydomain:subclass-of>
    Ontology
  </mydomain:subclass-of>
</rdf:Description>
</rdf:RDF>
```

En el ejemplo anterior la primera línea especifica que se usará XML y la versión del mismo, la tercera línea indica el espacio de nombres de donde se obtendrán los recursos necesarios. La etiqueta $< rdf : Description >$ es una descripción acerca del recurso $ontologyofrecords$. La descripción debe darse en un cierto orden ya que XML impone una serialización, aunque el orden no es importante en el modelo abstracto de RDF. La etiqueta $< mydomain : subclass - of >$ es una propiedad que contiene el valor del objeto en cuestión.

### 2.4.1. Ventajas

Las ventajas principales al utilizar RDF se listan a continuación:

- "*Reification*". Capacidad de hacer declaraciones acerca de declaraciones.
- *Empleo de tipos de datos*. Dar a un dato una única interpretación en base a un tipo específico.

- *Sintaxis en XML.* Al estar representada la sintaxis de RDF en lenguaje XML, no es necesario aprender otro lenguaje [DM98].
- *W3C Recommendation.* RDF es una recomendación de la *World Wide Web Consortium* [W3C06c].

### 2.4.2. Desventajas

La desventaja principal de RDF radica en que las propiedades sólo son binarias. Lo grave de esta desventaja es que a menudo se usan predicados con más de dos argumentos, y, a pesar de que se pueden simular estos predicados, una representación con más de 2 argumentos es más natural que una simulación con varios predicados binarios, restringiendo así la expresividad de RDF [Ant04]. Por lo tanto, la semántica que se puede representar es simple.

## 2.5. Representación de ontologías con RDF y RDF-Schema

Los lenguajes de ontologías permiten a los usuarios escribir conceptualizaciones explícitas y formales para modelar dominios. Sus principales requerimientos son:

- Sintaxis bien definida.
- Semántica formal.
- Soporte para el razonamiento eficiente.
- Suficiente poder de expresividad.

La expresividad de RDF y RDF-Schema es limitada. La principal restricción del primero es el uso de predicados binarios, en tanto, el segundo sólo considera jerarquías de clases y propiedades, junto con restricciones que permiten definir rangos y dominios.

RDF y RDF-Schema permiten la representación de algunos conocimientos ontológicos. Las principales primitivas de modelado de RDF/RDFS conciernen a la organización de vocabularios de jerarquías prestablecidas: subclases y relaciones de subpropiedades, dominios y restricciones de rango e instancias de clases. Sin embargo, carecen de las siguientes características [Ant04]:

- Propiedades de alcance local.
- Clases disjuntas.
- Combinaciones booleanas de clases.

- Restricciones de cardinalidad.
- Características especiales de propiedades.

El grupo de trabajo de la W3C, identifica un número de casos de uso característicos para la web semántica que requieren mayor expresividad que la que ofrecen RDF y RDF-Schema. Idealmente, se considera que el conjunto completo de requerimientos para un lenguaje de ontologías incluye soporte eficiente para razonamiento y manejo de expresiones, a través de un lenguaje que pudiera resultar de la combinación de RDFS con una lógica completa [Ant04]. OWL es un lenguaje que sobrepasa algunas de estas carencias.

## 2.6. OWL

Grupos de investigación en Estados Unidos y Europa, identificaron la necesidad de un lenguaje más potente para el modelado de ontologías. A través de la fusión de sus iniciativas, propusieron el lenguaje DAML+OIL. El nombre es la unión de la propuesta estadounidense DAML-ONT y el lenguaje europeo OIL. El grupo de trabajo de la W3C considera a DAML+OIL como punto de partida para definir al lenguaje OWL (*Ontology Web Language*), el cual pretenden se convierta en el estándar para representar ontologías en la web semántica [Ant04].

OWL está diseñado para ser usado en aplicaciones de software que necesitan procesar el contenido de la información en lugar de únicamente representar información para los humanos. OWL facilita un mejor mecanismo de interpretabilidad del contenido Web que con los mecanismos admitidos por XML, RDF, RDF-Schema proporcionando un vocabulario adicional junto con una semántica formal [W3C04]. OWL puede ser usado para representar explícitamente el significado de términos en vocabularios y las relaciones entre esos términos, que en conjunto forman lo que se conoce como ontología.

El lenguaje de ontologías OWL es la parte final de una serie de recomendaciones de la W3C para establecer la web semántica [W3C04], los cuales son:

1. XML proporciona una sintaxis superficial para documentos estructurados, pero no impone restricciones semánticas en el significado de estos documentos.
2. XML-Schema es un lenguaje que se utiliza para restringir la estructura de los documentos XML, además de permitir definir tipos de datos en los documentos XML.
3. RDF es un modelo de datos para recursos y las relaciones entre ellos, proporciona una semántica simple.
4. RDF-Schema es un vocabulario utilizado para describir propiedades y clases de recursos RDF, junto con una semántica para jerarquías de generalizaciones de dichas propiedades y clases.

5. OWL añade más vocabulario para describir propiedades y clases que RDF, tales como:

   *a*) Relaciones entre clases.

   *b*) Cardinalidad.

   *c*) Igualdad.

   *d*) Mejor forma de escribir propiedades.

   *e*) Características de propiedades.

   *f*) Clases enumeradas.

Existen tres diferentes sublenguajes de OWL, cada uno satisface diferentes aspectos del conjunto completo de requerimientos [W3C04]. A continuación se describen sus características principales:

- *OWL Full*. OWL completo utiliza todas las instrucciones de OWL. Permite la combinación de esas instrucciones en formas arbitrarias con RDF y RDFS. Incluye la posibilidad de cambiar el significado predefinido de las instrucciones aplicando instrucciones del lenguaje. La ventaja de OWL Full es la compatibilidad total con RDF y RDFS. De esta forma, cualquier documento legal RDF es también un documento legal OWL *Full*, y cualquier conclusión válida RDF/RDFS es también una conclusión válida OWL Full. La desventaja principal es que el lenguaje se vuelve tan poderoso como incontrolable (indecidible), terminando con cualquier esperanza de asegurar un soporte *eficiente* de razonamiento.

- *OWL DL*. OWL con Lógica Descriptiva (*Description Logic*), surge para recuperar eficiencia computacional. Es un sublenguaje de OWL Full que restringe el uso de constructores de OWL y RDF, así asegura que el lenguaje corresponde a una descripción lógica bien definida pero limitada. OWL DL incluye todas las construcciones del lenguaje OWL, pero pueden ser usados sólo bajo ciertas restricciones. La ventaja de este lenguaje es que permite el soporte eficiente de razonamiento. Su desventaja es la pérdida de compatibilidad con RDF y RDFS. Con OWL DL sólo se puede decir que cualquier documento legal OWL DL es un documento legal RDF y no lo contrario.

- *OWL Lite*. OWL Ligero es una restricción de OWL DL porque considera sólo un subconjunto de constructores del lenguaje. OWL *Lite* excluye las clases enumeradas, las declaraciones para la disjunción y la cardinalidad arbitraria. Su ventaja es que es un lenguaje fácil de entender y fácil de implementar. El inconveniente es su restringida expresividad.

Una representación gráfica de estos diferentes lenguajes se muestra en la Figura 2.3, donde se observa su jerarquía.

La elección del sublenguaje a utilizar debe ser congruente con la aplicación para la cual se necesite. En esta tesis se elige OWL DL por las siguientes razones:

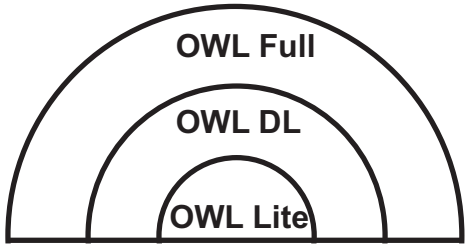


**Figura 2.3.** Sublenguajes de OWL.

- OWL DL incluye todas las construcciones del lenguaje OWL, conservando toda la expresividad OWL.
- Debido a que la traducción será utilizada por agentes de computadora (además de personas), es necesario tener un lenguaje que maneje eficientemente el razonamiento, conservando la mayor expresividad posible.
- La mayoría de los editores de OWL soportan OWL DL más que alguno de los otros sublenguajes.

## 2.7. Co4: Protocolo para la construcción cooperativa de ontologías

Co4 ( [Fra93], [Euz95] ) es un protocolo desarrollado en INRIA[4] para la construcción colaborativa de bases de conocimiento (BCs). Su objetivo es permitir la discusión y aceptación de conocimientos en las bases de conocimiento de un sistema.

El protocolo toma como entrada un conjunto de BCs organizadas en un árbol. Las hojas se llaman BCs de usuario y los nodos intermedios BCs de grupo. Las BCs de usuario no requieren conocimiento consensual, mientras que las de grupo representan el conocimiento consensual de sus hijos, (conocidos como BCs suscritos). Una BC puede estar suscrita a un solo grupo de BCs.

El sistema coordinado descrito en esta tesis implementa el protocolo Co4. La Figura 2.4 muestra la estructura del sistema. En ella, la ontología de registros 0 (OR 0) indica la ontología de registros final, mientras que las ontologías 1.x restantes (OR 1.x) se refieren a las ontologías que analiza cada integrante del grupo coordinado, el cual está formado por un moderador y varios integrantes, que juntos tienen como tarea principal, revisar y/o modificar la ontología de registros para formar una ontología final que representa el dominio indicado.

La Figura 2.4 representa el posible árbol de BCs. Cada una de las OR 1.x debe alcanzar un *consenso individual* para así poder alcanzar un consenso a nivel general de la ontología OR 0. Cada modificación propuesta en las ontologías OR 1.x deben

[4]Instituto National de Recherche en Informatique et en Automatique

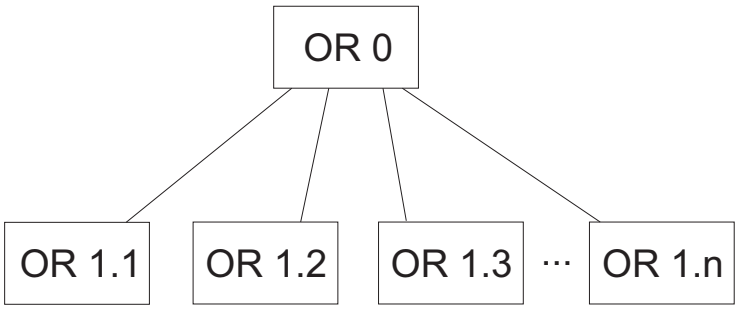


**Figura 2.4.** Estructura del sistema coordinado utilizando el protocolo Co4.

enviarse al nodo superior (nodo padre) para que así sea analizada por el resto del grupo coordinado. Cualquier modificación propuesta se acepta sólo si el resto del grupo está de acuerdo.

El siguiente ejemplo ilustra lo que ocurre cuando se lleva a cabo una modificación. Suponga que el sistema tiene 4 integrantes en el grupo coordinado y que el primer integrante propone una modificación a su ontología (OR 1.1). Estos son los pasos que se deben de seguir para que dicha modificación sea analizada (aceptada o rechazada):

- OR 1.1 envía un mensaje a OR 0.
- OR 0 transmite una petición de comentarios a todos sus ORs suscritos (OR 1.2, OR 1.3 ... OR 1.n).
- OR 1.2, OR 1.3 ... OR 1.n envían sus respuestas a OR 0 (aceptada o rechazada).
- OR 0 envía la decisión a todos sus ORs suscritos (modificación aceptada o modificación rechazada).

El protocolo Co4 considera la posibilidad de suscribir una nueva BC, aún cuando la construcción del resto de BCs ya haya comenzado. En este caso, el conocimiento que previamente ha sido aceptado por el grupo al cual la nueva BC se subscribe, será incluido en este nueva BC.

# Capítulo 3

# Diseño del sistema

---

> Mejor lo hago como le dije porque yo lo pensé así, y yo nomás pienso las cosas una vez, porque se me olvidan.
>
> *Gilberto Gómez Letras en No habrá final feliz, por PIT II*

## 3.1. Diseño del sistema

Cada uno de los módulos definidos en la Sección 1.4 guarda una estrecha relación, que se aprecia en la Figura 3.1. Cuando el proceso de revisión determina que es momento de traducir la ontología al siguiente lenguaje, pasa el control al módulo de construcción (relación 1). Si mediante el módulo de revisión se solicita una modificación a la ontología actual, entonces el control pasa al módulo de reconstrucción (relación 2), que junto con el modulo de visualización (relación 3), permiten que el participante realice los cambios necesarios. Los módulos de revisión, visualización y ayuda hacen uso del módulo de presentación (relación 4, 5 y 6) para mostrar a el usuario, mediante la interfaz, el estado actual del sistema.

La Figura 3.2 muestra el diagrama de actividades del sistema de traducción de ontologías de registros, que incluye los 7 componentes principales y el flujo de datos. Las entradas y salidas de estos componentes se describen en la Tabla 3.1:

## 3.2. Documentos requeridos

El proceso de traducción que realiza el Sistema de Traducción de Ontologías de Registros, necesita de archivos de referencia que le permitirán tener el control de los elementos de cada documento generado en la traducción. Los documentos de referencia son 4:

- Espacio de nombres. URI usado como prefijo de toda etiqueta XML exclusiva de una ontología de registros.

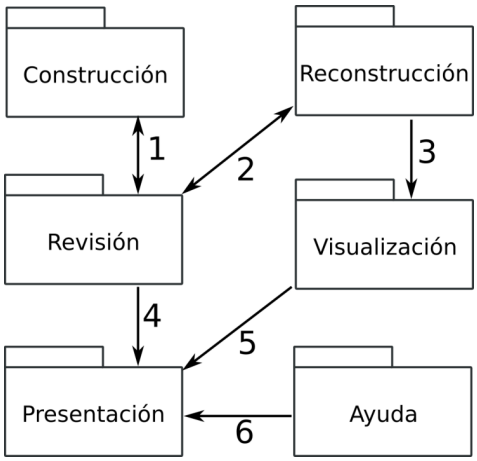


**Figura 3.1.** Relación entre los módulos del sistema.

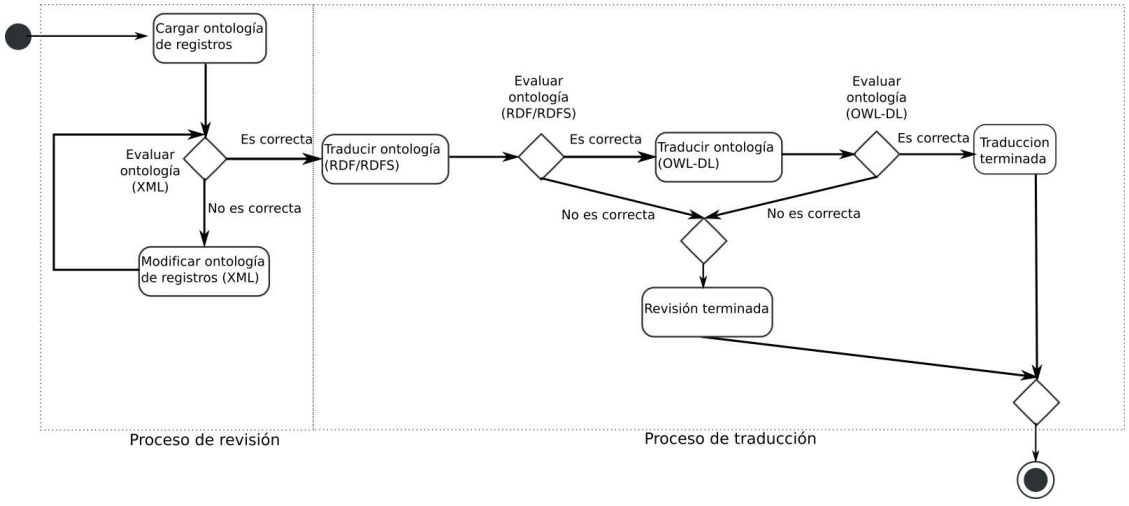


**Figura 3.2.** Diagrama de actividades del sistema de traducción de ontologías de registros.

| Componente | Descripción | Entradas | Salidas |
|---|---|---|---|
| Cargar la ontología de registros. | Este componente recupera la ontología de registros a traducir, la ontología debe incluir el espacio de nombres además del XML-Schema que indica su validez. | Ontología de registros. | Ontología de registros lista para el proceso de revisión y traducción. |
| Evaluar ontología (XML). | Mediante el proceso de revisión, este componente verifica que todos los miembros del grupo estén de acuerdo en que la ontología está lista para el siguiente paso. Si no estuvieran todos de acuerdo, tendría que hacerse una modificación. | Ontología de registros válida. | Ontología de registros aceptada por el grupo coordinado. |
| Modificar ontología en lenguaje XML. | Si existe algún cambio en la ontología válida, se realiza en este componente y se vuelve a evaluar hasta que sea aceptado por todo el grupo o rechazado definitivamente. | Ontología de registros válida. | Ontología de registros modificada. |
| Traducir ontología (RDF/RDFS). | Traduce la ontología de registros válida en lenguaje XML a una ontología escrita en lenguaje RDF/RDFS. | Ontología de registros válida en lenguaje XML. | Ontología en lenguaje RDF/RDFS. |
| Evaluar ontología (RDF/RDFS). | Evalua que la ontología en lenguaje RDF/RDFS sea aceptada por todos los miembros del grupo coordinado. | Ontología en lenguaje RDF/RDFS. | Ontología en lenguaje RDF/RDFS. |
| Traducir ontología (OWL-DL). | Realiza la traducción final de la ontología en lenguaje RDF/RDFS a una ontología en lenguaje OWL-DL. | Ontología en lenguaje RDF/RDFS. | Ontología en lenguaje OWL-DL. |
| Evaluar ontología (OWL-DL). | Evalua que la ontología en lenguaje OWL-DL sea aceptada por todos los miembros del grupo coordinado. | Ontología en lenguaje OWL-DL. | Ontología en lenguaje OWL-DL. |

**Tabla 3.1.** Descripción de los componentes del sistema de traducción de ontologías de registros.


**Figura 3.3.** Página XHTML que representa el espacio de nombres para una ontología de registros.

- XML-Schema. Valida el archivo XML para que sea un archivo que represente una ontología de registros.
- RDF-Schema. Sirve como referencia para los archivos RDF que deseen representar una ontología de registros.
- Modelo OWL. Define la estructura que debe tener un archivo OWL de instancias que represente una ontología de registros.

Cada uno de los archivos anteriores se muestra en secciones posteriores del trabajo de tesis.

### 3.2.1. El espacio de nombres

Un espacio de nombres es identificado por una referencia a una URI, dentro de él, cada nombre y atributo es único. Esto permite que se pueda utilizar el mismo nombre con diferente significado en un mismo documento sin que exista confusión [W3C06b].

El espacio de nombres creado para definir los nombres y atributos de una ontología de registros está localizado en la siguiente URI:

`http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml`

El estándar de construcción de un espacio de nombres establece que el documento debe estar escrito en lenguaje XHTML, y como contenido, debe tener una descripción del dominio que representa [W3C06b]. La Figura 3.3 muestra el contenido de la URI que representa el espacio de nombres.

# Capítulo 4

# Representación de ontologías de registros

---

> La investigación debe apropiarse de la materia en detalle.
>
> *Karl Marx*

## 4.1. Trabajo previo

### 4.1.1. Estructura de una ontología de registros

En la Sección 2.1 se presentó la definición de ontología de registros como una estructura jerárquica que agrupa registros de metadatos similares. Para definir la estructura de una ontología se deben de tomar en cuenta las siguientes afirmaciones:

- Una ontología de registros se genera a partir de un algoritmo jerárquico y parámetros para el mismo.
- Una ontología de registros contiene 1 o más grupos.
- Un grupo tiene asociado un nivel en la jerarquía.
- Un grupo está identificado por etiquetas.
- Un grupo puede tener 0 o más registros.
- Un registro contiene título, tema, descripción, identificador, url, proveedor de datos, formato de metadatos y una estampa de tiempo.

Una forma gráfica de mostrar la estructura que debe tener una ontología de registros se presenta en la Figura 4.1.

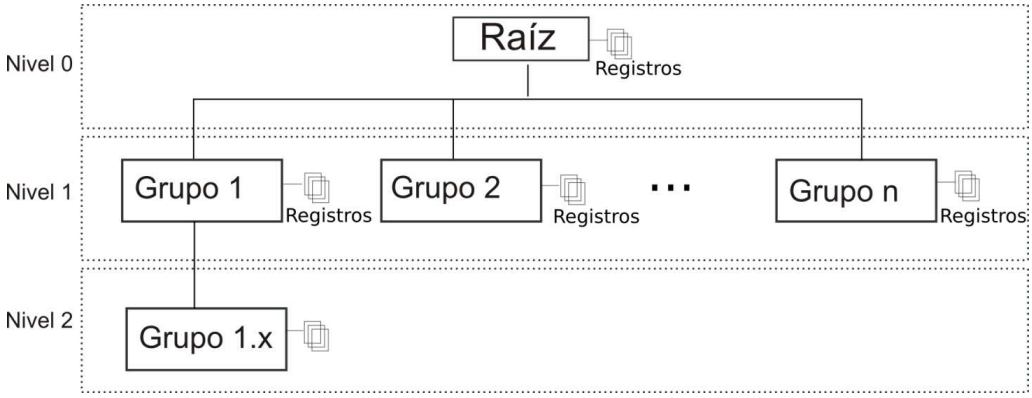


**Figura 4.1.** Estructura de una ontología de registros.

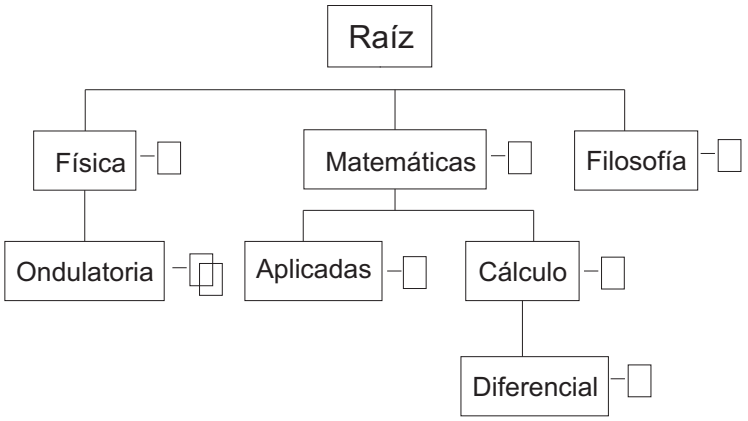


**Figura 4.2.** Ejemplo de una ontología de registros, tomando en cuenta los libros de Física, Matemáticas y Filosofía.

### 4.1.2. Un ejemplo de ontología de registros

Si se elije como dominio algunos libros de Física, Matemáticas y Filosofía de una biblioteca, se puede formar una ontología de registros como la que se muestra en la Figura 4.2.

En el ejemplo de la Figura 4.2, se puede observar que existen 8 grupos ó *clusters* (en 4 niveles), donde cada uno de ellos está caracterizado por la etiqueta con la que está marcado y la de sus antecesores. Por ejemplo, el único grupo del nivel 3 está representado por las etiquetas de sus 2 antecesores más la propia, por tanto, los registros almacenados en este grupo tienen en común los términos *matemáticas, cálculo* y *diferencial*. En este ejemplo, todos los grupos almacenan un sólo registro a excepción del grupo *Física Ondulatoria*, el cual almacena 2.

Una ontología de registros puede representarse en el lenguaje XML (el cual es el punto de partida de esta tesis), anidando etiquetas para representar la jerarquía de elementos. De forma que si se considera el ejemplo de la Figura 4.2 y éste se representa en XML se obtiene el código del Ejemplo 4.1.

**Ejemplo 4.1.**

```
<?xml version="1.0" encoding="UTF-8"?>
<ontologyofrecords date="2006-04-21">
<algorithm name="FIHC" globalsupport="25" clustersupport="30"/>
  <cluster>
    <label>Física</label>
    <level>1</level>
    <record>
      <title>Física I</title>
      <subject>Física</subject>
      <description>Resumen o prólogo</description>
      <identifier>oai:libroUTM:1</identifier>
      <url></url>
      <dataprovider>http://cs1.ist.psu.edu/cgi-bin/oai.cgi</dataprovider>
      <metadataformat>oai_dc</metadataformat>
      <datestamp>2006-01-01</datestamp>
    </record>
  </cluster>
  <cluster>
      <label>Física ondulatoria</label>
      <level>2</level>
      <record>
        <title>Física Odulatoria II</title>
        <subject>Física Ondulatoria</subject>
        <description></description>
        <identifier>oai:libroUTM:2</identifier>
        <url></url>
        <dataprovider>http://cs1.ist.psu.edu/cgi-bin/oai.cgi</dataprovider>
        <metadataformat>oai_dc</metadataformat>
```

```
        <datestamp>2006-01-01</datestamp>
      </record>
      <record>
        <title>Principios de Física Ondulatoria</title>
        <subject>Principios de Física Ondulatoria</subject>
        <description>/description>
        <identifier>oai:libroUTM:3</identifier>
        <url></url>
        <dataprovider>http://cs1.ist.psu.edu/cgi-bin/oai.cgi</dataprovider>
        <metadataformat>oai_dc</metadataformat>
        <datestamp>2006-01-01</datestamp>
      </record>
    </cluster>

    ...

</ontologyofrecords>
```

La primera instrucción del ejemplo 4.1 indica la versión de XML así como el tipo de juego de caracteres utilizados en el documento. El resto del código muestra el árbol de elementos que representa a la ontología de registros. Las etiquetas por sí solas indican el elemento que representan en la ontología de registros. A continuación se presenta una breve descripción de éstas.

- <ontologyofrecords date=‘‘2006-04-21’’> representa una ontología de registros. El atributo `date` indica la fecha de creación de ésta
- <algorithm name=‘‘FIHC’’ globalsupport=‘‘25’’ clustersupport=‘‘30’’ / > representa el algoritmo utilizado para construir la ontología, su valor es fijo e igual a FIHC, los atributos indican el nombre y valor para los atributos `globalsupport` y `clustersupport` [Med05]
- <cluster> representa el elemento grupo de una ontología de registros
  - <label> contiene los términos que representan las etiqueta del grupo
  - <level> indica el nivel que tienen el grupo en el árbol de grupos
  - <record> representa un registro del grupo en cuestión
    - <title> representa el nombre del recurso descrito por un registro
    - <subject> indica el tema del recurso
    - <description> contiene la descripción del recurso
    - <identifier> almacena el identificador único para el registro
    - <url> indica el url del registro
    - <dataprovider> indica de qué proveedor de datos se extrajo la información del registro

- <metadataformat> representa el formato de metadatos del registro
- <datestamp> contiene la fecha de creación o la última fecha de modificación del registro

Los elementos anteriores describen una ontología de registros, sin embargo, no todos son obligatorios. Los elementos indispensables son: `ontologyofrecords`, `algorithm`, `cluster`, `label` y `level`.

## 4.2. Implementación

### 4.2.1. Creación de un archivo de validación para las ontologías de registros

Una vez definida una ontología de registros en XML, es necesario crear un archivo de validación para verificar que un documento XML represente la estructura de una ontología de registros, tal y como se definió anteriormente.

El primer archivo de validación creado fue un DTD (*Document Type Definition*) [Med06], el cual define el orden y la cardinalidad de los elementos. Éste se muestra en el Ejemplo 4.2.

**Ejemplo 4.2.**

```
1. <!ELEMENT ontologyofrecords (algorithm, cluster+)>
2. <!ATTLIST ontologyofrecords date CDATA #REQUIRED >
3. <!ELEMENT algorithm EMPTY>
4. <!ATTLIST algorithm
             name CDATA #FIXED "FIHC"
             globalsupport  CDATA #REQUIRED
             clustersupport CDATA #REQUIRED >
5. <!ELEMENT cluster (label, level, record*)>
6. <!ELEMENT label (#PCDATA)>
7. <!ELEMENT level (#PCDATA)>
8. <!ELEMENT record (title, subject?, description?,
             identifier, url, dataprovider,
             metadataformat, datestamp)>
9. <!ELEMENT title (#PCDATA)>
10. <!ELEMENT subject (#PCDATA)>
11. <!ELEMENT description (#PCDATA)>
12. <!ELEMENT identifier (#PCDATA)>
13. <!ELEMENT url (#PCDATA)>
14. <!ELEMENT dataprovider (#PCDATA)>
15. <!ELEMENT metadataformat (#PCDATA)>
16. <!ELEMENT datestamp (#PCDATA)>
17. <!ENTITY generatedBy "OntoSIR Version 1.0">
```

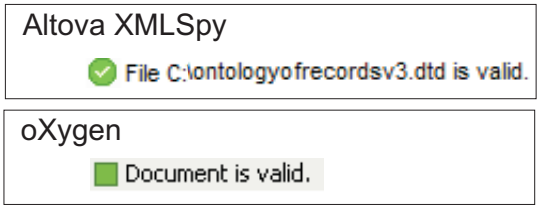


**Figura 4.3.** Mensajes de validación para el archivo DTD del ejemplo 4.2 en el software Altova XMLSpy y oXygen.

El DTD del Ejemplo 4.2 está validado de acuerdo al software Altova XML Spy[1] y oXygen[2]. Las pantallas que muestran los mensajes de validación se observan en la Figura 4.3

El Ejemplo 4.2 contiene las restricciones de orden de las etiquetas así como la cardinalidad de cada una de ellas. Algunas observaciones importantes son las siguientes:

- La etiqueta `ontologyofrecords` debe tener anidada una etiqueta `algorithm` y una o más etiquetas `cluster` (línea 1).
- La etiqueta `algorithm` contiene exactamente una etiqueta `name` con valor fijo igual a FIHC (línea 4) y los 2 atributos del algoritmo.
- La etiqueta `cluster` requiere por lo menos de una etiqueta `label` y `level` y 0 o más etiquetas `record`.
- La etiqueta `record` debe tener anidadas una etiqueta `title`, 0 ó 1 etiqueta `subject`, 0 o 1 etiqueta `description`, y una etiqueta `identifier`, `url`, `dataprovider`, `metadataformat` y `datestamp`.
- La última instrucción indica que la ontología de registros fue generada por la versión 1.0. de la aplicación OntoSIR.

Para indicar que un archivo XML está validado por el DTD creado, se necesita incluir la siguiente instrucción antes del elemento `ontologyofrecords` del XML:

`<!DOCTYPE ontologyofrecords SYSTEM ‘‘c:\ontologyofrecordsv3.dtd’’>`

La inclusión de esta instrucción en el XML se presenta en el Ejemplo 4.3.

**Ejemplo 4.3.**

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE ontologyofrecords SYSTEM "C:ontologyofrecordsv3.dtd">
<ontologyofrecords date="2006-04-21">
<algorithm name="FIHC" globalsupport="25" clustersupport="30"/>
```

[1] *http : //www.altova.com/products/xmlspy*
[2] *http : //www.oxygensoftware.com*

```
  <cluster>
    <label>FÍSICA</label>
    <level>1</level>
    <record>
    ...
</ontologyofrecords>
```

Una alternativa al DTD para validar documentos XML es el empleo de un XML-Schema, el cual además de establecer restricciones en cuanto al orden y la cardinalidad, también establece tipos de datos para las etiquetas, entre otros beneficios. Al transformar el DTD a un XML-Schema e indicar los tipos de datos se obtiene el XML-Schema del Ejemplo 4.4.

**Ejemplo 4.4.**

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<xs:schema
xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
<xs:element name="ontologyofrecords">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="algorithm"/>
      <xs:element ref="cluster" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attributeGroup ref="attlist.ontologyofrecords"/>
    </xs:complexType>
</xs:element>
<xs:attributeGroup name="attlist.ontologyofrecords">
  <xs:attribute name="date" type="xs:date" use="required"/>
</xs:attributeGroup>
<xs:element name="algorithm">
  <xs:complexType>
    <xs:attributeGroup ref="attlist.algorithm"/>
  </xs:complexType>
</xs:element>
<xs:attributeGroup name="attlist.algorithm">
  <xs:attribute name="name" default="FIHC">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="FIHC"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
 <xs:attribute name="globalsupport" use="required"/>
 <xs:attribute name="clustersupport" use="required"/>
</xs:attributeGroup>
```

```
<xs:element name="cluster">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="label"/>
      <xs:element ref="level"/>
      <xs:element ref="record" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="label" type="xs:string"/>
<xs:element name="level" type="xs:integer"/>
<xs:element name="record">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="title"/>
      <xs:element ref="subject" minOccurs="0"/>
      <xs:element ref="description" minOccurs="0"/>
      <xs:element ref="identifier"/>
      <xs:element ref="url"/>
      <xs:element ref="dataprovider"/>
      <xs:element ref="metadataformat"/>
      <xs:element ref="datestamp"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="title" type="xs:string"/>
<xs:element name="subject" type="xs:string"/>
<xs:element name="description" type="xs:string"/>
<xs:element name="identifier" type="xs:string"/>
<xs:element name="url" type="xs:anyURI"/>
<xs:element name="dataprovider" type="xs:string"/>
<xs:element name="metadataformat" type="xs:string"/>
<xs:element name="datestamp" type="xs:date"/>
</xs:schema>
```

Un XML-Schema está escrito en XML. Requiere de otra etiqueta para indicar que el archivo representa un XML-Schema, <`xs:schema`>. En diferentes partes del código se puede ver cómo se le asignan tipos de datos a los diferentes elementos de la ontología de registros. Por ejemplo, el elemento `level` es de tipo `integer` y el elemento `title` de tipo `string`. Es importante resaltar que todas las etiquetas, menos la primera, comienzan con el prefijo `xs:` el cual indica el espacio de nombres (*namespace*) donde están definidas dichas etiquetas. En particular, este espacio de nombres indica que todas las etiquetas necesarias para escribir el Schema están definidas en `http://www.w3.org/2001/XMLSchema`.

Al validar el XML-Schema del Ejemplo 4.4 utilizando Altova XMLSpy y oXygen

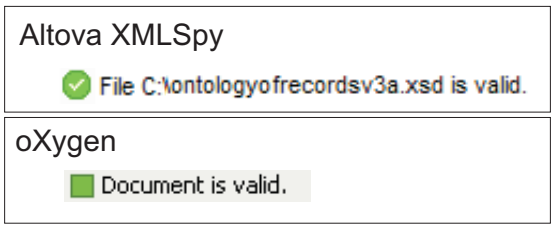


**Figura 4.4.** Mensajes de validación para el archivo XML-Schema en el software Altova XMLSpy y oXygen.

se obtienen los mensajes de la Figura 4.4.

Para indicar que un documento XML debe ser validado por el XML-Schema, es necesario incluir la propiedad `xsi:noNamespaceSchemaLocation` en la etiqueta `ontologyofrecords`. El valor de la propiedad debe ser la ruta del XML-Schema. Al incluir la propiedad correspondiente, se obtiene el código del Ejemplo 4.5, el cual representa la misma ontología de registros de los ejemplos anteriores validada por el XML-Schema ubicado en este ejemplo en: `C:\ontologyofrecordsv3a.xsd`.

**Ejemplo 4.5.**

```
<?xml version="1.0" encoding="UTF-8"?>
<ontologyofrecords
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="C:ontologyofrecordsv3a.xsd"
    date="2006-04-21">
<algorithm name="FIHC" globalsupport="25" clustersupport="30"/>
  <cluster>
    <label>fisica</label>
    <level>1</level>
    <record>
      ...
    </record>
  </cluster>
  <cluster>
      <label>fisica ondulatoria</label>
      <level>2</level>
      <record>
        ...
      </record>
      <record>
        ...
      </record>
    </cluster>
    ...
</ontologyofrecords>
```

Para evitar ambigüedades de las etiquetas de la ontología de registros con otras etiquetas, es conveniente definir un espacio de nombres desde donde se puede hacer referencia sólo a las etiquetas propuestas para una ontología de registros.

Un espacio de nombres no es un archivo restrictivo que contenga definiciones de las etiquetas, solamente es un URL que servirá como valor para el prefijo que se utiliza en algún archivo XML donde se usen etiquetas que describan el dominio de una ontología de registros.

Para incluir el espacio de nombres en nuestro archivo XML validado con XML-Schema Es necesario hacer 3 cambios:

- Agregar al XML-Schema los siguientes atributos en la etiqueta *xs:schema*:
  - `xmlns="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml"` con el que se indica que se incluirá el espacio de nombres de la ontología de registros.
  - `targetNamespace="http://www.utm.mx/~sigepr/recursos/ontologyofre cords.xhtml"` el cual indica que el espacio de nombres que se usará está en el URL `http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml`.
- Eliminar el atributo `xsi:noNamespaceSchemaLocation` de la etiqueta `ontologyo frecords`.
- Agregar en el XML los siguientes atributos a la etiqueta `ontologyofrecords`:
  - `xmlns:or="http://www.utm.mx/~sigepr/recursos/ontologyofrecords – .xhtml"` el cual indica que se va a usar el espacio de nombres. con prefijo `or` cuyo URL es: `http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml`
  - `xsi:schemaLocation="http://www.utm.mx/~sigepr/recursos/ontologyo frecords.xhtml" http://www.utm.mx/~sigepr/recursos/ontologyofre – cordsv3a.xsd` el cual indica la localización del XML-Schema con que se debe validar. el XML

Al realizar los cambios antes mencionados en los archivos correspondientes obtendremos los códigos mostrados en el Ejemplo 4.6.

**Ejemplo 4.6.**
`Archivo XML-Schema`

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<xs:schema
xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified">
<xs:element name="ontologyofrecords">
  <xs:complexType>
    ...
```

```
  </xs:complexType>
</xs:element>
<xs:attributeGroup name="attlist.ontologyofrecords">
  <xs:attribute name="date" type="xs:date" use="required"/>
</xs:attributeGroup>
<xs:element name="algorithm">
  <xs:complexType>
    <xs:attributeGroup ref="attlist.algorithm"/>
  </xs:complexType>
</xs:element>
<xs:attributeGroup name="attlist.algorithm">
  <xs:attribute name="name" default="FIHC">
    <xs:simpleType>
      ...
    </xs:simpleType>
  </xs:attribute>
 <xs:attribute name="globalsupport" use="required"/>
 <xs:attribute name="clustersupport" use="required"/>
</xs:attributeGroup>
<xs:element name="cluster">
  <xs:complexType>
    <xs:sequence>
      ...
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="label" type="xs:string"/>
<xs:element name="level" type="xs:integer"/>
<xs:element name="record">
  <xs:complexType>
    <xs:sequence>
      ...
    </xs:sequence>
  </xs:complexType>
</xs:element>

. . .

</xs:schema>
```

Archivo XML

```
<?xml version="1.0" encoding="UTF-8"?>
<ontologyofrecords
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xmlns:or="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml"
xsi:schemaLocation="http://www.utm.mx/~sigepr/recursos/ontologyofrecord
s.xhtml
http://www.utm.mx/~sigepr/recursos/ontologyofrecordsv3a.xsd" date="2006
-04-21">
<algorithm name="FIHC" globalsupport="25" clustersupport="30"/>
  <cluster>
    <label>fisica</label>
    <level>1</level>
    <record>
      ...
    </record>
  </cluster>
  <cluster>
      <label>fisica ondulatoria</label>
      <level>2</level>
      <record>
        ...
      </record>
      <record>
        ...
      </record>
    </cluster>

    ...

</ontologyofrecords>
```

El Ejemplo 4.6 es la versión final de nuestro XML válido, y es del que partiremos para comenzar la traducción hacia el lenguaje OWL.

## 4.3. Traducción de XML a RDF

RDF es un estándar de la W3C utilizado para describir recursos Web, provee un modelo de datos y una sintáxis para que partes independientes puedan intercambiar información. RDF está diseñado para ser leído y comprendido por computadoras. RDF se escribe conforme a las reglas XML, contribuyendo con sus propios elementos para definir recursos y relaciones entre ellos, entre otras cosas.

Un documento RDF debe estar basado en un documento RDF-Schema que indica la forma de construir un documento, éste último necesita ser creado primero para que el documento RDF pueda ser escrito conforme a las reglas que se establezcan.

### 4.3.1. El archivo RDF y RDF-Schema

En RDF la etiqueta raíz es `RDF` y el elemento de construcción básico es `Description`, el cual describe un recurso en base a sus atributos. Las declaraciones son representadas por tuplas (objeto, atributo, valor) o en terminos de recursos: (recurso, propiedad, valor de la propiedad). El Ejemplo 4.7 muestra un breve código para la definición de un recurso en RDF.

**Ejemplo 4.7.**

```
<RDF>
  <Description about="http://www.utm.mx/ontologyofrecords">
    <author>Maria Auxilio Medina Nieto</author>
    <homepage>http://www.utm.mx/mh</homepage>
  </Description>
</RDF>
```

En el Ejemplo 4.7 podemos observar que se describe el recurso cuya URL es `http://www.utm.mx/ontologyofrecords`, el cual tiene dos atributos, el atributo *autor* con valor *Maria Auxilio Medina Nieto* y el atributo *homepage* con valor `http://www.utm.mx/mh`. Las tuplas que representan este ejemplo son (`http://www.utm.mx/ontologyofrecords`, autor, Maria Auxilio Medina Nieto) y (`http://www.utm.mx/ontologyofrecords`, homepage, `http://www.utm.mx/mh`) respectivamente.

Antes de realizar la traducción del archivo XML del Ejemplo 4.6 a un archivo RDF se crea un RDF-Schema [Med07], el cual provee un marco de trabajo para describir clases de una aplicación específica, en este caso una ontología de registros, además de especificar los tipos de datos para cada valor de cada atributo e indicar el espacio de nombres a usar. El código del Ejemplo 4.8 muestra el archivo RDF-Schema desarrollado para definir un RDF de una ontología de registros [Med06].

**Ejemplo 4.8.**

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [ <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#"> ]>
<rdf:RDF
xml:base="http://www.utm.mx/~sigepr/recursos/ontologyofrecords/ontology
ofrecords"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
  <rdfs:Class rdf:ID="OntologyOfRecords"/>
  <rdfs:Datatype rdf:about="&xsd;date"/>
  <rdfs:Datatype rdf:about="&xsd;string"/>
  <rdfs:Datatype rdf:about="&xsd;integer"/>
  <rdfs:Datatype rdf:about="&xsd;anyURI"/>
  <rdfs:Property rdf:ID="isDescribedByDate">
    <rdfs:domain rdf:resource="#OntologyOfRecords"/>
```

```
    <rdfs:range rdf:resource="&xsd;date"/>
  </rdfs:Property>
<rdfs:Class rdf:ID="Calgorithm"/>
<rdfs:Property rdf:ID="isDescribedByClusterSupport">
  <rdfs:domain rdf:resource="#Calgorithm"/>
  <rdfs:range rdf:resource="&xsd;integer"/>
</rdfs:Property>
<rdfs:Property rdf:ID="isDescribedByGlobalSupport">
  <rdfs:domain rdf:resource="#Calgorithm"/>
  <rdfs:range rdf:resource="&xsd;integer"/>
</rdfs:Property>
<rdfs:Property rdf:ID="isDescribedByName">
  <rdfs:domain rdf:resource="#Calgorithm"/>
  <rdfs:range rdf:resource="&xsd;string"/>
</rdfs:Property>
<rdfs:Property rdf:ID="hasAlgorithm">
  <rdfs:domain rdf:resource="#OntologyOfRecords"/>
  <rdfs:range rdf:resource="#Calgorithm"/>
</rdfs:Property>
<rdfs:Class rdf:ID="Ccluster"/>
<rdfs:Property rdf:ID="hasCluster">
  <rdfs:domain rdf:resource="#OntologyOfRecords"/>
  <rdfs:range rdf:resource="#Ccluster"/>
</rdfs:Property>
<rdfs:Class rdf:ID="Clabel"/>
<rdfs:Property rdf:ID="hasLabel">
  <rdfs:domain rdf:resource="#Ccluster"/>
  <rdfs:range rdf:resource="&xsd;string"/>
</rdfs:Property>
<rdfs:Class rdf:ID="Clevel"/>
<rdfs:Property rdf:ID="hasLevel">
  <rdfs:domain rdf:resource="#Ccluster"/>
  <rdfs:range rdf:resource="&xsd;integer"/>
</rdfs:Property>
<rdfs:Class rdf:ID="Crecord"/>
<rdfs:Property rdf:ID="hasRecord">
  <rdfs:domain rdf:resource="#Ccluster"/>
  <rdfs:range rdf:resource="#Crecord"/>
</rdfs:Property>
<rdfs:Class rdf:ID="Ctitle"/>
<rdfs:Property rdf:ID="hasTitle">
  <rdfs:domain rdf:resource="#Crecord"/>
  <rdfs:range rdf:resource="&xsd;string"/>
</rdfs:Property>
<rdfs:Class rdf:ID="Csubject"/>
<rdfs:Property rdf:ID="hasSubject">
```

```
    <rdfs:domain rdf:resource="#Crecord"/>
    <rdfs:range rdf:resource="&xsd;string"/>
  </rdfs:Property>
  <rdfs:Class rdf:ID="Cdescription"/>
  <rdfs:Property rdf:ID="hasDescription">
    <rdfs:domain rdf:resource="#Crecord"/>
    <rdfs:range rdf:resource="&xsd;string"/>
  </rdfs:Property>
  <rdfs:Class rdf:ID="Cindentifier"/>
  <rdfs:Property rdf:ID="hasIdentifier">
    <rdfs:domain rdf:resource="#Crecord"/>
    <rdfs:range rdf:resource="&xsd;string"/>
  </rdfs:Property>
  <rdfs:Class rdf:ID="Curl"/>
  <rdfs:Property rdf:ID="hasUrl">
    <rdfs:domain rdf:resource="#Crecord"/>
    <rdfs:range rdf:resource="&xsd;anyURI"/>
  </rdfs:Property>
  <rdfs:Class rdf:ID="Cdataprovider"/>
  <rdfs:Property rdf:ID="hasDataProvider">
    <rdfs:domain rdf:resource="#Crecord"/>
    <rdfs:range rdf:resource="&xsd;string"/>
  </rdfs:Property>
  <rdfs:Class rdf:ID="Cmetadataformat"/>
  <rdfs:Property rdf:ID="hasMetadataFormat">
    <rdfs:domain rdf:resource="#Crecord"/>
    <rdfs:range rdf:resource="&xsd;string"/>
  </rdfs:Property>
  <rdfs:Class rdf:ID="Cdatestamp"/>
  <rdfs:Property rdf:ID="hasDatestamp">
    <rdfs:domain rdf:resource="#Crecord"/>
    <rdfs:range rdf:resource="&xsd;date"/>
  </rdfs:Property>
</rdf:RDF>
```

En el Ejemplo 4.8 podemos observar bajo la etiqueta `xml` que aparece una instrucción `<!DOCTYPE ... >` la cual indica que se va a usar el espacio de nombres del XML-Schema, para poder utilizar los tipos de datos propios de un Schema. En la etiqueta raíz `RDF` inicia la definición del documento, luego aparecen 3 atributos `xml:base` , `xmlns:rdf` y `xmlns:rdfs` los cuales indican el espacio de nombres base, los espacios de nombres de RDF y RDFS respectivamente. Las etiquetas que siguen definen jerárquicamente los elementos de una ontología de registros, incluyendo los tipos de datos para los mismos.

RDF-Schema utiliza las etiquetas `Class` para identificar un elemento, las etiquetas `Property` para indicar que un elemento contiene datos de un determinado

tipo, mientras que la etiqueta `domain` relaciona a un elemento final con una clase, mostrando la relación objeto - atributo - valor. Cada elemento del archivo XML se traduce a un elemento equivalente representado por clases, propiedades y la relación de las mismas.

El RDF-Schema del Ejemplo 4.8 indica la forma que debería tener un archivo RDF que quiera representar una ontología de registros, también se ocupa para evitar alguna confusión o ambigüedad en la creación de este tipo de ontologías. La aplicación del RDF-Schema es diferente de la aplicación del XML-Schema, ya que esta última sirve para validar un documento XML y el RDF-Schema sirve como guía para la creación de ontologías de registros, por lo tanto, no valida estrictamente un RDF.

El archivo RDF creado que cumple con el RDF-Schema se muestra en el Ejemplo 4.9, en el cual se puede ver que la estructura coincide con la propuesta, además que los tipos de datos para los elementos son congruentes con los establecidos en el RDF-Schema.

**Ejemplo 4.9.**

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
xmlns:or="http://www.utm.mx/~sigepr/recursos/ontologyofrecords/ontology
ofrecords#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
<rdf:Description rdf:ID="OntologyOfRecords">
 <or:hasAlgorithm or:isDescribedByClusterSupport="30"
       or:isDescribedByGlobalSupport="25"
     or:isDescribedByName="FIHC"/>
  <or:hasCluster>
    <rdf:Description
      rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords/c
      luster">
     <or:hasLabel> fisica </or:hasLabel>
     <or:hasLevel> 1 </or:hasLevel>
     <or:hasRecord>
      <rdf:Description
        rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords
        /record">
       <or:hasTitle> Fisica I </or:hasTitle>
       <or:hasSubject> Fisica </or:hasSubject>
       <or:hasDescription>  </or:hasDescription>
       <or:hasIdentifier> oai:libroUTM:1 </or:hasIdentifier>
       <or:hasUrl> </or:hasUrl>
       <or:hasDataProvider>
           http://cs1.ist.psu.edu/cgi-bin/oai.cgi
       </or:hasDataProvider>
       <or:hasMetadataFormat> oai_dc </or:hasMetadataFormat>
```

```
      <or:hasDateStamp> 2006-01-01 </or:hasDateStamp>
     </rdf:Description>
    </or:hasRecord>
   </rdf:Description>
  </or:hasCluster>
  <or:hasCluster>
   <rdf:Description
    rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords/c
    luster">
    <or:hasLabel> fisica ondulatoria </or:hasLabel>
    <or:hasLevel> 2 </or:hasLevel>
    <or:hasRecord>
     <rdf:Description
      rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords
      /record">
      <or:hasTitle> Fisica Odulatoria II</or:hasTitle>
      <or:hasSubject>  Fisica Odulatoria </or:hasSubject>
      <or:hasDescription>  </or:hasDescription>
      <or:hasIdentifier> oai:libroUTM:2 </or:hasIdentifier>
      <or:hasUrl> </or:hasUrl>
      <or:hasDataProvider>
          http://cs1.ist.psu.edu/cgi-bin/oai.cgi
       </or:hasDataProvider>
      <or:hasMetadataFormat> oai_dc </or:hasMetadataFormat>
      <or:hasDateStamp> 2006-01-01 </or:hasDateStamp>
     </rdf:Description>
    </or:hasRecord>
    <or:hasRecord>
     <rdf:Description
      rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords
      /record">
      <or:title> Principios de la Fisica Odulatoria </or:title>
      <or:subject> Principios de Fisica Ondulatoria </or:subject>
      <or:description> </or:description>
      <or:identifier> oai:libroUTM:3 </or:identifier>
      <or:url> </or:url>
      <or:dataprovider>
          http://cs1.ist.psu.edu/cgi-bin/oai.cgi
       </or:dataprovider>
      <or:metadataformat> oai_dc </or:metadataformat>
      <or:datestamp> 2006-01-01 </or:datestamp>
     </rdf:Description>
    </or:hasRecord>
   </rdf:Description>
  </or:hasCluster>
  <or:hasCluster>
```

```
    <rdf:Description
      rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords/c
      luster">
     <or:hasLabel> matematicas </or:hasLabel>
     <or:hasLevel> 1 </or:hasLevel>
     <or:hasRecord>
     <rdf:Description
       rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords/
       record">
      <or:hasTitle> Matematicas</or:hasTitle>
      <or:hasSubject> Matematicas </or:hasSubject>
      <or:hasDescription> Abstract o prologo </or:hasDescription>
      <or:hasIdentifier> oai:libroUTM:4 </or:hasIdentifier>
      <or:hasUrl> </or:hasUrl>
      <or:hasDataProvider>
          http://cs1.ist.psu.edu/cgi-bin/oai.cgi
      </or:hasDataProvider>
      <or:hasMetadataFormat> oai_dc </or:hasMetadataFormat>
      <or:hasDateStamp> 2006-01-01 </or:hasDateStamp>
     </rdf:Description>
    </or:hasRecord>
   </rdf:Description>
  </or:hasCluster>
  <or:hasCluster>
    ...
  </or:hasCluster>
  <or:hasCluster>
    ...
  </or:hasCluster>
  <or:hasCluster>
    ...
  </or:hasCluster>
  <or:hasCluster>
    ...
  </or:hasCluster>
</rdf:Description>
</rdf:RDF>
```

Todas las etiquetas y atributos del Ejemplo 4.9 tienen la misma función que la que tuvieron en archivos anteriores. Como se puede ver, para la traducción de XML a RDF se necesitó definir un RDF-Schema y luego seguirlo para crear en base a etiquetas *Description* una representación en RDF que representa una ontología de registros. De esta forma podemos afirmar que una ontología de registros representada en XML puede ser traducida a RDF con ayuda de un RDF-Schema y espacios

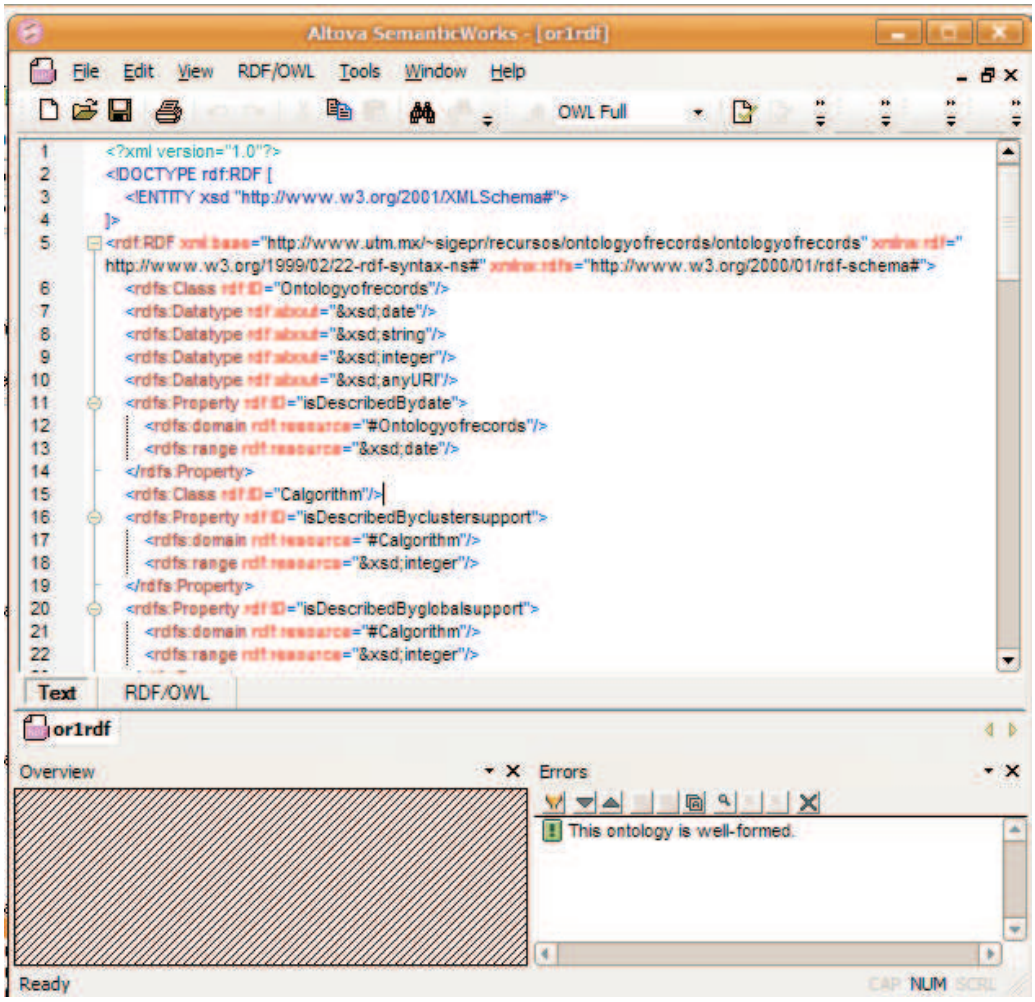


**Figura 4.5.** Pantalla del software Altova Semantics Works al reconocer el archivo RDF.

de nombres. El archivo RDF es leído y manipulado por editores RDF.

Para verificar que el RDF generado fuera correcto, se utilizó el software Altova Semantics Works. Al abrir y validar el archivo RDF se obtuvieron las imágenes de la Figura 4.5 con las cuales podemos afirmar que el archivo RDF obtenido es válido y por lo tanto reconocido por el programa.

## 4.4. Traducción de RDF a OWL

El lenguaje OWL está creado para usarse cuando la información contenida en un documento requiera ser procesada por computadoras y también cuando la información sea mostrada a los humanos.

La traducción consiste en convertir el archivo RDF anterior al lenguaje OWL. Como se comentó en la Sección 2.6, el archivo resultante de la traducción será un

archivo en lenguaje OWL DL.

En general, una ontología escrita en OWL está formada por 2 archivos principales:

- *Modelo OWL*. Este archivo define las reglas de construcción que deberá seguir la ontología, establece las propiedades y sus características, clases, relaciones entre clases, cardinalidad, clases enumeradas, tipos de datos, dependencias, las reglas de inferencia y las restricciones de la misma.
- *Archivo de instancias*. Este archivo crea un documento con información final, y se rige por el modelo creado previamente para formar una ontología correcta y congruente con un dominio definido.

### 4.4.1. El modelo OWL

El modelo OWL debe representar todas las reglas establecidas en los documentos XML y RDF antes creados para una ontología de registros. Al igual que con RDF, se utiliza sintaxis XML para definir el modelo. Algunas etiquetas usadas en el lenguaje OWL son propias de RDF, y algunas otras, precedidas por el espacio de nombres de OWL, son propias. El prefijo que representa el espacio de nombres permitirá identificar qué etiqueta corresponde a qué dominio.

La etiqueta XML principal para definir el modelo OWL es la etiqueta `rdf:RDF`, la cual es una etiqueta del modelo de datos RDF. La diferencia radica en que ahora se agrega un espacio de nombres nuevo, el espacio de nombres de OWL: `xmlns:owl="http://www.w3.org/2002/07/owl"`, que se utiliza para identificar los nombres y propiedades del lenguaje OWL. El modelo debe tener un etiqueta `owl:Ontology` para establecer información sobre la ontología que se está definiendo.

El modelo OWL se escribe utilizando principalmente 3 etiquetas:

- `owl:Class`. Define un grupo de cosas que comparten algunas propiedades.
- `owl:ObjectProperty`. Relaciona instancias de dos clases.
- `owl:DatatypeProperty`. Relaciona instancias de clases y litereales RDF además de tipos de datos de un XML-Schema.

Como primer paso, se crea el encabezado del modelo OWL.

**Ejemplo 4.10.**

```
<?xml version="1.0"?>

<rdf:RDF xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:xml="http://www.w3.org/XML/1998/namespace"
 xmlns:or="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml"
```

```
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns="http://localhost/sistor/modelo.owl#"
 xml:base="http://localhost/sistor/modelo.owl#">
<owl:Ontology xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
rdf:about="http://localhost/sistor/modelo.owl">
  <rdfs:comment xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
   Modelo para una Ontologia de Registros en OWL, Version 1
  </rdfs:comment>
</owl:Ontology>

...

<rdf:RDF>
```

En el Ejemplo 4.10 la etiqueta `rdf:RDF` contiene la definición de todos los espacios de nombres que se utilizan para crear el modelo, también muestra la etiqueta `owl:Ontology` que tiene como etiqueta anidada a `rdfs:comment`, que crea un comentario sobre el documento. Es importante señalar que en el lenguaje OWL se pueden utilizar etiquetas de otro lenguaje cuando se necesiten, y es ahí donde los espacios de nombres evitan la ambigüedad.

Lo siguiente a definir son las etiquetas `algorithm` y `cluster`, primero se declara su uso (Ejemplo 4.11) y posteriormente se completa su definición.

**Ejemplo 4.11.**

```
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.x
      html#algorithm"/>
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xh
      tml#cluster"/>
```

Los elementos `label` y `level` que forman parte del elemento `cluster` tienen una definición similar. En el Ejemplo 4.12 se define solo `label`.

**Ejemplo 4.12.**

```
<owl:DatatypeProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     rdf:about="http://www.utm.mx/~sigepr/recursos/ontology
   ofrecords.xhtml#label">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
```

```
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords
     .xhtml#cluster"/>
</owl:DatatypeProperty>
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofreco
     rds.xhtml#cluster">
  <rdfs:subClassOf xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
    <owl:Restriction>
      <owl:onProperty rdf:resource="http://www.utm.mx/~sigepr/recursos/
      ontologyofrecords.xhtml#label"/>
      <owl:minCardinality
        rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeIn
        teger">1</owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

La etiqueta `label` en el Ejemplo 4.12 se define como una propiedad de tipo de dato (`owl:DatatypeProperty`) en la que se establece el rango del elemento, es decir, el tipo de dato que contendrá el elemento (en este caso una cadena), y el dominio en donde estará definido el elemento (elemento `cluster`). Una vez establecido el rango y el dominio del elemento, es necesario indicar que este elemento será una subclase de una clase padre, en este caso `cluster`, también se debe indicar la cardinalidad, la cual determina cuántas instancias del tipo `label` pueden estar contenidas en una instancia de la clase `cluster`.

El siguiente paso es definir un elemento `record`, el cual debe ser una clase que contendrá a las subclases ya conocidas. El Ejemplo 4.13 muestra la definición de una clase `record`.

**Ejemplo 4.13.**

```
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xh
     tml#record"/>
```

Cada elemento dentro de la clase `record` debe ser definido como una propiedad de tipo de datos y despúes como una subclase de `record`, tal y como se hizo en el Ejemplo 4.13, de manera que la plantilla para cada elemento dentro de `record` quedaría tal y como lo muestra el Ejemplo 4.14. Donde `nombre_del_elemento` representa el identificador del elemento a definir (title, subject, description, etc.), `tipo_de_dato_del_elemento` el tipo de los datos que podrá almacenar este elemento y `cardinalidad_minima_del_elemento` el número mínimo de instancias de este elemento que deberán estar contenidas en una instancia de la clase `record`.

**Ejemplo 4.14.**

```
<owl:DatatypeProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xh
     tml#nombre_del_elemento">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
        rdf:resource="http://www.w3.org/2001/XMLSchema#tipo_de_dato_del
        _elemento"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
        rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofreco
        rds.xhtml#record"/>
</owl:DatatypeProperty>
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xh
     tml#record">
  <rdfs:subClassOf xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
    <owl:Restriction>
      <owl:onProperty
        rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofr
        ecords.xhtml#nombre_del_elemento"/>
      <owl:minCardinality
        rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeIn
        teger">
         cardinalidad_minima_del_elemento
      </owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

Una vez definidos los elementos particulares, es necesario establecer dentro de qué elementos están contenidos. El Ejemplo 4.15 muestra la utilidad de las etiquetas `range` y `domain`, indicando el tipo de datos con la etiqueta `range` y el identificador de otro elemento previamente definido (`algorithm`). Posteriormente se establece la clase del elemento recién creado, usando para ello algunas etiquetas ya mencionadas.

**Ejemplo 4.15.**

```
<owl:ObjectProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 rdf:ID="hasontologyofrecords.xhtmlalgorithm">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecor
    ds.xhtml#algorithm"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
        rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofreco
        rds.xhtml#ontologyofrecords"/>
```

```
</owl:ObjectProperty>
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xh
 tml#ontologyofrecords">
  <rdfs:subClassOf xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasontologyofrecords.xhtmlalgorithm"/>
      <owl:minCardinality
        rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeIn
        teger">
           0
      </owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

El proceso del Ejemplo 4.15 se utiliza para incorporar los elementos `cluster` y `record` a las clases correspondientes.

Finalmente se modifica el tipo de datos de algunos atributos usados, y se establece el elemento al que corresponden. El Ejemplo 4.16 presenta la definición de la propiedad `date` e indica que pertenece al elemento `ontologyofrecord`. El resto de los atributos de la definición de una ontología de registros: `clustersupport`, `globalsupport` y `name` se definen de manera análoga.

**Ejemplo 4.16.**

```
<owl:DatatypeProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
rdf:ID="dataPropertydate">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.
   xhtml#ontologyofrecords"/>
</owl:DatatypeProperty>
```

Una vez creado el modelo OWL, es necesario probar que los editores OWL lo reconocen como un documento OWL DL. Para demostrar que el modelo es válido y además cumple con los requisitos de estar escrito en OWL DL, se utilizan los editores Altova SemanticWorks, Protegé y el validador web que recomienda la W3C[3]. Las Figuras 4.6, 4.7 y 4.8 muestran los resultados de cada software al validar el modelo

[3]http://www.mygrid.org.uk/OWL/Validator

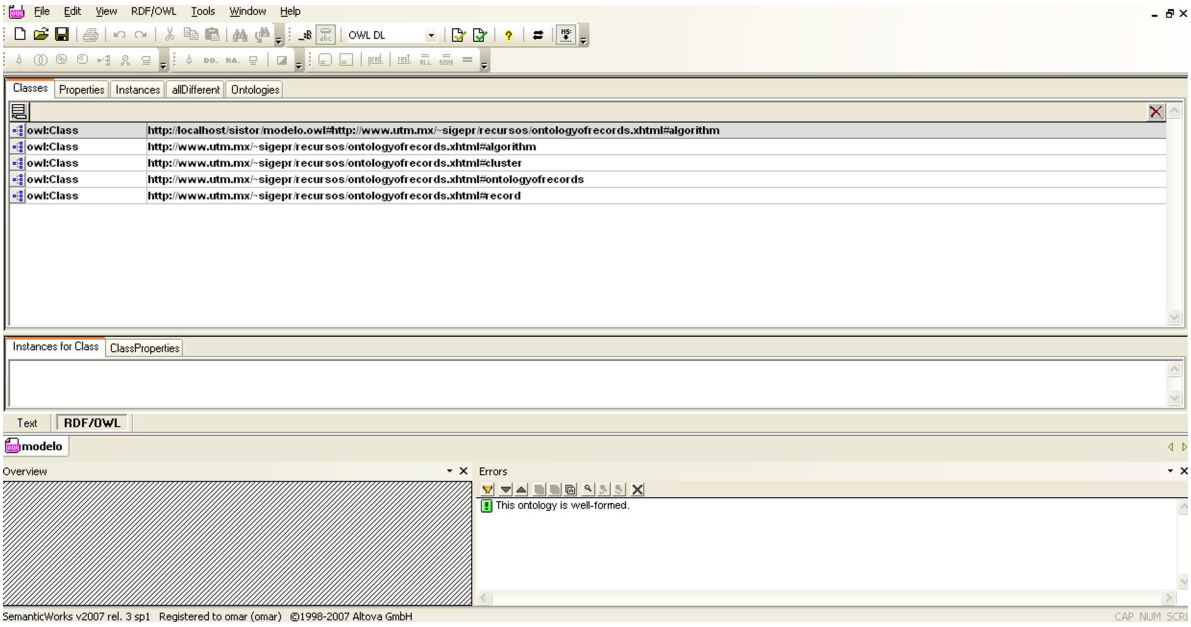


**Figura 4.6.** Cuadro de resultados obtenidos al validar el modelo OWL en el software Altova SemanticWorks.

creado. Con esto se garantiza que el modelo OWL es válido y además reconocido como una ontología escrita en lenguaje OWL DL.

### 4.4.2. El archivo de instancias

Una vez que el modelo se ha validado, se debe crear el archivo OWL instancias, el cual además de contener valores específicos para cada una de los elementos definidos en el modelo, cumple con las reglas establecidas del mismo.

El archivo de instancias OWL de una ontología de registros contiene los datos de una ontología de registros. El Ejemplo 4.17 muestra de forma resumida como sería un archivo de instancias.

**Ejemplo 4.17.**

```
<or:algorithm rdf:ID="idalgorithm">
  <model:dataPropertyclustersupport>30</model:dataPropertyclustersupport>
  <model:dataPropertyglobalsupport>25</model:dataPropertyglobalsupport>
  <model:dataPropertyname>FIHC</model:dataPropertyname>
</or:algorithm>
<or:ontologyofrecords rdf:ID="idontologyofrecords">
  <xsi:schemaLocation>
   http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml
   http://www.utm.mx/~sigepr/recursos/ontologyofrecordsv3a.xsd
  </xsi:schemaLocation>
```

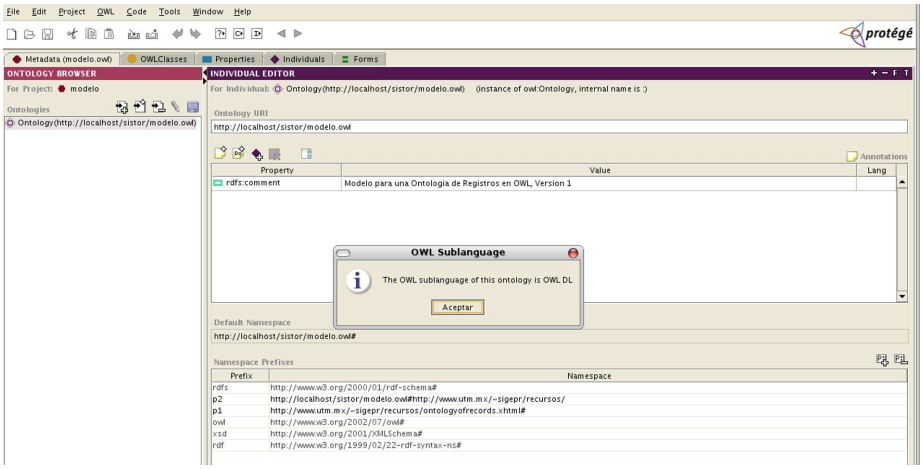


**Figura 4.7.** Cuadro de resultados obtenidos al validar el modelo OWL en el software Protegé.


**Figura 4.8.** Cuadro de resultados obtenidos al validar el modelo OWL en el validador web recomendado por la W3C.

```
    <model:dataPropertydate>2006-04-21</model:dataPropertydate>
  </or:ontologyofrecords>
  <or:cluster rdf:ID="idcluster_1">
    <or:level> 1 </or:level>
    <or:label> fisica </or:label>
    <model:hasrecord rdf:resource="idrecord_2"/>
    <model:hasrecord rdf:resource="idrecord_3"/>
    <model:hasrecord rdf:resource="idrecord_4"/>
  </or:cluster>

  ...

    <or:record rdf:ID="idrecord_2">
    <or:title> Fisica I </or:title>
    <or:subject> Fisica </or:subject>
    <or:description> Abstract o prologo </or:description>
    <or:identifier> oai:libroUTM:1 </or:identifier>
    <or:url> </or:url>
    <or:dataprovider>
     http://cs1.ist.psu.edu/cgi-bin/oai.cgi
    </or:dataprovider>
    <or:tmetadataformat> oai_dc </or:tmetadataformat>
    <or:datestamp> 2006-05-12 </or:datestamp>
  </or:record>
  <or:record rdf:ID="idrecord_3">
    <or:title> Fisica Ondulatoria II </or:title>
    <or:subject> Fisica Ondulatoria </or:subject>
    <or:description> </or:description>
    <or:identifier> oai:libroUTM:2 </or:identifier>
    <or:url> </or:url>
    <or:dataprovider>
     http://cs1.ist.psu.edu/cgi-bin/oai.cgi
    </or:dataprovider>
    <or:tmetadataformat> oai_dc </or:tmetadataformat>
    <or:datestamp> 2006-01-01 </or:datestamp>
  </or:record>
  <or:record rdf:ID="idrecord_4">
    <or:title> Principios de la Fisica Odulatoria </or:title>
    <or:subject> Principios de Fisica Ondulatoria </or:subject>
    <or:description> </or:description>
    <or:identifier> oai:libroUTM:3 </or:identifier>
    <or:url> </or:url>
    <or:dataprovider>
     http://cs1.ist.psu.edu/cgi-bin/oai.cgi
    </or:dataprovider>
    <or:tmetadataformat> oai_dc </or:tmetadataformat>
```

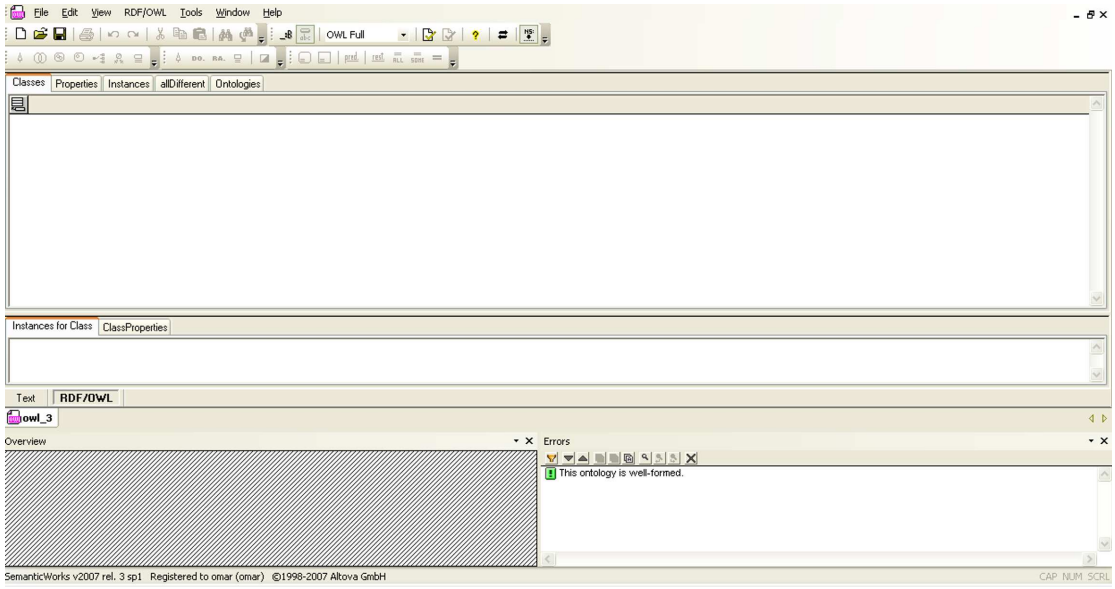


**Figura 4.9.** Archivo de instancias OWL abierto por el editor de ontologías Altova SemanticWorks.

```
    <or:datestamp> 2006-01-01 </or:datestamp>
  </or:record>
```

Los espacios de nombres que se utilizan en el Ejemplo 4.17 deben ser declarados al inicio del archivo, y como se puede apreciar, el atributo `rdf:ID` es muy utilizado para hacer referencia a elementos que se crearán posteriormente, como es el caso de los elementos `record`.

Al igual que con el modelo, el archivo de instancias representa un documento OWL y debe ser validado y reconocido por los editores previamente mencionados, además de permitir, como con otras ontologías creadas nativamente en cada editor, gestionar diferentes características de la ontología usando herramientas propias.

Las Figuras 4.9 y 4.10 muestran las pantallas que se obtienen al abrir el archivo completo de instancias del Ejemplo 4.17 en los editores Altova SemanticWorks y Protegé, con lo que podemos observar que dicho archivo es reconocido por los editores OWL.

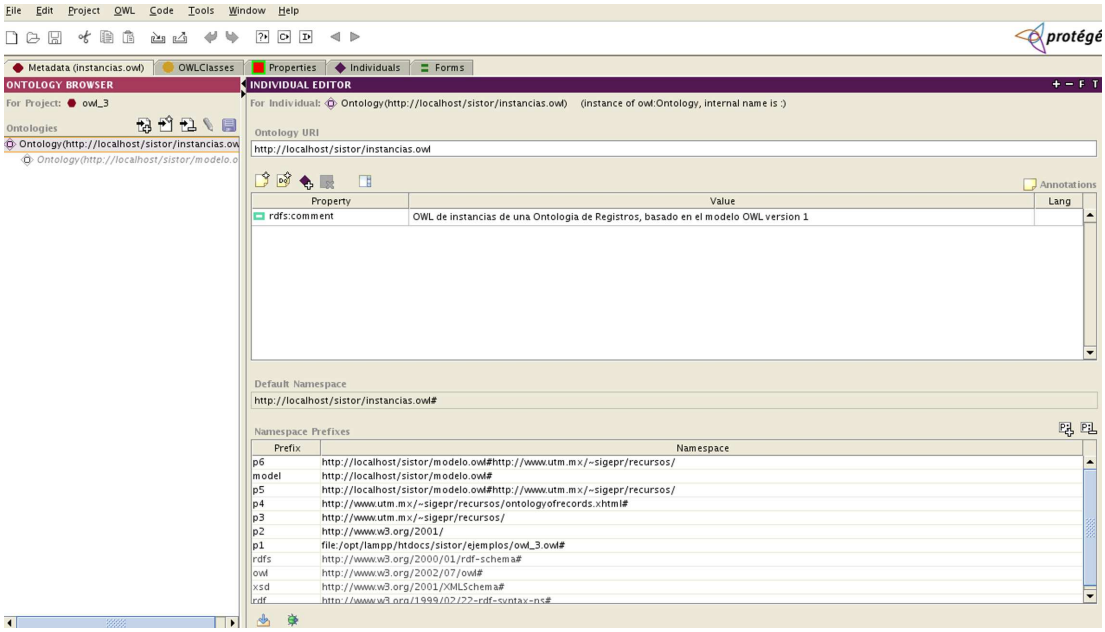


**Figura 4.10.** Archivo de instancias OWL abierto por el editor de ontologías Protegé.

# Capítulo 5

# Implementación

Sólo hay esperanza en la acción.

*J. P. Sartre*

## 5.1. El sistema de traducción de ontologías de registros

Partiendo del requisito de que el sistema de revisión, nombrado: Sistema de Traducción de Ontologías de Registros, debe permitir la construcción coordinada de ontologías de registros, se resaltan las características iniciales siguientes:

- Debe ser accesible de forma remota.
- Manejar tipos de usuarios y grupos coordinados.
- Implementar el protocolo Co4 para el control coordinado de las modificaciones y traducciones de la ontología.
- Contener un editor para implementar la operación de fusión.

Para cumplir con el requerimiento de accesibilidad remota, el sistema se desarrolla como aplicación web. Para desarrollar una aplicación web es necesario utilizar un lenguaje de programación aplicable al entorno web. Para mantener los datos sobre el sistema y permitir el acceso vía web, se emplea un manejador de base de datos. El lenguaje de programación seleccionado es PHP y el manejador de base de datos MySQL, ambos son software libre y no representa ningún problema legal su uso para el desarrollo de aplicaciones. Realizar una aplicación en web requiere de un servidor web. Para ello se seleccionó Apache Web Server, el cual es compatible con PHP y MySQL.


**Figura 5.1.** Pantalla de inicio del Sistema de traducción de ontologías de registros.

### 5.1.1. Implementación de Co4

De acuerdo al protocolo Co4, la revisión debe ser coordinada por una persona que representa una guía en la traducción y modificación. Esta persona indica a las demás **cuándo** y **qué** tareas realizar. Por tal motivo, la implementación del sistema requiere de usuarios con diferentes privilegios.

El sistema será controlado por dos tipos de usuarios:

- *Administrador*. Este tipo de usuario podrá dar de alta a otros usuarios no administradores (participantes) con el fin de generar un grupo coordinado, además de ser el único encargado en dirigir el proceso de revisión y traducción.

- *Participante*. Este tipo de usuario deberá atender las peticiones del administrador de efectuar revisiones y/o modificaciones de algún documento.

Un grupo coordinado está formado por un administrador y un conjunto de participantes (expertos), los cuales revisan una ontología de registros para generar una representación de la misma en RDF u OWL.

Cada uno de los usuarios del sistema debe acceder mediante un nombre de usuario y una contraseña que se le solicita ingrese al registrarse en el sistema, la Figura 5.1 muestra la captura de la pantalla de acceso al sistema.

Un grupo coordinado se crea al mismo tiempo que un administrador se registra en el sistema, de esta forma se establece que sólo un administrador puede dirigir un grupo coordinado específico. El administrador es el encargado de registrar a los participantes, y sólo el administrador puede registrar participantes para el grupo coordinado que dirige. La Figura 5.2 muestra el cuadro de diálogo para el registro de un administrador y de su grupo coordinado.

Cada tipo de usuario tiene permitido realizar un grupo de tareas de acuerdo al rol con el que participa en el sistema, la Tabla 5.1 muestra cada una de las tareas permitidas para cada tipo de usuario.

El entorno de trabajo para cada tipo de usuario en el sistema lo muestra la Figura 5.3, donde sólo se aprecia algunas de las opciones de la Tabla 5.1, el resto son accesibles sólo por medio de la ejecución de otras.

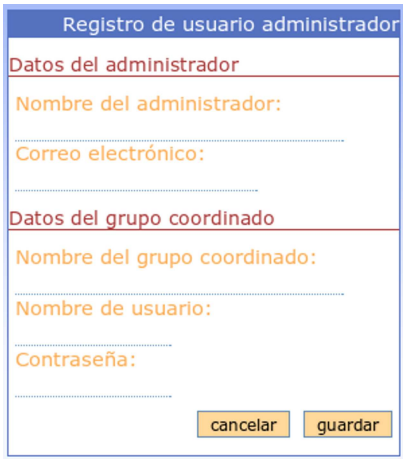


**Figura 5.2.** Cuadro de diálogo para registrar a un administrador y un grupo coordinado.

| **Tarea** | **Usuario administrador** | **Usuario participante** |
|---|---|---|
| Crear usuarios (participantes) | si | no |
| Crear notificaciones de acuerdo | si | no |
| Crear notificaciones de modificación | si | no |
| Eliminar notificaciones | si | no |
| Ver notificaciones | si | si |
| Responder notificaciones | no | si |
| Editar ontología XML | no | si |
| Cargar la ontología de registros inicial | si | no |
| Establecer las modificaciones de la ontología XML | si | no |
| Traducir ontologías | si | no |
| Acceder a los archivos de referencia | si | si |
| Ver los archivos del grupo coordinado | si | si |
| Ver los participantes del grupo coordinado | si | no |
| Accesar a la ayuda | si | si |

**Tabla 5.1.** Tareas permitidas para cada tipo de usuario en el sistema

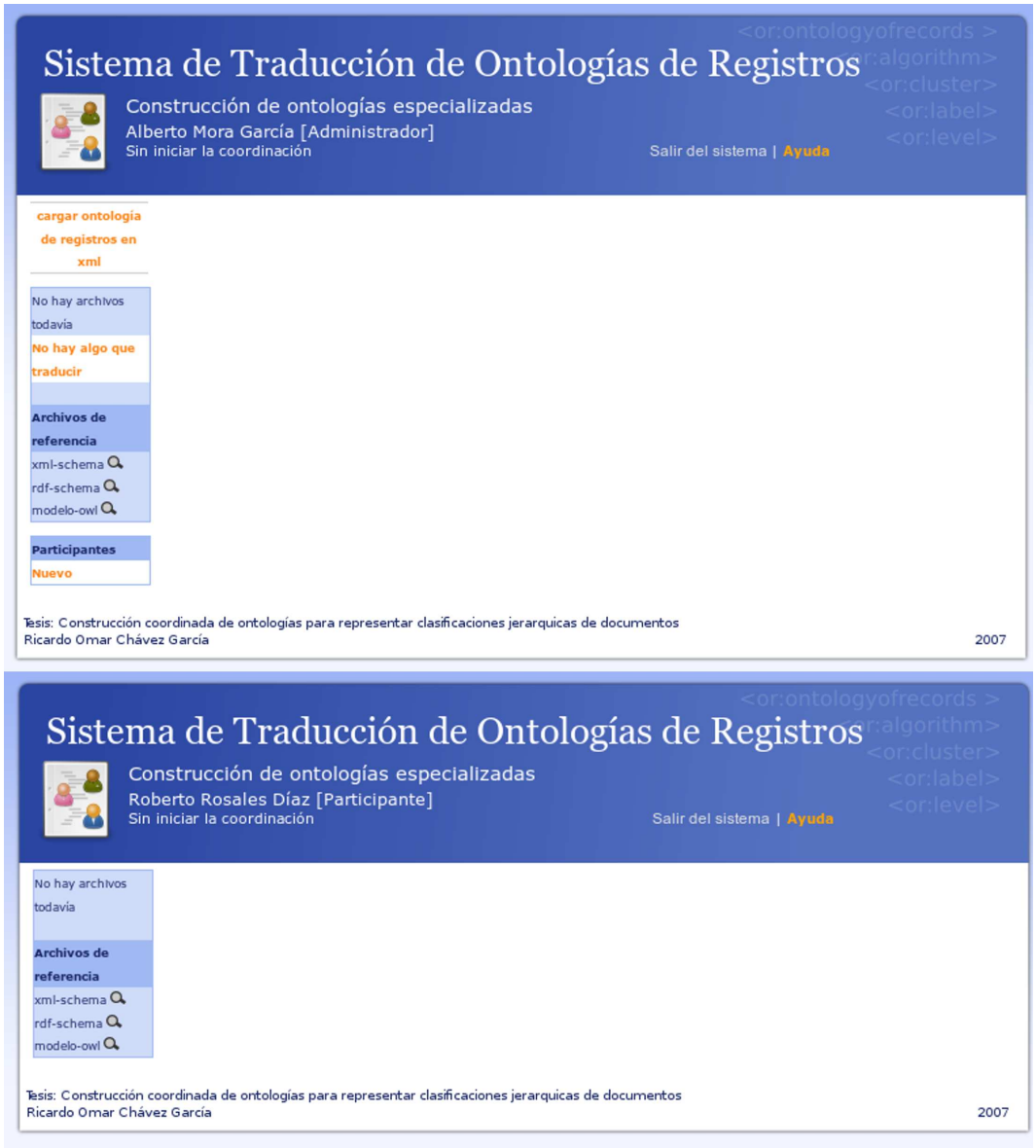


**Figura 5.3.** Entornos de trabajo dentro del sistema para un usuario administrador y un usuario participante, respectivamente.

### 5.1.2. El proceso de revisión

El sistema implementa el protocolo Co4 junto con el proceso de revisión y fusión. Co4 basa su control en el envío y recepción de notificaciones. La aplicación del protocolo en esta tesis considera lo siguiente:

- Un documento que representa una ontología de registros se considera una base de conocimientos.
- Existen dos tipos de peticiones:
  - *Peticiones de acuerdo*. Se emitirán del administrador a cada participante, solicitando revise un documento específico y responda si el documento le parece correcto o incorrecto.
  - *Peticiones de modificación*. Se emitirán del administrador a cada participante, solicitando revise un documento específico para verificar que sea correcto y en caso contrario lo modifique según lo considere.
- Las peticiones de acuerdo sirven para verificar que todos los participantes coincidan en que el documento en cuestión representa correctamente el dominio particular.
- Las peticiones de modificación se emiten cuando se necesita aplicar la operación de fusión al documento XML. El documento XML se modifica sólo si las notificaciones de acuerdo se aceptan.
- Se traduce un documento si todos los participantes del grupo coordinado aprueban la notificación de acuerdo que acepta el documento como correcto.
- Cada nivel de traducción representa una nueva base de conocimientos.

La Figura 5.4 muestra gráficamente la implementación del protocolo Co4 en el sistema, el pseudocódigo de la implementación es el siguiente:

**Caso 1**. Revisar si el documento es correcto

1 Para cada participante:
  1.1 El administrador emite una notificación de acuerdo (NA)
  1.2 El participante recibe la notificación y revisa el documento
  1.3 En caso de que responda (RNA) que el documento no es correcto, entonces se rechaza el documento
2 En caso de que todos los participantes respondan que el documento es correcto, entonces se acepta el documento

**Caso 2**. Modificación de un documento

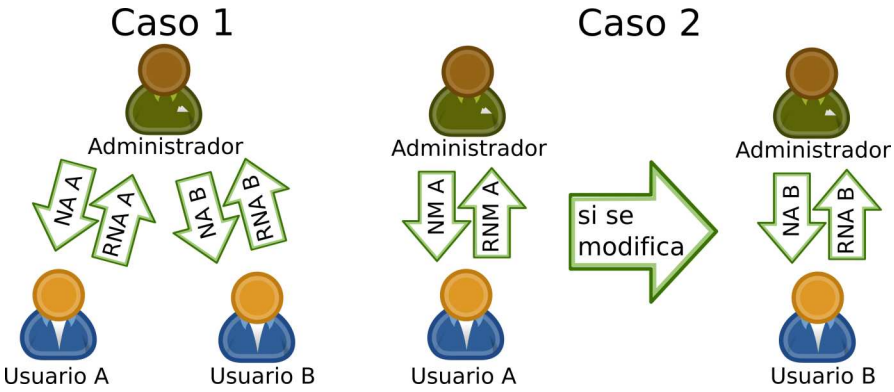


**Figura 5.4.** Ejemplo de la implementación del protocolo Co4, con el uso de notificaciones de acuerdo(NA) y modificación (NM) en el sistema.

1 Para cada participante:

1.1 El administrador emite una notificación de modificación (NM)

1.2 El participante recibe la notificación y revisa el documento

1.3 En caso de que considere que el documento necesita modificación, efectua la modificación y responde (RNM) que ha modificado el documento

1.3.1 Se realizan los pasos del Caso 1 para determinar si el documento modificado es correcto

1.4 En caso de que se considere que no necesita modificación el documento, responde que el documento es correcto

### 5.1.3. El proceso de traducción

La traducción es una de las partes fundamentales que cumple el sistema, y consta de transformar un documento XML inicial (ontología inicial) que cumple con el DTD y XML-Schema que se establecieron para validar documentos que representen ontologías de registros, en una ontología escrita en lenguaje RDF y respaldada por el RDF-Schema correspondiente o bien, en una ontología escrita en lenguaje OWL y validada por el modelo OWL establecido.

El proceso de traducción de las ontologías en el sistema se efectua mediante los siguientes pasos:

1. Cargar el documento XML que represente una ontología de registros en XML y que esté validado por el DTD y el XML-Schema establecido.

2. Verificar mediante el proceso coordinado (implementado por el protocolo Co4) que el XML inicial sea correcto.

3. Después de llegar a un acuerdo, establecer el XML final, el cual servirá como partida para la traducción.

4. Verificar mediante el proceso coordinado que la ontología escrita en XML (XML final) es apta para su traducción al lenguaje RDF.

5. Traducir la ontología en XML a una ontología escrita en lenguaje RDF.

6. Verificar mediante el proceso coordinado que la ontología escrita en RDF es apta para su traducción al lenguaje OWL.

7. Traducir la ontología en RDF a una ontología escrita en lenguaje OWL.

8. Verificar mediante el proceso coordinado que la ontología escrita en OWL es correcta.

Los pasos que realiza la traducción de XML a RDF y de RDF a OWL se justifican en la Sección 4.3 y 4.4, donde ya se analizaron y determinaron la forma de cada uno de los nuevos elementos de la ontología, ya sea en RDF o en OWL.

En general, el algoritmo para realizar la traducción de la ontología en lenguaje A a una ontología en lenguje B es el siguiente:

0. Inicio

1. Lectura del documento que representa la ontología en lenguaje A

2. Escritura de la estructura básica de la ontología en lenguaje B

3. Extracción de datos característicos de la ontología en lenguaje A, como son: la fecha y los datos del algoritmo utilizado

4. Escribir los datos característicos de la ontología en lenguaje A en la ontología en lenguaje B

5. Para cada cluster en la ontología en lenguaje A hacer:

   *a*) Extraer la información característica del cluster, como son: el nivel y las etiquetas

   *b*) Escribir en un cluster temporal en lenguaje B la información característica extraída del cluster actual

   *c*) Para cada registro en el cluster actual:

      1) Extraer la información característica del registro, como son: el título, tema descripción, identificador, url, proveedor de datos, formato de los metadatos y la estampa de tiempo

      2) Escribir dentro del cluster temporal la información característica extraída del registro actual


**Figura 5.5.** Evolución de la sección de archivos, según el nivel del proceso de revisión y traducción.

*d*) Escribir el cluster temporal en el lenguaje B en la ontología en lenguaje B

6. Fin

El sistema maneja una sección de archivos en la cual sólo el administrador puede efectuar modificaciones. La sección de archivos evoluciona según lo avanzado del proceso de revisión y traducción tal y como lo muestra la Figura 5.5, donde se pueden ver los diferentes estados de la sección de archivos, a la cual se le agrega un archivo más al ingresar un XML inicial, al fijar el XML final, al realizar la traducción de XML a RDF y al realizar la traducción de RDF a OWL. La sección de archivos también es visible por los participantes, sólo que ellos no pueden realizar modificaciones a la sección, la pueden utilizar solamente para verificar el contenido de los documentos.

El proceso de traducción considera que cada documento es válido y bien formado, para ello pone a la disposición de los usuarios el conjunto de archivos de referencia sobre las ontologías, los cuales son:

- DTD y XML-Schema para validar una ontología de registros en XML.
- RDF-Schema para construir un RDF que representa una ontología de registros.
- Modelo OWL para validar un archivo OWL de instancias.

De tal manera que si un documento A es válido (cumple con las referencias respectivas) y bien formado, se asegura que la ontología B, resultado de la traducción, también lo es.

Si en alguna etapa del proceso no se llega a un acuerdo en cuanto a lo correcto de la ontología en revisión, el proceso de traducción se detiene y se da por finalizado.

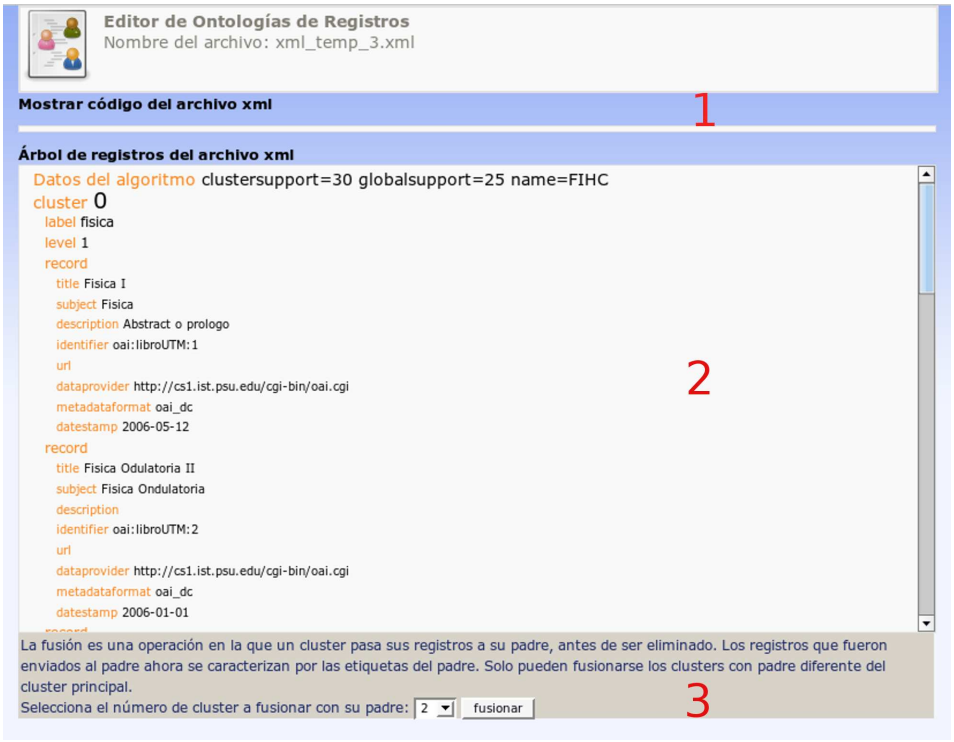


**Figura 5.6.** Interfaz del editor de ontologías de registros del Sistema de Traducción de Ontologías de Registros.

### 5.1.4. El editor de ontologías de registros

La modificación de una ontología de registros en XML se realiza después de una notificación de modificación. Sólo de esta manera es posible acceder al editor. Al recibir una notificación de modificación, automáticamente el sistema envía un acceso al editor de ontologías, el cual además de mostrar la ontología (en forma de código y en forma de árbol de grupos y registros) a modificar, también permite realizar la operación de fusión, la cual cumple con las reglas establecidas en la Sección 2.2. La Figura 5.6 muestra la interfaz del editor XML, de la cual resaltan 3 partes:

1. Área del código XML de la ontología de registros.
2. Área del árbol de clusters y registros de la ontología de registros.
3. Área de aplicación de la operación fusión.

El área de aplicación de la operación fusión establece los requisitos para realizar dicha operación, además de restringir los clusters posibles a fusionar mediante una lista de posibilidades. El editor de ontologías muestra sólo los grupos que tienen un nivel mayor a 0.

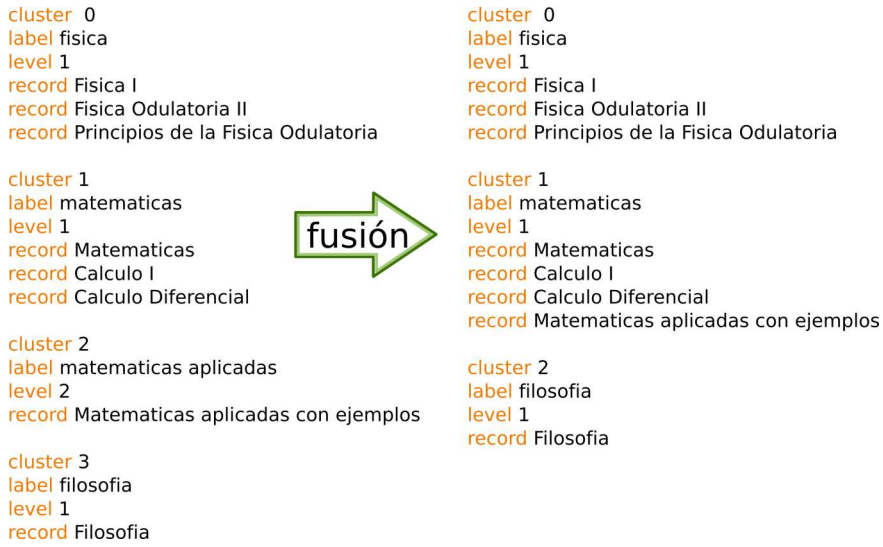


**Figura 5.7.** Ejemplo de la operación fusión en el editor de ontologías.

Un ejemplo de la operación fusión en el editor se presenta en la Figura 5.7. En un estado inicial se tienen dos grupos, el cluster 1 y su hijo el cluster 2, con sus respectivos registros. Al elegir el cluster 2 para fusionar con su cluster padre se obtiene un nuevo grupo, el cluster 1, que ahora contiene los registros del cluster 2. El cluster 2 se elimina y la numeración de los clusters siguientes se modifica.

# Capítulo 6

# Conclusiones

---

País sordo, ciudad quemada. La hoguera nos llama, hoy por hoy,
no habrá final feliz.

*PIT II*

No es cierto.

*Yo*

Para concluir esta tesis, esta sección resume los aspectos principales y propone a su vez trabajo futuro.

En relación al lenguaje XML, con el cual se representaron inicialmente las ontologías de registros [Med05], se concluye que es apto y suficiente para representar estructuras jerárquicas que no requieran de una interpretación única. Sin embargo, XML no permite representar la semántica de la estructura de sus documentos. Como resultado, es difícil contar con mecanismos que permitan a los humanos o a las máquinas hacer deducciones correctas sobre el contenido o la estructura. De ahí que surja la necesidad de traducir las ontologías de registros a otros lenguajes más expresivos.

Conforme se analizan las características de XML, RDF y OWL, se observa que la expresividad es mayor en el último. Sin embargo, a medida que se agrega expresividad, la legibilidad de parte de los usuarios humanos se ve disminuida. Por ello, es conveniente el uso de editores y herramientas que apoyan la interpretación y faciliten el manejo de los documentos en general y de las ontologías de registros en particular.

En la tesis se demostró la confirmación de la hipótesis, es decir, que fue posible la construcción sin pérdida de generalidad de las ontologías de registros en RDF y OWL, y que se puede acceder a éstas a través de diferentes editores. Esto de forma implícita implica que fue necesario el desarrollo de plantillas, esquemas y espacios de nombres para generar ontologías de registros válidas.

En este sentido, es importante precisar que el empleo de un solo editor no hubiera sido suficiente para mostrar la validez de los documentos generados, porque

a la fecha, no existe un lenguaje único ni herramientas estándares para manejar ontologías.

Por otro lado, a través del sistema de traducción de ontologías de registros, en la tesis se implementó de forma exitosa el protocolo Co4, el cual apoya la evaluación de las ontologías construidas de acuerdo a los criterios de un grupo coordinado de participantes. Esto promueve su reutilización y aplicación a otros dominios de interés.

Las ontologías traducidas pueden incorporar otras características y aplicarse en otros dominios. A manera de trabajo a futuro, se propone que se utilicen en sistemas de inferencia para extraer información implícita y para garantizar formalmente la consistencia de las mismas. También es posible implementar un proceso que automatice la fusión de grupos y permita que las ontologás resultantes sirvan de entrada al proceso de revisión.

Otra tendencia es la posible extensión de las ontologías a partir de la combinación de ontologías previas mediante la aplicación de los mecanismos de traducción propuestos en esta tesis.

# Apéndice A

# Archivos de referencia

---

Este apéndice muestra los archivos de referencia creados y utilizados para el proceso de traducción a lo largo del trabajo de tesis.

## A.1. El archivo DTD

El Código A.1 muestra el archivo DTD que se utiliza para validar a un xml que representa una ontología de registros.

**Código A.1.**

```
<!ELEMENT ontologyofrecords (algorithm, cluster+)>
<!ATTLIST ontologyofrecords date CDATA #REQUIRED >
<!ELEMENT algorithm EMPTY>
<!ATTLIST algorithm name CDATA #FIXED "FIHC" globalsupport
CDATA #REQUIRED clustersupport CDATA #REQUIRED >
<!ELEMENT cluster (label, level, record*)>
<!ELEMENT label (#PCDATA)> <!ELEMENT level (#PCDATA)>
<!ELEMENT record (title, subject?, description?, identifier,
url, dataprovider, metadataformat, datestamp)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT subject (#PCDATA)>
<!ELEMENT description (#PCDATA)>
<!ELEMENT identifier (#PCDATA)>
<!ELEMENT url (#PCDATA)>
<!ELEMENT dataprovider (#PCDATA)>
<!ELEMENT metadataformat (#PCDATA)>
<!ELEMENT datestamp (#PCDATA)>
<!ENTITY generatedBy "OntoSIR Version 1.0">
```

## A.2. El archivo XML-Schema

El Código A.2 muestra el archivo XML-Schema que se utiliza para validar aun xml que presenta una ontología de registros.

**Código A.2.**

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<xs:schema
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml"
targetNamespace="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml"
elementFormDefault="qualified">
<xs:element name="ontologyofrecords">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="algorithm"/>
      <xs:element ref="cluster" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attributeGroup ref="attlist.ontologyofrecords"/>
    </xs:complexType>
```

```
</xs:element>
<xs:attributeGroup name="attlist.ontologyofrecords">
  <xs:attribute name="date" type="xs:date" use="required"/>
</xs:attributeGroup>
<xs:element name="algorithm">
  <xs:complexType>
    <xs:attributeGroup ref="attlist.algorithm"/>
  </xs:complexType>
</xs:element>
<xs:attributeGroup name="attlist.algorithm">
  <xs:attribute name="name" default="FIHC">
    <xs:simpleType>
      <xs:restriction base="xs:string">
        <xs:enumeration value="FIHC"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
 <xs:attribute name="globalsupport" use="required"/>
 <xs:attribute name="clustersupport" use="required"/>
</xs:attributeGroup>
<xs:element name="cluster">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="label"/>
      <xs:element ref="level"/>
      <xs:element ref="record" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="label" type="xs:string"/>
<xs:element name="level" type="xs:integer"/>
<xs:element name="record">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="title"/>
      <xs:element ref="subject" minOccurs="0"/>
      <xs:element ref="description" minOccurs="0"/>
      <xs:element ref="identifier"/>
      <xs:element ref="url"/>
      <xs:element ref="dataprovider"/>
      <xs:element ref="metadataformat"/>
      <xs:element ref="datestamp"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
<xs:element name="title" type="xs:string"/>
<xs:element name="subject" type="xs:string"/>
<xs:element name="description" type="xs:string"/>
<xs:element name="identifier" type="xs:string"/>
<xs:element name="url" type="xs:anyURI"/>
<xs:element name="dataprovider" type="xs:string"/>
<xs:element name="metadataformat" type="xs:string"/>
<xs:element name="datestamp" type="xs:date"/>
</xs:schema>
```

## A.3. El archivo RDF-Schema

El Código A.3 es el RDF-Schema utilizado como vocabulario para la generar el archivo RDF.

**Código A.3.**

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">
]>
<rdf:RDF
      xml:base="http://www.utm.mx/~sigepr/recursos/ontologyofrecords/ontologyofrecords"
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

      <rdfs:Class rdf:ID="OntologyOfRecords"/>

<rdfs:Datatype rdf:about="&xsd;date"/>
<rdfs:Datatype rdf:about="&xsd;string"/>
<rdfs:Datatype rdf:about="&xsd;integer"/>
<rdfs:Datatype rdf:about="&xsd;anyURI"/>

<rdfs:Property rdf:ID="isDescribedByDate">
<rdfs:domain rdf:resource="#OntologyOfRecords"/>
<rdfs:range rdf:resource="&xsd;date"/>
</rdfs:Property>
```

```
<rdfs:Class rdf:ID="Calgorithm"/>
<rdfs:Property rdf:ID="isDescribedByClusterSupport">
<rdfs:domain rdf:resource="#Calgorithm"/>
<rdfs:range rdf:resource="&xsd;integer"/>
</rdfs:Property>

<rdfs:Property rdf:ID="isDescribedByGlobalSupport">
<rdfs:domain rdf:resource="#Calgorithm"/>
<rdfs:range rdf:resource="&xsd;integer"/>
</rdfs:Property>

<rdfs:Property rdf:ID="isDescribedByName">
<rdfs:domain rdf:resource="#Calgorithm"/>
<rdfs:range rdf:resource="&xsd;string"/>
</rdfs:Property>

<rdfs:Property rdf:ID="hasAlgorithm">
<rdfs:domain rdf:resource="#OntologyOfRecords"/>
<rdfs:range rdf:resource="#Calgorithm"/>
</rdfs:Property>

<rdfs:Class rdf:ID="Ccluster" />

<rdfs:Property rdf:ID="hasCluster">
<rdfs:domain rdf:resource="#OntologyOfRecords"/>
<rdfs:range rdf:resource="#Ccluster"/>
</rdfs:Property>

<rdfs:Class rdf:ID="Clabel"/>

<rdfs:Property rdf:ID="hasLabel">
<rdfs:domain rdf:resource="#Ccluster"/>
<rdfs:range rdf:resource="&xsd;string"/>
</rdfs:Property>

<rdfs:Class rdf:ID="Clevel"/>

<rdfs:Property rdf:ID="hasLevel">
<rdfs:domain rdf:resource="#Ccluster"/>
<rdfs:range rdf:resource="&xsd;integer"/>
</rdfs:Property>

<rdfs:Class rdf:ID="Crecord"/>

<rdfs:Property rdf:ID="hasRecord">
<rdfs:domain rdf:resource="#Ccluster"/>
<rdfs:range rdf:resource="#Crecord"/>
</rdfs:Property>

<rdfs:Class rdf:ID="Ctitle"/>

<rdfs:Property rdf:ID="hasTitle">
<rdfs:domain rdf:resource="#Crecord"/>
<rdfs:range rdf:resource="&xsd;string"/>
</rdfs:Property>

<rdfs:Class rdf:ID="Csubject"/>

<rdfs:Property rdf:ID="hasSubject">
<rdfs:domain rdf:resource="#Crecord"/>
<rdfs:range rdf:resource="&xsd;string"/>
</rdfs:Property>

<rdfs:Class rdf:ID="Cdescription"/>

<rdfs:Property rdf:ID="hasDescription">
<rdfs:domain rdf:resource="#Crecord"/>
<rdfs:range rdf:resource="&xsd;string"/>
</rdfs:Property>

<rdfs:Class rdf:ID="Cindentifier"/>

<rdfs:Property rdf:ID="hasIdentifier">
<rdfs:domain rdf:resource="#Crecord"/>
<rdfs:range rdf:resource="&xsd;string"/>
</rdfs:Property>

<rdfs:Class rdf:ID="Curl"/>

<rdfs:Property rdf:ID="hasUrl">
<rdfs:domain rdf:resource="#Crecord"/>
<rdfs:range rdf:resource="&xsd;anyURI"/>
</rdfs:Property>

<rdfs:Class rdf:ID="Cdataprovider"/>

<rdfs:Property rdf:ID="hasDataProvider">
<rdfs:domain rdf:resource="#Crecord"/>
```

```
<rdfs:range rdf:resource="&xsd;string"/>
</rdfs:Property>

<rdfs:Class rdf:ID="Cmetadataformat"/>

<rdfs:Property rdf:ID="hasMetadataFormat">
<rdfs:domain rdf:resource="#Crecord"/>
<rdfs:range rdf:resource="&xsd;string"/>
</rdfs:Property>

<rdfs:Class rdf:ID="Cdatestamp"/>

<rdfs:Property rdf:ID="hasDatestamp">
<rdfs:domain rdf:resource="#Crecord"/>
<rdfs:range rdf:resource="&xsd;date"/>
</rdfs:Property>
</rdf:RDF>
```

# A.4. El modelo OWL

El modelo representado en el Código A.4 es el archivo utilizado como modelo para la creación de archivos OWL de instancias.

**Código A.4.**

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [<!ENTITY xml "http://www.w3.org/XML/1998/namespace">
<!ENTITY or "http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml">
<!ENTITY xsi "http://www.w3.org/2001/XMLSchema-instance">
<!ENTITY xsd "http://www.w3.org/2001/XMLSchema">]>

<rdf:RDF xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:xml="http://www.w3.org/XML/1998/namespace"
 xmlns:or="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
 xmlns="http://localhost/sistor/modelo.owl#" xml:base="http://localhost/sistor/modelo.owl#">
<owl:Ontology xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" rdf:about="http://localhost/sistor/modelo.owl">
  <rdfs:comment xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">Modelo para una Ontologia de Registros en OWL, Version 1</rdfs:comment>
</owl:Ontology>
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#algorithm"/>
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#cluster"/>
<owl:DatatypeProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#label">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#cluster"/>
</owl:DatatypeProperty>
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#cluster">
  <rdfs:subClassOf xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
    <owl:Restriction>
      <owl:onProperty rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#label"/>
      <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:DatatypeProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#level">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#cluster"/>
</owl:DatatypeProperty>
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#cluster">
  <rdfs:subClassOf xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
    <owl:Restriction>
      <owl:onProperty rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#level"/>
      <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">1</owl:minCardinality>
```

```
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#record"/>
<owl:DatatypeProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#title">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#record"/>
</owl:DatatypeProperty>
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#record">
  <rdfs:subClassOf xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
    <owl:Restriction>
      <owl:onProperty rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#title"/>
      <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">0</owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:DatatypeProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#subject">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#record"/>
</owl:DatatypeProperty>
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#record">
  <rdfs:subClassOf xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
    <owl:Restriction>
      <owl:onProperty rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#subject"/>
      <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">0</owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:DatatypeProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#description">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#record"/>
</owl:DatatypeProperty>
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#" xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#record">
  <rdfs:subClassOf xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
    <owl:Restriction>
      <owl:onProperty rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#description"/>
      <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">0</owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:DatatypeProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#identifier">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#record"/>
</owl:DatatypeProperty>
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#record">
  <rdfs:subClassOf xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
    <owl:Restriction>
      <owl:onProperty rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#identifier"/>
      <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">0</owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:DatatypeProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#url">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#record"/>
</owl:DatatypeProperty>
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#record">
  <rdfs:subClassOf xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
```

```
    <owl:Restriction>
      <owl:onProperty rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#url"/>
      <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">0</owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:DatatypeProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#dataprovider">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#record"/>
</owl:DatatypeProperty>
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#record">
  <rdfs:subClassOf xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
    <owl:Restriction>
      <owl:onProperty rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#dataprovider"/>
      <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">0</owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:DatatypeProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#metadataformat">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#record"/>
</owl:DatatypeProperty>
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#" xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#record">
  <rdfs:subClassOf xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
    <owl:Restriction>
      <owl:onProperty rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#metadataformat"/>
      <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">0</owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:DatatypeProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#datestamp">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   rdf:resource="http://www.w3.org/2001/XMLSchema#date"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#record"/>
</owl:DatatypeProperty>
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#record">
  <rdfs:subClassOf xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
    <owl:Restriction>
      <owl:onProperty rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#datestamp"/>
      <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">0</owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:ObjectProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
rdf:ID="hasontologyofrecords.xhtmlalgorithm">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#algorithm"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#ontologyofrecords"/>
</owl:ObjectProperty>
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#ontologyofrecords">
  <rdfs:subClassOf xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasontologyofrecords.xhtmlalgorithm"/>
      <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">0</owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:ObjectProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
rdf:ID="hasontologyofrecords.xhtmlcluster">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#cluster"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#ontologyofrecords"/>
</owl:ObjectProperty>
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#"
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#ontologyofrecords">
  <rdfs:subClassOf xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
```

```
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasontologyofrecords.xhtmlcluster"/>
      <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">0</owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:ObjectProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
rdf:ID="hasontologyofrecords.xhtmlrecord">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#record"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#cluster"/>
</owl:ObjectProperty>
<owl:Class xmlns:owl="http://www.w3.org/2002/07/owl#" xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 rdf:about="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#cluster">
  <rdfs:subClassOf xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasontologyofrecords.xhtmlrecord"/>
      <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#nonNegativeInteger">0</owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:DatatypeProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
rdf:ID="dataPropertydate">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#ontologyofrecords"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
rdf:ID="dataPropertyclustersupport">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#algorithm"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  rdf:ID="dataPropertyglobalsupport">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#algorithm"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
rdf:ID="dataPropertyname">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   rdf:resource="http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtml#algorithm"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty xmlns:owl="http://www.w3.org/2002/07/owl#"
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
rdf:ID="dataPropertyContent">
  <rdfs:range xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
    <owl:Class>
      <owl:unionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#http://www.utm.mx/~sigepr/recursos/ontologyofrecords.xhtmlalgorithm"/>
      </owl:unionOf>
    </owl:Class>
  </rdfs:domain>
</owl:DatatypeProperty>
</rdf:RDF>
```

# Bibliografía


[Ant04] G. Antoniou. *A Semantic Web Primer*. MIT Press, 2004.

[DM98] D-Lib Magazine. An Introduction to the Resource Description Framework, 1998. http://www.dlib.org/dlib/may98/miller/05miller.html.

[Euz95] J. Euzenat. Building consensual knowledge bases: context and architecture. *2nd International Conference on Building and Sharing Very Large-Scale Knowledge Bases (KBKS), Enschede (NL)*, pages 43–155, 1995.

[Far05] Gowadia V. Jain A. Roy D. Farkas, C. From XML to RDF: Syntax, Semantics, Security and Integrity. In *Joint Working Conference on Security Management, Integrity, and Internal Control in Information Systems*, 2005.

[Fra93] R. Françfois. Building and sharing large knowledge bases in molecular genetics. *1st international conference on building and sharing very large-scale knowledge bases (KBKS)*, pages 291–301, 1993.

[Gra02] M. Graves. *Designing XML Databases*. Prentice Hall, 2002.

[Gru93] T. Gruber. A Translation Approach to Portable Ontology Specifications. *Knowledge Acquisition*, 1993.

[Med05] Sánchez J. A. Brisaboa N. Medina, M. A. OntoSIR: An OAI Service for Multi-Collection Document Retrieval Based on Ontologies of Metadata Records. In *Latin American Web Congress (LA-WEB)*. IEEE Computer Society Press, 2005.

[Med06] Chávez-Aragón A. Chávez R. O. Medina, M. A. Construction, Implementation and Maintenance of Ontologies of Records. In *4th Latin American Web Congress (LA-WEB 06, Puebla, México)*, 2006.

[Med07] Sánchez-J. A. Chávez R. O. Medina, M. A. An RDF-based model for encoding document hierarchies. In *17th International Conference on Electronics, Communications and Computers (Conielecomp 2007, Puebla, México)*, 2007.

[VdS02] Lagoze C. Van de Sompel, H. Notes From the Interoperability Front: A Progress Report on the Open Archives Initiative. In *6th European Conference on Research and Advanced Technology for Digital Libraries*, pages 144–157, 2002.

[W3C98] W3C. Uniform Resource Identifiers (URI): Generic syntax, 1998. http://www.ietf.org/rfc/rfc2396.txt.

[W3C04] W3C. OWL Web Ontology Language, 2004. http://www.w3.org/TR/owl-features/.

[W3C06a] W3C. Extensible Markup Language XML 1.0 (Fourth Edition), 2006. http://www.w3.org/TR/REC-xml/.

[W3C06b] W3C. Namespaces in XML 1.0, 2006. http://www.w3.org/TR/REC-xml-names/#sec-namespaces.

[W3C06c] W3C. Resource Description Framework, 2006. http://www.w3.org/RDF/#specs.

[W3S06a] W3Schools. DTD Tutorial, 2006. http://www.w3schools.com/dtd/default.asp.

[W3S06b] W3Schools. XML Schema Tutorial, 2006. http://www.w3schools.com/schema/default.asp.