bid textos universitaris de biblioteconomia i documentació
ISSN 1575-5886
DL B-19.675-1998
número 13
desembre 2004
Facultat de Biblioteconomia i Do
Universitat de Barcelona

Automatización de tesauros y su utilización en la web semántica

[Versió catalana]

JOSÉ RAMÓN PÉREZ AGÜERA
Departamento de Biblioteconomía y Documentación
Universidad Complutense de Madrid
jose.aguera@ccinf.ucm.es

Resumen [Abstract] [Resum]

Se presenta una propuesta básica de automatización y utilización de tesauros documentales en entornos distribuidos de recuperación de información mediante servicios web basados en Resource Description Framework (RDF). Por esta razón, se revisan en primer lugar las propuestas de marcado descriptivo aparecidas en los últimos cuatro años para la codificación de tesauros documentales. A continuación, se muestra una arquitectura básica de un servidor de tesauros implementado en Java, y, por último, se repasan los distintos protocolos de comunicación e intercambio entre aplicaciones que se pueden usar para implementar este servicio. El texto se acompaña de la aplicación informática que se ha desarrollado.

1 Introducción

Los tesauros documentales son un tipo de lenguaje combinatorio que consta de listas de términos que representan un ámbito científico y técnico determinado y que posee una serie de relaciones semánticas entre los términos que lo conforman. Estas relaciones semánticas son de tres tipos concretos: equivalencia, asociación y jerarquía. Este tipo de lenguajes documentales cuentan con una gran flexibilidad y capacidad de especialización, lo que los hace muy útiles en entornos de recuperación de información (RI) como Internet. La definición más aceptada de *tesauro* es la de un lenguaje documental de estructura combinatoria, de carácter especializado, que se basa en expresiones conceptuales llamadas *descriptores*, provistas de las citadas relaciones semánticas.

Existen varias normas internacionales que establecen las directrices para la construcción de tesauros, entre las que destacan la ISO 2788:1986 y su posterior evolución Z39.19:1993. Según estas normas, los tesauros son realmente instrumentos de control terminológico en entornos de RI, y aunque se pueden encontrar ciertas analogías con otros recursos como las ontologías, la estructura de los tesauros suele ser más mucho más simple y menos definida, además de contar con una menor diferenciación léxico-semántica.

La utilización de tesauros documentales en entornos de RI viene siendo una constate desde hace muchos años. Pese a esto, los procesos de automatización de la RI no siempre han incluido estas herramientas para la desambigüación y normalización semántica de los términos utilizados, como demuestran el gran número de aplicaciones de RI que se basan únicamente en cálculos de tipo estadístico y análisis de frecuencias de aparición de términos como las distintas variantes que existen del *term frequency - inverse document frequency* (TF-IDF).

No obstante, existen algunos proyectos de RI en los que se ha trabajado con recursos lingüísticos informatizados similares a los tesauros para tareas de desambigüación. Ejemplo de esto es la profusión con la que se ha utilizado *Wordnet* durante los años 90 para la realización de este tipo de tareas. El uso de tesauros documentales, aunque más restringido, también está presente en este tipo de proyectos, y es por esta razón por la que consideramos de interés la migración de este tipo de recursos a los procesos de RI basados en Internet.

La RI en Internet es un proceso que tiende a realizarse de forma distribuida, con lo que es interesante que la integración de tesauros documentales en este proceso se adapte a este tipo de arquitecturas. Esta adaptación puede concebirse de distintas formas. En el caso que aquí nos ocupa, proponemos la definición de un servicio de información que permita la utilización distribuida de tesauros mediante una aplicación específica destinada a servir de interfaz entre el tesauro y la aplicación que necesite hacer uso de él. La idea es permitir el uso transparente del tesauro a todas aquellas aplicaciones de RI que necesiten utilizarlo de la misma forma que trabajan los servicios web. Para realizar estas tareas debemos automatizar el tesauro e implementar las funcionalidades básicas de acceso y consulta del tesauro.

Para la realización de este experimento se ha utilizado el tesauro *Spines*, en su versión española, publicada por el CINDOC,[2] aunque el sistema informático desarrollado se ha pensado para que pueda trabajar con cualquier tesauro que cumpla la norma ISO 2788.

2 Modelos de marcado para tesauros

En primer lugar, antes de pensar en la implementación del *software* que realizará las funciones de servidor de tesauros, debemos estudiar las distintas formas de codificación que existen en Internet, donde destacan las propuestas realizadas en el marco de la web semántica para la codificación de tesauros y sistemas de organización de conocimiento.

Si partimos de la base de que todos los tesauros tradicionalmente utilizados en centros de documentación y bibliotecas contienen siempre algún tipo de marcado, es fácil llegar a la conclusión de que lo único que debemos hacer para una primera adaptación de estos recursos al ámbito de la web semántica es codificar este marcado de forma que se adapte a los estándares establecidos. De esta forma podemos aprovechar una gran cantidad de trabajo ya hecho y comenzar a utilizarlo en

Internet de forma inmediata.

La aparición de XML y su amplia aceptación nos dota de la herramienta necesaria para realizar esta conversión del marcado y tan solo nos resta decidir qué modelo es el más adecuado para realizar esta tarea.

2.1 RDF/XML

Si bien la aparición de XML marca el comienzo de una tendencia, sin duda la aparición de RDF (Resource Description Framework) supone un punto de inflexión en cuanto a la creación de una infraestructura semántica que dé soporte a la información que existe en Internet. RDF es un lenguaje para la representación de información sobre recursos que debe ser procesada por máquinas sin que esto suponga una pérdida de significado. RDF se basa en la idea de que podemos identificar los elementos a partir de URIs (Uniform Resource Identification) describiendo los recursos en términos de propiedades simples o pares propiedad-valor, lo cual permite representar las declaraciones simples sobre recursos como un grafo de nodos y arcos que representan los recursos, sus propiedades y sus valores.[3]

En el siguiente grafo RDF se observa como se puede describir a una persona, incluyendo información complementaria sobre ella.

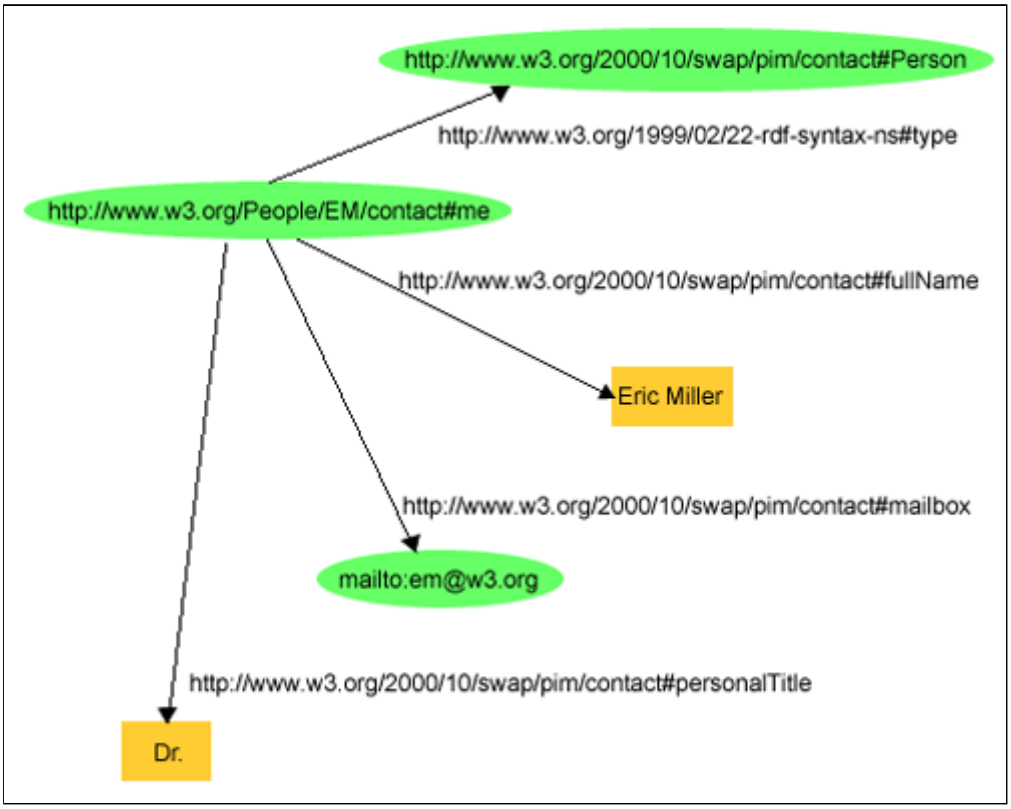

Figura 1. Grafo RDF
(fuente: http://ww.w3.org/TR/rdf-primer/)

En la figura anterior describimos a un individuo, Eric Miller, identificado por el URI: http://www.w3.org/People/EM/contact#em. A su vez se puede identificar con otro URI que lo contiene y define su tipo como: http://www.w3.org/2000/10/swap/pim/contact#Person. Finalmente entre otras propiedades tiene mailbox, identificada por el URI: http://www.w3.org/2000/10/swap/pim/contact#mailbox, cuyo valor es mailto:em@w3.org.

Este grafo se puede representar mediante sintaxis XML de la siguiente manera:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:contact="http://www.w3.org/2000/10/swap/pim/contact#">
    <contact:Person rdf:about="http://www.w3.org/People/EM/contact#em">
        <contact:fullName>Eric Miller</contact:fullName>
        <contact:mailbox rdf:resource="mailto:em@w3.org"/>
        <contact:personalTitle>Dr.</contact:personalTitle>
    </contact:Person>
</rdf:RDF>
```

2.1.1 RDF para la codificación de tesauros

La utilidad de RDF para la representación de tesauros no tardó mucho tiempo en ser descubierta, y ya en el año 2000 se propone el primer schema RDF para la codificación de tesauros. Posteriormente surgen otras propuestas hasta la llegada de SKOS-Core del que hablaremos ampliamente más adelante, pero aquí nos referiremos con detalle únicamente a la propuesta del proyecto CERES como principal antecedente, ya que es la que abre la vía de desarrollo de este tipo de especificaciones.[4]

Esta primera propuesta fue impulsada por la California Resources Agency,[5] y se define a partir del estándar americano para la elaboración de tesauros monolingües Z39.19:1993, cuyo correspondiente internacional es la norma ISO 2788.[6] El objetivo de esta iniciativa es el de proporcionar un método para el intercambio de tesauros entre aplicaciones, ya sea en su totalidad o parcialmente. La razón por la cual se escoge RDF/XML reside en que este lenguaje permite especificar de forma natural las distintas relaciones que existen entre los conceptos que forman el tesauro. A esto le debemos añadir el auge de este tipo de tecnologías que ya existía en Internet en la fecha de elaboración de esta propuesta.

La forma en la que trabaja esta propuesta es muy sencilla, ya que se apoya en la utilización básica de XML para que los documentos puedan ser analizados por cualquier *parser* de XML. Los recursos en esta especificación son considerados como instancias de *Descriptor*, *Category* o *EntryTerm*. A su vez estos, son subtipos de *Term*. Esta implementación utiliza las formas de producción *typedNode* para todos estos recursos.

A su vez las siguientes *propertyTypes* pueden ser especificadas para los *Term*:

- SN: notas aclaratorias (*Scope Notes*)para los descriptores.

- CN: notas del catalogador (*Cataloger Notes*)para los descriptores.
- HN: notas históricas (*Historical Notes*)para los términos.
- *Source*: indicación de la fuente de un término.
- *Status*: indicación del status de un término.

Todas estas *propertyTypes* son a su vez recursos de dos *propertyTypes, Label* y *Term* (*Descriptor — EntryTerm — Category*) que ya hemos visto antes:

- IC: descriptor dentro de una categoría.
- CAT: Categoría del descriptor.
- UF: término de entrada (*EntryTerm*) para el cual se debe usar un descriptor preferente.
- TT: cabeza de jerarquía (*topmost term*) para un descriptor.
- BT: término general (*broader term*) para un descriptor.
- RT: término relacionado (*related term*) para un descriptor.
- NT: término específico para un descriptor.
- USE: descriptor preferente por el cual se debe sustituir un término de entrada (*EntryTerm*).

Mediante la utilización de las anteriores *propertyTypes* obtenemos una representación de los términos que componen el tesauro que se codifica mediante la sintaxis siguiente:

```
<?xml version="1.0"?>
 <?xml:namespace ns='http://www.w3.org/TR/WD-rdf-syntax/'
prefix='RDF' ?>
 <?xml:namespace ns='http://www.w3.org/TR/WD-rdf-schema/'
prefix='RDFS' ?>
 <?xml:namespace ns='http://ceres.ca.gov/thesaurus/' prefix='Z19' ?>
 <RDF:RDF>
 <Category RDF:id="01">
 <Label>Natural Environment</Label>
 <IC>
    <Label>Biosphere</Label>
    <Descriptor RDF:resource="0101"/>
 </IC>
 <IC>
    <Label>Lithosphere</Label>
    <Descriptor RDF:resource="0102"/>
 </IC>
 </Category>
 <Descriptor RDF:id="0101">
  <Label>Biosphere</Label>
  <CAT>
    <Label>Natural Environment</Label>
    <Category RDF:resource="01"/>
  </CAT>
  <TT>
    <Label>Biosphere</Label>
    <Descriptor RDF:resource="0101"/>
  </TT>
  <NT>
    <Label>Ecosystems</Label>
    <Descriptor RDF:resource="010101"/>
  </NT>
</Descriptor>
```

Como se puede ver por los ejemplos mostrados en este caso, la aplicación de RDF al marcado de tesauros es bastante sencilla y funcional, lo cual permitió la adopción de este modelo por varias entidades de importancia.[7] No obstante, a día de hoy, ninguna de ellas conserva este modelo, ya que si bien en su momento proporcionaba una buena solución, el avance de las tecnologías de marcado y la

proliferación de recomendaciones del W3C más adaptadas a su utilización sobre sistemas de organización del conocimiento, provocó un abandono progresivo de este modelo y la evolución hacia otros más elaborados.

2.2 OWL

Sin duda la aparición de OWL (Ontology Web Language) supone un nuevo horizonte en el marcado de sistemas de organización del conocimiento que tiene como punto de partida las experiencias previas realizadas con DAML-OIL en las cuales se inspiraron los creadores de OWL para desarrollar el lenguaje junto con el campo de desarrollo de la lógica descriptiva. OWL es un lenguaje de marcado para la publicación de ontologías en la WWW y tiene como objetivo facilitar un modelo de marcado, construido sobre RDF y codificado en XML que permita representar ontologías a partir de un vocabulario más amplio y una sintaxis más fuerte que la que permite RDF.[8] Por este motivo OWL puede ser utilizado para representar de forma explícita el significado de términos pertenecientes a un vocabulario y definir las relaciones que existen entre ellos.[9]

OWL se divide en tres sublenguajes OWL-Lite, OWL-DL y OWL-Full, cada uno de los cuales proporciona un conjunto definido sobre el que trabajar, siendo el más sencillo OWL-Lite y el más completo OWL-Full. Para los objetivos que aquí nos ocupan OWL-Lite es más que suficiente, ya que, como se indica en la especificación del W3C, permite establecer, entre otras, relaciones jerárquicas entre los conceptos que componen la ontología, a la vez que aporta una menor complejidad formal que sus hermanos mayores. De hecho OWL Lite, en palabras de la propia recomendación del Consorcio, proporciona una forma rápida de migrar tesauros y otras taxonomías al ámbito de la web semántica.[10]

Al igual que en el caso de RDF existen propuestas concretas para la utilización de OWL en la representación de tesauros. Entre ellas destaca la de D.H. Fischer para el tesauro del National Cancer Institute en Alemania el cual se puede descargar gratuitamente desde Internet.[11] El interés de esta propuesta reside en que el autor va más allá de la mera codificación del tesauro mediante OWL-Lite y se plantea la utilización de OWL-DL con la intención de convertir el tesauro en una ontología, no sólo formalmente sino también conceptualmente, lo que le lleva a replantearse la propia organización del vocabulario. No es objetivo de este trabajo profundizar en este aspecto, pero es sin duda de gran interés reflexionar sobre los cambios que suponen en la concepción de los lenguajes documentales clásicos la adopción de estos modelos de representación, cuya utilización aunque en principio sólo afecta en el carácter formal del tesauro, modifica de hecho la organización interna del mismo, como se puede ver en el trabajo ya mencionado de Fischer.

2.3 SKOS-Core

Obviando la reflexión iniciada en el apartado anterior, pero aún salpicados por algunas de sus consecuencias, llegamos a lo que a día de hoy es la propuesta más concreta para la representación de tesauros en el entorno de la web semántica, se trata de SKOS-Core. SKOS-Core es un *schema* RDF para la representación de tesauros y sistemas similares de organización de conocimiento. Esto lo sitúa en la misma línea que el proyecto CERES que hemos comentado en la sección dedicada a RDF, si bien esta aproximación al problema, además de ser una propuesta del W3C, proporciona mecanismos mucho más elaborados que la mostrada anteriormente.

El objetivo fundamental de SKOS-Core es proporcionar un modelo para la migración de sistemas de organización de conocimiento al entorno de la web semántica. Además sirve para construir esquemas de conceptos simples para su utilización en la Web. SKOS-Core está pensado como un complemento a OWL, ya que proporciona un marco básico para la construcción de esquemas de conceptos pero sin la definición semántica tan estricta que exige la utilización de OWL. Se trata en cierta medida una simplificación mayor de la que encontramos ya en OWL-Lite, lo cual permite acceder a un mayor número de personas a este tipo de tecnologías para la representación del conocimiento.

Viendo el desarrollo de SKOS-Core y su vinculación con OWL no podemos evitar pensar en la que en su día tuvieron SGML y XML, ya que en ambos casos se trata de una simplificación del modelo con el objetivo de hacerlo más atractivo a un público más amplio. No es nuestra intención afirmar que SKOS-Core venga a sustituir a OWL, ya que la representación de ontologías requiere de unas capacidades que este *schema* es incapaz de ofrecer, pero sí que es significativo que surja una especificación en el propio seno del Consorcio pensada específicamente para la migración directa de tesauros ya existentes y para su utilización por parte de un mayor número de usuarios. No es sino, desde nuestro punto de vista, la misma fórmula que hizo popular en su día a HTML, o que provocó la sustitución en un gran número de casos de SGML por XML, una fórmula basada en la puesta a disposición del gran público de un modelo sencillo, de fácil utilización y de rápido aprendizaje que permita una expansión de la web semántica generalizada y no sólo en forma de islas.[12] Es prematuro afirmar que SKOS-Core se vaya a convertir en un *schema* de amplia utilización incluso por usuarios no especializados, pero si nos atrevemos afirmar que este tipo de iniciativas ponen al alcance de los profesionales de la información unas herramientas que se adaptan a su nivel de conocimientos técnicos, lo cual será sin duda una de las claves del éxito de la web semántica a nivel global.

Una vez hemos situado SKOS-Core en su contexto con respecto al resto del maremagnum que supone la web semántica podemos pasar a conocer su estructura y composición. La idea base de este *schema* RDF reside en su capacidad para permitir la definición de conceptos y esquemas de conceptos. Un *concepto* se define como una unidad de pensamiento que puede ser definida o descrita. A su vez, un *esquema de conceptos* no es otra cosa que una colección de conceptos. Un concepto puede tener una serie de etiquetas asociadas, donde cada etiqueta es una

palabra, frase o símbolo que suele utilizarse para referirse a ese concepto.

Cada concepto, y aquí ya entramos en ideas familiares para los documentalistas, sólo puede tener una etiqueta preferente, lo que se denomina un descriptor o término preferente, y un número ilimitado de etiquetas alternativas, lo que se denomina un No-Descriptor o término no preferente en la terminología habitual de los vocabularios controlados.

Las mismas relaciones que encontramos en los tesauros tradicionales (equivalencia, jerarquía y asociación) pueden ser asignadas entre conceptos pertenecientes a un mismo esquema de conceptos, el cual podemos asimilar a un tesauro en sí mismo donde las relaciones definidas son de carácter semántico. Además podemos realizar mapeos o equivalencias entre conceptos pertenecientes a diferentes esquemas de conceptos lo que se puede entender como una asociación semántica entre conceptos pertenecientes a distintos tesauros.

Los tipos de relaciones básicas contempladas por SKOS-Core pueden verse en su conjunto en la siguiente figura:

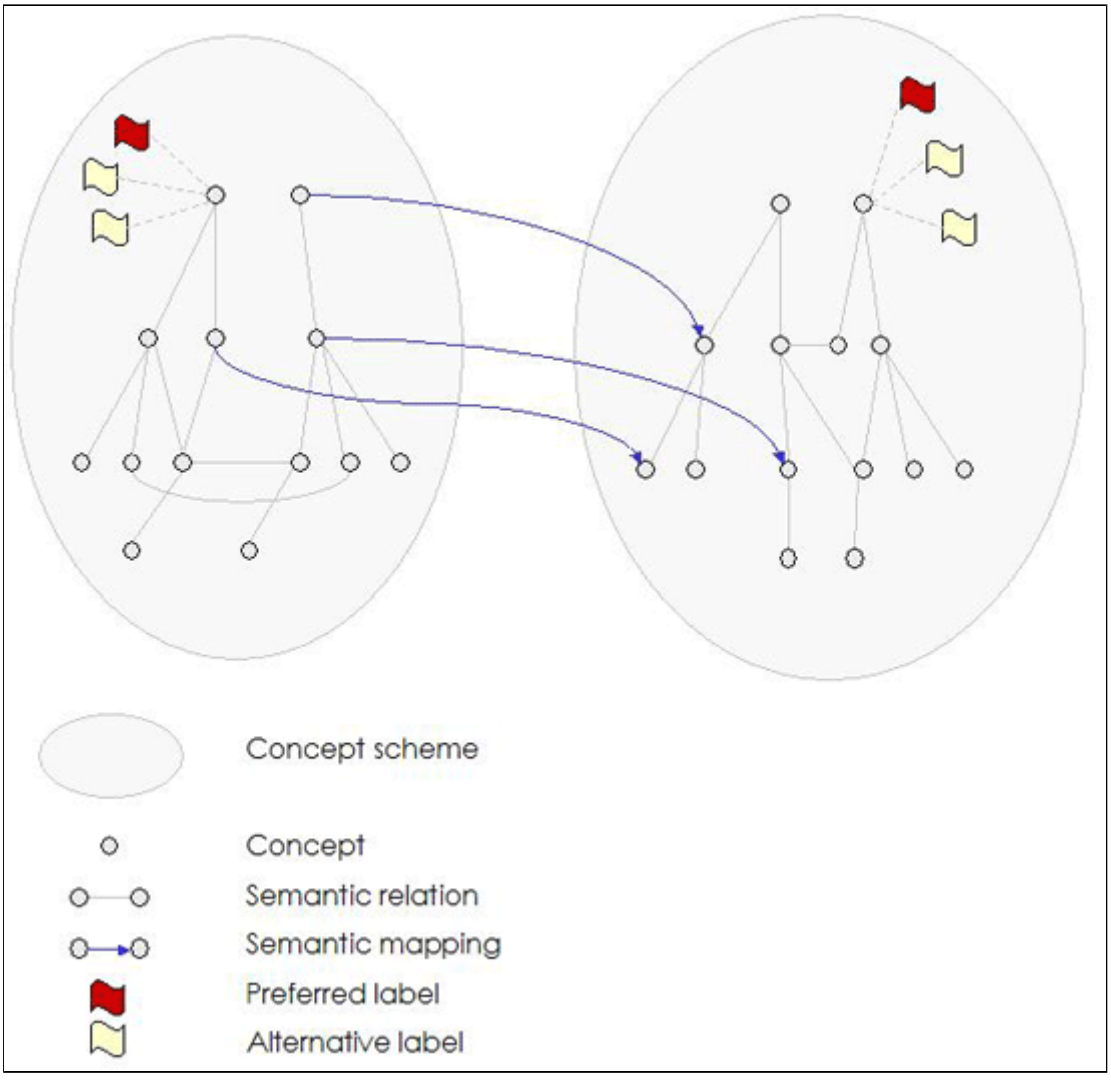

Figura 2. Representación gráfica de dos esquemas de conceptos y sus componentes (fuente: http://ww.w3.org/2001/sw/Europe/reports/thes/1.0/guide/)

Como hemos dicho, el esquema de conceptos lo podemos asimilar a un tesauro. La creación de un esquema de conceptos skos:ConceptScheme es bastante sencilla, ya que tan sólo se necesita definirlo de forma unívoca mediante un URI. A su vez podemos utilizar metadatos, por ejemplo Dublin Core, para describirlo de forma global, como se puede ver en el siguiente código de ejemplo:[13]

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:skos="http://www.w3.org/2004/02/skos/core#"
   xmlns:dc="http://purl.org/dc/elements/1.1/">
   <skos:ConceptScheme rdf:about="http:/spines.org/thesaurus">
      <dc:title>SPINES</dc:title>
      <dc:description>Tesauro de política científica</dc:description>
      <dc:creator>UNESCO</dc:creator>
   </skos:ConceptScheme>
</rdf:RDF>
```

Para definir cada concepto skos:Concept es necesario, asignarle también un identificador unívoco, el cual suele ser un URI, de tal forma que se definen de la forma siguiente:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:skos="http://www.w3.org/2004/02/skos/core#">
   <skos:Concept rdf:about="http:/spines.org/concept/0001">
      <skos:inScheme rdf:resource="http:/spines.org/thesaurus"/>
   </skos:Concept>
</rdf:RDF>
```

Como se puede observar, el concepto es asignado al tesauro creado en el ejemplo anterior a través de la propiedad skos:inScheme. La especificación de esta propiedad es opcional, ya que podemos crear un concepto sin necesidad de que pertenezca a un esquema de conceptos o tesauro concreto. Asímismo, podemos asignar un concepto a varios tesauros repitiendo esta propiedad las veces que sea necesario lo que nos permite ahorrar espacio en especificaciones multitesauro.

Una vez hemos identificado un concepto mediante un URI es necesario que le asignemos las etiquetas correspondientes a los términos con los que ese concepto está relacionado, ya que ésta es la forma en la que relacionamos las palabras o términos con los conceptos. Esta separación tan clara entre información léxica e información semántica es típica de la elaboración de ontologías, y abandonan la línea definida durante años por *Wordnet* para diferenciar, de forma clara, ambos tipos de información. Ésta es, por tanto, una de las muestras a las que hacíamos mención antes cuando comentábamos que la concepción tradicional de los tesauros se ve modificada en mayor o menor medida por la llegada de nuevas perspectivas que, en este caso, vienen de la mano de la web semántica.

La codificación de las etiquetas correspondientes a los términos preferentes y no preferentes pertenecientes a un concepto, se realiza mediante las propiedades skos:prefLabel, para los términos preferentes

o Descriptores y skos:altLabel, para los términos no preferentes o No Descriptores. Esta segunda etiqueta da cobertura a la relación de equivalencia o sinonimia entre términos tan usual en la totalidad de los tesauros utilizados hoy en día. El código tiene el aspecto siguiente:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:skos="http://www.w3.org/2004/02/skos/core#">
   <skos:Concept rdf:about="http:/spines.org/concept/0001">
      <skos:externalID>A.01.0001</skos:externalID>
      <skos:prefLabel>Capital</skos:prefLabel>
      <skos:altLabel>Activo</skos:altLabel>
      <skos:altLabel>Riqueza</skos:altLabel>
      <skos:inScheme rdf:resource="http:/spines.org/thesaurus"/>
      </skos:Concept>
</rdf:RDF>
```

Como se puede ver por el ejemplo anterior, existe la posibilidad de utilizar de forma adicional identificadores para los conceptos que no sean URIs, aprovechando aquellos tesauros que ya cuenten con ellos. Para hacer esto se utiliza la propiedad skos:externalID, la cual acota al identificador que se esté usando.

La utilización de notas aclaratorias es otra de las necesidades que se encuentran cubiertas en SKOS-Core. De hecho podemos ir más allá y asignar definiciones a los conceptos a la vez que ejemplos contextuales e incluso imágenes que estén relacionadas o hagan referencia al concepto. Las propiedades correspondientes son skos:scopeNote para las notas aclaratorias, skos:definition para las definiciones de conceptos, skos:example para los ejemplos contextuales y foaf:depiction ppara las imágenes. La razón por la cual el tratamiento de las imágenes es distinto reside en que estas imágenes son a su vez un recurso en si mismo y por lo tanto deben contar con su propio URI para que se las identifique de forma unívoca. Los siguientes fragmentos de código ilustran estas situaciones.

En el caso de las notas aclaratorias el código correspondiente tendría el aspecto siguiente:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:skos="http://www.w3.org/2004/02/skos/core#">
   <skos:Concept rdf:about="http:/spines.org/concept/0011">
      <skos:prefLabel>Ergonomía</skos:prefLabel>
      <skos:scopeNote>Adaptación del trabajo a los requisitos
      fisiológicos y psicológicos del ser humano</skos:scopeNote>
      <skos:inScheme rdf:resource="http:/spines.org/thesaurus"/>
   </skos:Concept>
</rdf:RDF>
```

Para el caso de las imágenes el código se representaría de la forma siguiente:

```
<rdf:RDF
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:skos="http://www.w3.org/2004/02/skos/core#"
   xmlns:foaf="http://xmlns.com/foaf/0.1/">
   <foaf:Image rdf:about="http:/spines.org/img/red.jpg"/>
   <skos:Concept rdf:about="http:/spines.org/concept/0001">
      <skos:prefLabel>Eritrocitos</skos:prefLabel>
      <skos:altLabel>Glóbulos rojos</skos:altLabel>
      <skos:altLabel>Hematíes</skos:altLabel>
      <skos:inScheme rdf:resource="http:/spines.org/thesaurus"/>
      <foaf:depiction rdf:resource="http:/spines.org/img/red.jpg"/>
```

```
    </skos:Concept>
</rdf:RDF>
```

Como se puede ver en el ejemplo se utiliza *foaf*[14] como propiedad para referirse a una imagen. De la misma forma se pueden utilizar imágenes para referirse a los conceptos a partir de símbolos gráficos.

2.3.1 Definición de relaciones semánticas

Una vez se han definido los elementos que pueden formar parte de un concepto es importante centrarse en las capacidades de expresión de las relaciones semánticas que nos ofrece SKOS-Core, las cuales cubren sobradamente las necesidades de la mayor parte de tesauros utilizados hoy en día en centros de documentación. Así pues, se contemplan relaciones jerárquicas y asociativas, todas ellas agrupadas en torno a una familia de propiedades destinadas a representar relaciones simples entre conceptos. El nivel más alto de esta familia es la propiedad skos:semanticRelation, bajo la cual encontramos las relaciones jerárquicas definidas por las propiedades skos:narrower y skos:broader, para términos específicos y generales respectivamente, y skos:related, para términos relacionados. Recordemos que la relación de equivalencia ya se ha definido previamente a través de la propiedad skos:altLabel.

Como hemos dicho skos:broader y skos:narrower nos permiten crear jerarquías de conceptos. Además es interesante destacar que SKOS-Core nos permite la asignación de varios skos:broader a un mismo concepto, con lo que se contempla la representación de la idea de polijerarquía. El siguiente código muestra de forma más completa la utilización de estas propiedades jerárquicas:

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:skos="http://www.w3.org/2004/02/skos/core#">
    <skos:Concept rdf:about="http://spines.org/concept/0001">
        <skos:prefLabel>Eritrocitos</skos:prefLabel>
        <skos:altLabel>Glóbulos rojos</skos:altLabel>
        <skos:altLabel>Hematíes</skos:altLabel>
        <skos:inScheme rdf:resource="http:/spines.org/thesaurus"/>
        <skos:broader rdf:resource="http:/spines.org/concept/0002"/>
    </skos:Concept>
    <skos:Concept rdf:about="http://spines.org/concept/0002">
        <skos:prefLabel>Sangre</skos:prefLabel>
        <skos:altLabel>Plasma</skos:altLabel>
        <skos:altLabel>Suero sanguíneo</skos:altLabel>
        <skos:inScheme rdf:resource="http:/spines.org/thesaurus"/>
        <skos:narrower rdf:resource="http:/spines.org/concept/0001"/>
    </skos:Concept>
</rdf:RDF>
```

Como hemos mencionado antes, la relación asociativa entre términos relacionados también se encuentra representada mediante la propiedad skos:related, siendo su codificación la siguiente:

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:skos="http://www.w3.org/2004/02/skos/core#">
    <skos:Concept rdf:about="http://spines.org/concept/0001">
        <skos:prefLabel>Eritrocitos</skos:prefLabel>
        <skos:altLabel>Glóbulos rojos</skos:altLabel>
        <skos:altLabel>Hematíes</skos:altLabel>
        <skos:inScheme rdf:resource="http:/spines.org/thesaurus"/>
        <skos:related rdf:resource="http:/spines.org/concept/0002"/>
```

```
    </skos:Concept>
    <skos:Concept rdf:about="http://spines.org/concept/0002">
        <skos:prefLabel>Sangre</skos:prefLabel>
        <skos:altLabel>Plasma</skos:altLabel>
        <skos:altLabel>Suero sanguíneo</skos:altLabel>
        <skos:inScheme rdf:resource="http:/spines.org/thesaurus"/>
        <skos:related rdf:resource="http:/spines.org/concept/0001"/>
    </skos:Concept>
</rdf:RDF>
```

SKOS-Core también nos proporciona mecanismos para la representación de términos en otros idiomas a través de un etiquetado multilingüe. Este etiquetado se basa en la utilización de las propiedades skos:prefLabel y skos:altLabel, a las cuales se le añade el atributo de idioma de RDF expresado mediante el literal correspondiente al idioma en el que esté expresado el término. Para cada idioma se puede poner un término preferente mediante skos:prefLabel, y todos los términos no preferentes que sean necesarios mediante skos:altLabel. La codificación tendría el aspecto siguiente:

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:skos="http://www.w3.org/2004/02/skos/core#">
    <skos:Concept rdf:about="http:/spines.org/concept/0001">
        <skos:prefLabel xml:lang="en">English cuisine</skos:prefLabel>
        <skos:altLabel xml:lang="en">English dishes</skos:altLabel>
        <skos:altLabel xml:lang="en">English food</skos:altLabel>
        <skos:prefLabel xml:lang="fr">Cuisine anglaise</skos:prefLabel>
        <skos:altLabel xml:lang="fr">Plats anglais</skos:altLabel>
        <skos:prefLabel xml:lang="és">Cocina inglesa</skos:altLabel>
        <skos:prefLabel xml:lang="it">Cucina inglese</skos:prefLabel>
        <skos:inScheme rdf:resource="http:/spines.org/thesaurus"/>
    </skos:Concept>
</rdf:RDF>
```

2.3.2 Ampliación de las relaciones semánticas

Hasta aquí hemos visto como SKOS-Core nos proporciona un esquema para el marcado de tesauros lo suficientemente extenso y flexible como para codificar la mayor parte de tesauros con los que se trabaja hoy en día. Ahora bien, las posibilidades de este *schema* RDF no acaban aquí, ya que cuenta con propiedades que nos permiten definir con mayor precisión las relaciones semánticas entre conceptos. Estas propiedades extienden el *schema* más allá de lo que normalmente se hubiera considerado necesario para un tesauro tradicional, y se construyen a partir de especificaciones más generales de las relaciones que ya hemos visto. De esta forma tenemos como tipos de propiedades concretas para las relaciones jerárquicas las siguientes propiedades:

- skos:broaderGeneric y skos:narrowerGeneric
- skos:broaderInstantive y skos:narrowerInstantive
- skos:broaderPartitive y skos:narrowerPartitive

El subconjunto Generic se debe utilizar únicamente para especificar relaciones de subsunción entre dos conceptos. Realmente la semántica de esta propiedad la hereda de la propiedad de RDF rdfs:subClassOf. Por su parte el subconjunto Instantive expresa que un concepto es una instancia de otro en este caso la propiedad heredada de RDF es rdf:type. Finalmente el subconjunto Partitive expresa la idea de que un concepto forma parte de otros lo que no es más que una forma de concretar aún

más la idea de término específico.

El tipo de relación de asociación expresada por los términos relacionados también cuenta con una serie de refinamientos definidos por las siguientes propiedades: skos:relatedHasPart y skos:relatedPartOf.

Ambas propiedades sirven para establecer relaciones asociativas parciales. Si las observamos bien caeremos en la cuenta de que su semejanza con el esquema jerárquico que representan skos:broaderPartitive y skos:narrowerPartitive las hace incluso equivalentes. Los autores de la especificación reconocen esta equivalencia, que ya hemos podido ver en otros casos, y la justifican de cara a favorecer la interoperabilidad entre distintos esquemas de conceptos, lo cual en principio parece bastante razonable.

Para finalizar este repaso a la propuesta actual del Consorcio en materia de tesauros hemos de decir que SKOS-Core admite también la definición de cabezas de jerarquía mediante la propiedad skos:TopConcept la cual se puede usar también para la codificación de tesauros facetados.

2.4 Otros modelos de marcado para la representación de tesauros en Internet

Pese a que el objetivo de este texto es situar la automatización de tesauros dentro del ámbito de la web semántica no podemos dejar de mencionar, aunque sea de forma muy sucinta otros esfuerzos de automatización de tesauros y sistemas de organización del conocimiento de relevancia en Internet.

2.4.1 Zthes

Zthes describe un modelo abstracto para la representación y búsqueda de tesauros tal y como se describen en la norma ISO 2788 antes mencionada. La idea fundamental que reside tras esta propuesta es la de proponer un modelo que permita la implementación de tesauros para que se pueda acceder a ellos mediante el protocolo Z39.50 y SRW. Esta propuesta, debido al auge de los lenguajes de marcado, no puede dejar de ofrecer una DTD para la representación del tesauro, si bien se trata fundamentalmente de una iniciativa centrada en el protocolo Z39.50 con los condicionamientos que esto le supone.[15]

2.4.2 Topic Maps

En segundo lugar tenemos la propuesta que viene de mano de Topic Maps, que es un estándar para la navegación conceptual que posibilita la representación de tesauros. Si bien se trata de una iniciativa con cierta solera y desarrollada en un inicio sobre SGML y HyTime se ha

renovado últimamente mediante la creación de una DTD para la representación de los Topic Maps.[16]

2.4.3 SKOS-Core frente Zthes y Topic Maps

Tanto Topic Maps como Zthes vienen de líneas de trabajo más antiguas que se han visto en la necesidad de adaptarse a los nuevos tiempos marcados por el ritmo y las modas impuestas por la aparición de XML. Esto no quiere decir que estos proyectos sean mejores ni peores sino simplemente que encuentran un campo de aplicación más reducido y suponen soluciones concretas a problemas concretos, por lo cual están lejos de convertirse en un estándar de facto como suele ocurrir con las tecnologías propuestas por el Consorcio. Esta es precisamente la razón por la que hemos optado en este trabajo por un desarrollo basado en RDF/OWL/SKOS-Core.

3 Arquitectura del agente de gestión de tesauros

Una vez hemos elegido SKOS-Core como formato para la codificación de tesauros, podemos pasar a describir el diseño e implementación del núcleo del servidor de tesauros. Este núcleo del sistema se compone de dos clases[17] destinadas a gestionar las funcionalidades básicas de acceso al tesauro, *Thesaurus* y *Concept*, y otras dos destinadas a proporcionar funcionalidades básicas de normalización conceptual, *NormalizarTermino* y *Searcher*.

La clase *Thesaurus* abstrae la idea del tesauro, con el objetivo de modelizarlo y permitir de esta forma su manejo mediante instancias por todas aquellas clases que necesiten realizar acciones sobre él. La clase *Thesaurus* contiene un objeto de tipo *TreeMap* como atributo de clase destinado al almacenamiento del tesauro, donde el par clave/valor viene definido por los descriptores y no descriptores que sirven como punto de entrada al tesauro y por objetos *Concept* que modelizan la idea de concepto, respectivamente. Cada clave descriptor o no descriptor contiene como valor el objeto *Concept* al que pertenece.

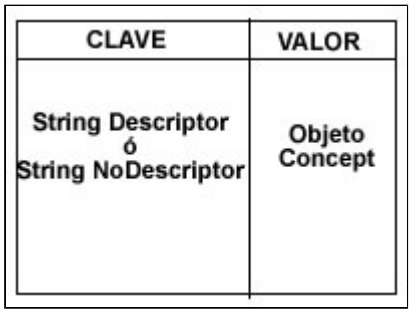

Figura 3. Esquema de la estructura de datos utilizada para la implementación del tesauro

Como hemos mencionado, la clase *Thesaurus* utiliza objetos de tipo

Concept para la abstracción del concepto de descriptor. Este objeto *Concept* incluye atributos relacionados con los componentes que forman parte de un descriptor en los tesauros tradicionales. De esta forma encontraremos entre sus atributos o variables de clase un *String*[18] relativo al término preferente que representa al descriptor y, por lo tanto, a su concepto, y dos *Strings* relativos a la traducción del término preferente al inglés y al francés. A su vez encontraremos también una serie de listas, implementadas mediante *ArrayList*,[19] referentes a términos generales de primer, segundo y tercer nivel, términos específicos de primer, segundo y tercer nivel, categorías a las que pertenece el concepto, términos relacionados y términos equivalentes o no descriptores. Finalmente contamos también con un *String* correspondiente a notas aclaratorias sobre el significado y utilización de los conceptos en el momento de la indización.

A través de esta estructura definimos un concepto de forma muy similar a como se ha hecho en muchos de los recursos lingüísticos informatizados utilizados en los últimos diez años en el campo de la recuperación de información y el procesamiento del lenguaje natural, es decir basado en sus relaciones semánticas con otros conceptos, si bien nos ajustamos a la concepción pura de tesauro documental definida mediante el estándar ISO 2788.

En la siguiente figura se muestran las clases que forman parte de este núcleo:

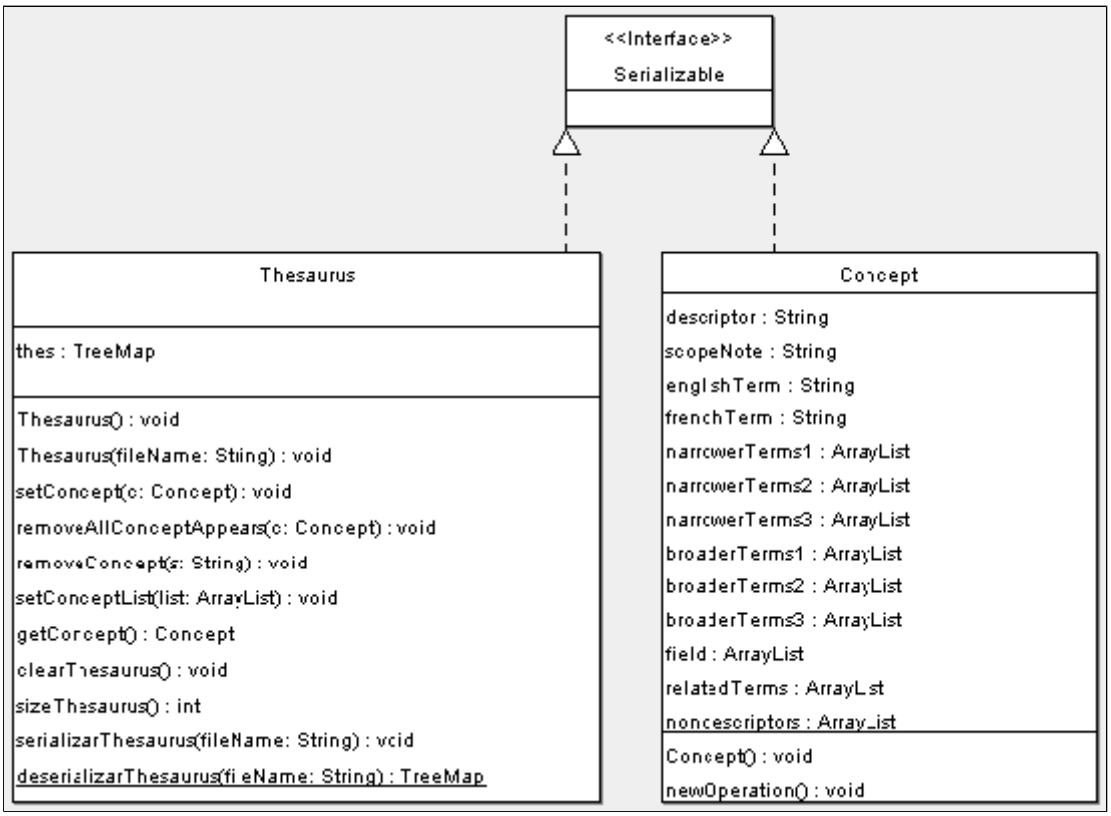

Figura 4. Esquema UML de las clases que componen el núcleo del agente servidor de tesauros

Como se puede ver, ambas clases implementan el interfaz *Serializable*esto es debido a que el almacenamiento del tesauro y de sus descriptores se realiza en un mismo fichero donde se almacenan sendos objetos serializados para que permanezcan de forma persistente más allá de la utilización del programa que los ha creado. Al inicio de la ejecución de la aplicación que utiliza estas clases, se cargan en memoria primaria para ser usados desde aquí durante la ejecución. Esto también permite la carga de varios tesauros por un mismo agente mediante mecanismos de serialización, lo que facilita el mapeo de conceptos entre varios tesauros. Esta forma de almacenamiento difiere de la utilizada normalmente sobre base de datos, y tiene como objetivo mantener la consistencia de los objetos más allá de la ejecución del programa con el objetivo de no tener que generar desde una base de datos los objetos *Thesaurus* y *Concept* cada vez que se usa el programa.[20] Esta implementación no perjudica en exceso la eficiencia del programa y nos permite mantener el enfoque orientado a objetos en todo momento de cara a contar con una mayor simplicidad en el diseño e implementación del sistema.

A partir de este núcleo hemos diseñado a modo de ejemplo un conjunto de clases encargadas de permitir operaciones avanzadas sobre el tesauro a partir de los métodos básicos que contiene la clase *Thesaurus*. Así pues contamos con la clase *NormalizarTermino* y la clase *Searcher*. La clase *NormalizarTermino* pnos permite normalizar conceptos de forma automática contra el tesauro. La funcionalidad es muy sencilla ya que a partir de una entrada compuesta por una palabra o cadena de palabras,[21] esta clase es capaz de devolvernos los términos que normalizan los conceptos representados por el término o términos introducidos. Este proceso de normalización se realiza sobre la totalidad del tesauro, utilizando los términos preferentes y no preferentes como puntos de entrada al mismo. De esta forma, garantizamos la coherencia del proceso de normalización en virtud del tesauro que estamos utilizando. Un ejemplo de utilización de esta clase sería el siguiente:

```
"jose@leviathan:/eclipse/workspace/ThesaurusManager$ java
NormalizarTermino spines cancer"
```

La consulta es “cancer”. El descriptor correspondiente es “neoplasmas malignos”. El número de términos es de 10.772. Aunque en el ejemplo realizamos una normalización simple por un término, la aplicación es capaz de procesar cadenas enteras de términos y devolver los descriptores correspondientes por separado. Asímismo, permite la recuperación de los términos relacionados semánticamente con cada descriptor. Se puede incluso pasar un texto completo para realizar la normalización sobre los conceptos que aparecen en él.

Por su parte la clase *Searcher* funciona de la misma forma que la clase anterior, si bien la diferencia principal reside en que *Searcher* devuelve el resultado de la consulta en formato RDF/SKOS-Core, con el mismo formato que hemos definido anteriormente. La respuesta de esta clase a la consulta anterior tiene el aspecto siguiente:

```
jose@leviathan:/eclipse/workspace/ThesaurusAgent$ java thes.Searcher
```

```
cáncer
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:skos="http://www.w3.org/2004/02/skos/core#"> <skos:Concept
    rdf:about="http://spines/neoplasmas%20malignos">
        <skos:broader rdf:resource="http://spines/enfermedades"/>
        <skos:related rdf:resource="http://spines/transformación%
        20neoplásica%20celular"/>
        <skos:prefLabel>neoplasmas malignos</skos:prefLabel>
        <skos:prefLabel xml:lang="en">malignant neoplasms</skos:prefLabel>
        <skos:prefLabel xml:lang="fr">neoplasmes malins</skos:prefLabel>
        <skos:related rdf:resource="http://spines/neoplasmas%20benignos"/>
        <skos:related rdf:resource="http://spines/hábito%20de%20fumar"/>
        <skos:related rdf:resource="http://spines/enfermedades%
        20incurables"/>
        <skos:altLabel>cáncer</skos:altLabel>
        <skos:altLabel>carcinoma</skos:altLabel>
        <skos:related rdf:resource="http://spines/i+d%20médica"/>
        <skos:related rdf:resource="http://spines/neoplasmas%
        20experimentales"/>
        <skos:related rdf:resource="http://spines/pechos"/>
        <skos:narrower rdf:resource="http://spines/neoplasmas%20inducidos%
        20por%20radiación"/>
        <skos:related rdf:resource="http://spines/antineoplásicos"/>
        <skos:related rdf:resource="http://spines/enfermedades%20de%20la%
        20mama"/>
        <skos:related rdf:resource="http://spines/enfermedades%
        20gastrointestinales"/>
        <skos:broader rdf:resource="http://spines/neoplasmas"/>
        <skos:related rdf:resource="http://spines/condiciones%
        20precancerosas"/>
        <skos:related rdf:resource="http://spines/enfermedades%
        20ginecológicas"/>
        <skos:related rdf:resource="http://spines/cancerígenos%
        20ambientales"/>
        <skos:related rdf:resource="http://spines/enfermedades%20del%
        20aparato%20genital"/>
        <skos:related rdf:resource="http://spines/amianto"/>
        <skos:narrower rdf:resource="http://spines/leucemias"/>
        <skos:narrower rdf:resource="http://spines/sarcoma"/>
    </skos:Concept>
</rdf:RDF>
```

Como se puede ver en este caso la consulta devuelve un documento RDF con el marcado definido en el schema de SKOS-Core con toda la información referente al concepto solicitado.[22]

El objetivo de ambas clases es mostrar algunos ejemplos de las funcionalidades que ofrece la estructura de clases *Thesaurus* y *Concept* anteriormente descrita, tanto para su consulta por usuarios humanos, como es el caso de normalizar término, como para su uso por sistemas de indización automática, como es el caso de *Searcher*.

A partir de estas descripciones de clases podemos definir un núcleo básico del servidor de tesauros que implementa tanto al tesauro como las operaciones que se realizan sobre él y sobre los conceptos. De esta forma el diseño de la aplicación se correspondería con la figura siguiente:

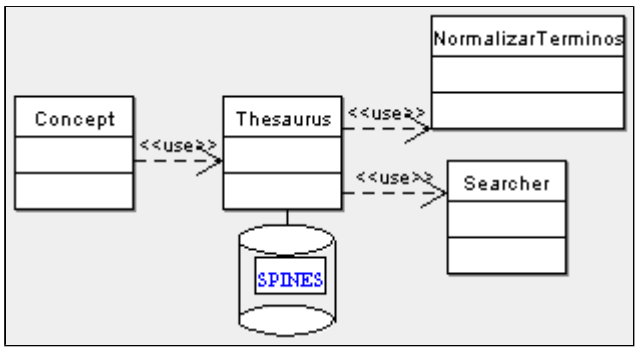

Figura 5. Diseño UML del sistema presentado en este artículo para la normalización conceptual basada en tesauros

3.1 Funcionalidades internas del servidor de gestión de tesauros

Como hemos mencionado anteriormente, nuestro concepto de servidor de tesauros va más allá de permitir la mera consulta del lenguaje, y pasa por la realización de tareas de mantenimiento interno y actualización del tesauro. La necesidad de la actualización de tesauros es un problema que viene siendo tratado desde hace tiempo en la literatura del área de Documentación. Esta afirmación se puede comprobar en el *Manual de lenguajes documentales*, de la doctora Blanca Gil Urdiciain, de la doctora Blanca Gil Urdiciain donde se dedica una sección del capítulo referente a tesauros precisamente a este tema.[23] En este texto, la doctora Gil destaca el hecho de que la extensibilidad que facilita la estructura de los tesauros documentales permite ampliar y modificar el vocabulario en función de las necesidades que surjan a lo largo del uso continuado del mismo. Las necesidades de modificación se hacen más patentes sobre todo en los tesauros de nueva creación, ya que se requiere un tiempo de adaptación del lenguaje documental al entorno en el que se utiliza.

Además, la actualización del tesauro ha de hacerse tanto para incorporar la terminología derivada del desarrollo de la ciencia o materia a la que se dedica como para cubrir lagunas o fallos detectados durante su utilización, así como para adaptarlo a las necesidades de recuperación manifestadas por los usuarios a través de sus búsquedas. Este segundo proceso de corrección y adaptación puede ser automatizado a partir de la detección por parte del servidor de tesauros de estos fallos o lagunas. Uno de los ejemplos más claros de la automatización de este proceso se ejecuta a partir del estudio de las consultas realizadas por los usuarios al recuperar documentos del sistema.

Estas consultas pueden ser procesadas de forma estadística para establecer la capacidad de recuperación de los descriptores utilizados en el tesauro. A partir de la medida de la capacidad de recuperación de cada descriptor y no descriptor contenidos en el tesauro podemos establecer un índice de relevancia que mantenga siempre como descriptor el término con mayor capacidad de recuperación. Podríamos decir que el agente de gestión de tesauros puede ser capaz de aprender, a partir de la práctica de los usuarios, qué términos son los más utilizados

para la recuperación de determinados documentos, de tal forma que asigne siempre a estos documentos los términos con mayor capacidad de recuperación. A esta labor de aprendizaje podemos añadir distintas técnicas automáticas de evaluación, mantenimiento y generación de tesauros que permitan mantener la consistencia del tesauro en todo momento.

4 La comunicación con otras aplicaciones

La automatización del tesauro, tal y como la hemos descrito en la sección anterior, posibilita el uso de tesauros documentales tanto en entornos de indización manual como en sistemas de indización automática y en general por cualquier sistema de RI. Ahora bien, la utilización distribuida del tesauro requiere de la adopción de estándares que permitan la comunicación de forma que el tesauro pueda ser consultado por otras aplicaciones para su uso en sistemas de recuperación de información distribuidos.

En la siguiente sección vamos a describir someramente algunas de las distintas fórmulas de acceso remoto que puede proporcionar un servidor de tesauros. Haremos un breve repaso sobre las distintas opciones que debe implementar un servidor para ser flexible y contar con una amplia capacidad de comunicación. Desde la perspectiva de agentes software veremos FIPA ACL como lenguaje de comunicación entre agentes, mientras que, de cara a los servicios web, describiremos brevemente el protocolo SOAP.

4.1 Comunicación mediante FIPA-RDF

El paradigma de agentes proporciona una teoría sólida sobre la que basar la comunicación entre aplicaciones informáticas; por esa razón la primera consideración que haremos de nuestro sistema de gestión de tesauros será su funcionalidad como un agente. La idea fundamental de este enfoque reside en concebir este sistema como un agente que ofrece como servicio la normalización conceptual de términos extraídos de textos por otros agentes destinados a ejecutar otras tareas del proceso de RI como, por ejemplo, la indización de palabras clave o la extensión de consultas en buscadores.

Si incluimos, por ejemplo, a un agente de indización automática la visión del flujo de trabajo se clarifica, ya que tendremos por un lado un agente de indización de documentos HTML y, por otro, un agente de gestión de tesauros como el descrito en la sección anterior. Cuando el agente de indización extrae los términos literales que aparecen en un documento debe normalizarlos antes de realizar las operaciones de contabilización de frecuencias y asignación de relevancia, ya que se necesita de una normalización semántica antes de realizar las operaciones cuantitativas que permitirán al agente de indización terminar su tarea de manera correcta. De esta forma, se establece una comunicación entre ambos agentes de cara a resolver el problema de la

normalización conceptual de los términos de indización.

El estándar FIPA divide la comunicación entre agentes en actos de comunicación, protocolos de interacción y lenguajes de contenido. Los *actes de comunicació* se componen de bloques constituyentes del diálogo entre agentes donde se define el significado de los mensajes independientemente del contexto. Los *protocols d'interacció* definen una secuencia de mensajes que representan un diálogo completo entre dos agentes. Finalmente, los *llenguatges de contingut* establecen el lenguaje para el contenido del mensaje.

Nuestro servidor de tesauros, al que trataremos aquí como un agente de gestión de tesauros, se puede adaptar a esta forma de comunicación, de tal manera que puede comunicar a otro agente que lo solicite los resultados de la normalización de términos.

La solicitud de normalización de un término se basa más concretamente en la utilización del protocolo de comunicación FIPA-Request, que se usa cuando un agente pide a otro que realice una acción. En nuestro caso, el agente de gestión de tesauros que actúa como destinatario puede aceptar o rechazar la petición y, en caso de aceptarla, deberá realizar la normalización e indicárselo al otro agente cuando finalice. En el siguiente diagrama (figura 6) se observa el flujo de los mensajes. Los que están en blanco son los que envía el "Initiator" al que llamaremos Agente1 y que corresponde con el agente de indización, mientras que los del "Responder"(Agente2), que corresponde al agente de gestión de tesauros, están en gris.

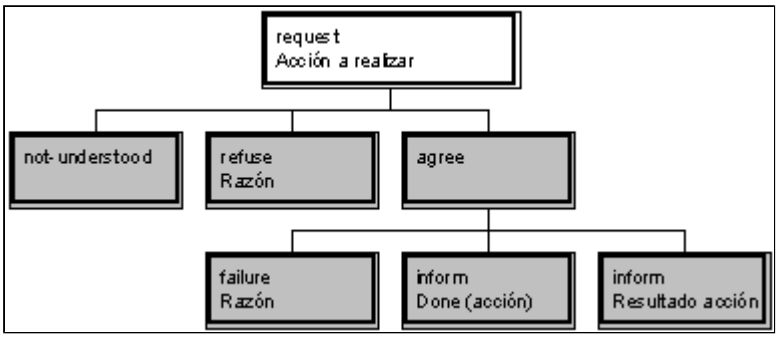

Figura 6. Esquema de comunicación FIPA-Request
(fuente: http://grasia.fdi.ucm.es/SP/index.html)

Gracias a esta secuencia de mensajes, el diálogo coordinado entre los agentes es posible, y además se permite la utilización de los servicios del agente de gestión de tesauros por todos aquellos agentes que lo necesiten. En un sistema multiagente, el número de agentes de gestión de tesauros puede ser tan alto como el número de tesauros que queramos que existan en el sistema.

Por su parte, el atributo *Content* incluido en la respuesta del agente de gestión de tesauros puede contener el documento RDF como los que hemos mostrado antes,[24] con lo cual se mantiene el contenido del mensaje y lo que se hace realmente es adaptar el servidor de tesauros a la forma de trabajar que tendría como agente de gestión de tesauros.

4.2 Servicios web de normalización conceptual con SOAP

El enfoque de agentes no es el único que nos permite afrontar el problema de la comunicación entre aplicaciones de RI. En los últimos años han proliferado en Internet los servicios web muy ligados al desarrollo de los lenguajes de marcado. Estos servicios web basados en XML permiten que las aplicaciones compartan información y que además invoquen funciones de otras aplicaciones independientemente de cómo se hayan creado las aplicaciones, cuál sea el sistema operativo o la plataforma en que se ejecutan y cuáles sean los dispositivos utilizados para obtener acceso a ellas. Aunque los servicios web XML son independientes entre sí, pueden vincularse y formar un grupo de colaboración para realizar una tarea determinada.

Los servicios web no pretenden eliminar del mapa a las librerías o módulos de programación, ya que no son una versión mejorada de éstas, sino una herramienta con distintas aplicaciones en determinados casos. Así, por ejemplo, si necesitamos una rutina que decodifique un fichero de vídeo, no es aconsejable utilizar un servicio web, ya que su utilización supondría enviar el fichero de vídeo al servidor del servicio web, para que éste lo decodifique y envíe el vídeo en formato plano, sin compresión de ningún tipo. Esto supondría un consumo de ancho de banda tan grande que hace que el procesado en local del codec de vídeo sea mucho más eficiente que el procesado remoto.

Sin embargo, hay otras ocasiones en que es interesante utilizar un servicio web, en vez de una rutina de una librería. Por ejemplo, si queremos que una aplicación sepa el precio de un determinado libro dado su ISBN[25] podemos utilizar un servicio web de cara a implementar una aplicación que utilice y procese estos datos para ofrecer al usuario un servicio de valor añadido sobre la información tomada inicialmente. En el caso de la recuperación de información destaca el servicio web basado en SOAP que ofrece *Google* para la utilización de las funcionalidades de su buscador en distintas aplicaciones.[26]

En nuestro caso, utilizaremos los servicios web para permitir a distintas aplicaciones utilizar las funciones de normalización y consulta del tesauro sin necesidad de que implementen aplicaciones ad-hoc. La idea es que se pueda contar con estas funcionalidades sobre el tesauro a través de Internet como si de una caja negra se tratase.[27]

Para la implementación de este servicio se puede utilizar SOAP (Simple Object Access Protocol) debido a que es un protocolo propuesto por el W3C basado en XML para la comunicación de información estructurada entre aplicaciones lo cual se adapta muy bien al uso que le queremos dar. Además SOAP nos permite, al igual que FIPA-RDF, embeber el código RDF generado por nuestro servidor de tesauros en el cuerpo del mensaje,[28] lo que nos proporciona otra forma de comunicación sin necesidad de modificar el formato de la respuesta de nuestro servidor de tesauros.

5 Conclusiones

Como se puede ver por la multiplicidad de opciones comentadas, es posible proporcionar una gran capacidad de comunicación a la aplicación encargada de la gestión del tesauro. El objetivo de los distintos interfaces de consulta mostrados aquí es precisamente éste, ya que la utilización del tesauro como base de conocimiento es un proceso lo suficientemente estandarizado como para que sea de utilidad para los desarrolladores contar con aplicaciones que permitan esta funcionalidad de forma transparente.

La omnipresencia de la RI en Internet y el alto número de aplicaciones que trabajan en Internet para facilitar este servicio hace que sea de interés la concepción de una arquitectura distribuida para el proceso de RI, de tal forma que no haya necesidad de repetir una y otra vez la misma tarea. Este trabajo no es más que un ejemplo sencillo de cómo se pueden llevar a cabo servicios web de RI de utilidad para los desarrolladores de este tipo de sistemas y que faciliten a los documentalistas sistemas de normalización conceptual basados en tesauros.

6 Referencias bibliográficas

Alistair, Milers; Rogers, Nikki; Beckett, Dave (2004). *SKOS-Core 1.0 guide: an RDF schema for thesauri and related knowledge organisation systems*. SWAD-Europe Thesaurus Activity, W3C. <http://www.w3.org/2001/sw/Europe/reports/thes/1.0/guide/> [Consulta: 15/9/2004].

Alistair, Milers; Rogers, Nikki; Beckett, Dave (2004). *SKOS-Core 1.0 guidelines for migration: guidelines and case studies for generating RDF encodings of existing thesauri*. SWAD-Europe Thesaurus Activity, W3C . <http://www.w3.org/2001/sw/Europe/reports/thes/1.0/migrate/> [Consulta: 15/9/2004].

Gil Urdiciain, Blanca (1999). *Manual de lenguajes documentales*. Madrid: Noesis.

McBride, Brian (2002). *An introduction to RDF and the Jena RDF API*. <http://jena.sourceforge.net/tutorial/RDF_API/> [Consulta: 15/9/2004].

Méndez Rodríguez, Eva María (2002). *Metadatos y recuperación de información: estándares, problemas y aplicabilidad en bibliotecas digitales*. Gijón: Trea.

Ogbuji, Uche (2002). *Using RDF with SOAP: beyond remote procedure calls*. <http://www-106.ibm.com/developerworks/webservices/library/ws-soaprdf/> [Consulta: 15/9/2004].

Rogers, Nikki; Beckett, Dave (2004). *SWAD-Europe: use cases for a thesaurus service (draft document)*. <http://www.w3.org/2001/sw/Europe/200311/thes/Use_cases_Thes_Service.html> [Consulta: 15/9/2004].

W3C (2004). *RDF primer: W3C recommendation 10 February 2004*. Brian McBride (series editor). <http://www.w3.org/TR/rdf-primer/> [Consulta: 15/9/2004].

Anexo. Aplicación informática

Fecha de recepción: 28/07/2004. Fecha de aceptación: 3/10/2004.

Notas

[1] Una primera versión de este texto se presentó en el taller "Introducción al uso de la web semántica" organizado por SWAD-Europe en Madrid el 13 de junio de 2004 (http://www.w3.org/2001/sw/Europe/events/200406-esp/.

[2] Puede consultarse en: http://pci204.cindoc.csic.es/tesauros/SpinTes/Spines.htm.

[3] Puede consultarse en: http://www.w3.org/TR/rdf-syntax-grammar/.

[4] En http://www.w3c.rl.ac.uk/SWAD/deliverables/8.2.html#4.1 se encuentra un excelente repaso a la evolución de las distintas propuestas de marcado para tesauros desde el año 2000.

[5] Puede consultarse en: http://ceres.ca.gov/thesaurus/RDF.html.

[6] Puede consultarse en: http://www.niso.org/standards/standarddetail.cfm?stdid=518.

[7] Puede consultarse en: http://ceres.ca.gov/thesaurus/.

[8] Puede consultarse en: http://www.w3schools.com/rdf/rdf_owl.asp.

[9] Recordemos que el significado de los términos y sus relaciones es lo que denominamos una ontología.

[10] Puede consultarse ena: http://www.w3.org/TR/2004/REC-owl-features-20040210/.

[11] Puede consultarse en: http://www.mindswap.org/2003/CancerOntology/.

[12] Eva M. Méndez Rodríguez, *Metadatos y recuperación de información: estándares, problemas y aplicabilidad en bibliotecas digitales* (Gijón: Trea, 2002).

[13] En todos los ejemplos se utiliza el modo abreviado de RDF, ya que es más comprensible y legible.

[14] FOAF es un vocabulario destinado a la descripción de todo tipo de recursos Web, en este caso concreto se trata de imágenes.

[15] Para más información se puede visitar la siguiente dirección: http://zthes.z3950.org/.

[16] Puede consultarse en: http://www.topicmaps.org/.

[17] La palabra *clase* se ha heredado de la metodología orientada a objetos que se ha utilizado para la programación del agente de gestión de tesauros. La aplicación ha sido desarrollada en Java por lo que es interesante que el lector repase algunos conceptos de este lenguaje para la comprensión total del texto que sigue a continuación.

[18] Un *String* es un tipo de dato que se refiere a una cadena de carácteres.

[19] Un *ArrayList* es una de las implementaciones que ofrece Java para las listas dinámicas.

[20] También existe la opción de utilizar Hibernate (http://www.hibernate.org/) para realizar estas tareas, pero no se ha usado debido a que el sistema descrito está dando buenos resultados con tesauros de mediano tamaño.

[21] Incluso admitiría como entrada un texto completo, ya que realiza un tratamiento de cadenas de palabras lo suficientemente complejo como para tratar cadenas que contengan varios conceptos al mismo tiempo.

[22] La generación de RDF se hace mediante el uso de Jena 2.1.

[23] Blanca Gil Urdiciain, *Manual de lenguajes documentales* (Madrid: Noesis, 1996), p. 215–220.

[24] El estandar FIPA-RDF especifica como realizar esta tarea.

[25] Es un servicio web que ofrece Barnes and Noble.

[26] Puede consultarse en: http://www.google.com/apis/.

[27] Los ejemplos de servicios web y la introducción han sido extraídos de http://web-services.bankhacker.com/.

[28] Uche Ogbuji, *Using RDF with SOAP: beyond remote procedure calls*, <http://www-106.ibm.com/developerworks/webservices/library/ws-soaprdf/> [Consulta: 15/9/2004].