Resource Description Framework

Autor: José A. Senso Ruiz (Universidad de Granada)

Citación recomendada: José A. Senso Ruiz. *Resource Description Framework* [en linea]. "Hipertext.net", núm. 1, 2003. <http://www.hipertext.net>

1. Introducción

Resulta evidente que las estructuras de metadatos están adquiriendo una posición preponderante en lo que se refiere a la descripción de recursos electrónicos entendidos como objetos. Cada vez son más numerosos los proyectos, sitios Web o sistemas de consulta que se valen de ellos para lograr mejores prestaciones a la hora de la representación, localización y recuperación de recursos electrónicos.

Al contrario de lo que sucede con formatos de metadatos más complejos y menos flexibles como TEI (Text Encoding Initiative), el sistema más extendido en la actualidad -Dublin Core Metadata Initiative (DCMI en adelante)- pone más énfasis en facilitar al máximo el acceso al recurso y menos en proporcionar una descripción exhaustiva del mismo. Esto resulta vital, ya que ha sido un fallo tradicional en los catálogos bibliotecarios en los que, por el contrario, se hace más hincapié en la descripción que en dotar a los registros de más y mejores elementos de recuperación.

Junto a DCMI destaca, sobre manera, RDF (Resource Description Framework), un sistema de metadatos que sirve tanto para la descripción por sí mismo de recursos electrónicos, como para envolver otros sistemas de metadatos con el fin de logran un marco genérico de trabajo, facilitando de esta forma la interconexión entre diferentes métodos de descripción. En este trabajo analizaremos, de manera breve, las principales características de este sistema así como las dos variantes de su sintaxis, la serializada y la abreviada.

2. Resource description framework

RDF fue creado en agosto de 1997 bajo los auspicios del World Wide Web Corsortium (W3C) con el fin de crear un formato que permitiera alcanzar la compatibilidad entre los diversos sistemas de metadatos, suministrando para ello

una arquitectura genérica de metainformación. Para ello se decidió utilizar el lenguaje XML como sistema de comunicación.

El primer borrador público data del 2 de octubre de 1997 (en agosto fue cuando se reunió por primera vez el grupo de trabajo que se encargaría del desarrollo del formato) y, tras diferentes esbozos, correcciones y propuestas, el 17 de febrero de 1999 aparece la última versión publicada como Recomendación W3C.

Tal y como afirma Hjelm (Hjelm, Johan , 2001), RDF es un formato que tiene como origen dos ramas recientes de la Documentación. Por un lado se encuentran los metadatos -al ser este un sistema que, además de servir como modelo de metadatos, es capaz de interconectar sistemas entre sí- y, por otro, de la representación del conocimiento -encarnada ahora en el novedoso concepto de "semantic Web"-.

La capacidad que tiene RDF para procesar metadatos facilita la interoperabilidad entre diversas aplicaciones, proporcionando un mecanismo perfecto intercambio de información a través del Web. Tal y como se afirma en la Recomendación W3C, RDF tiene distintas áreas de aplicación; como la recuperación de recursos (proporcionando mejores prestaciones a los motores de búsqueda), la catalogación en bibliotecas digitales (especificando también las relaciones de contenido disponibles en un sitio Web determinado), los agentes inteligentes (facilitando el intercambio de conocimiento), en sistemas de gestión de propiedad intelectual (expresando políticas de privacidad de un determinado objeto)... (Brickley, Dan y Guha, R. V. , 2000).

Tras un análisis detallado de la Recomendación, así como de la documentación generada por autores afines al W3C, podemos concluir que los siguientes elementos pueden formar parte de una posible definición de RDF:

- Sistema que permite la interoperabilidad entre aplicaciones mediante el intercambio de información legible por ordenador a través del Web (Brickley, Dan y Guha, R. V. , 2000).
- Mecanismo que facilita la automatización de procesos susceptibles de ser realizados con recursos Web (Lassila, Ora , 199?).
- Infraestructura que permite la codificación, intercambio y reutilización de metadatos estructurados (Miller, Eric , 1998). Es capaz, además, de fusionar diferentes sistemas de metadatos utilizados para la descripción de recursos Web (Iannella, Renato , 199?).
- Es una forma de expresar relaciones entre objetos (Hjelm, Johan , 2001).

El objetivo fundamental de RDF se centra en establecer un mecanismo que permita describir recursos -entendidos estos como objetos- que tengan como principios la multiplataforma (es decir, independencia de software y/o sistema operativo) y la interoperatividad de metadatos (que posibilite fusionar diferentes descripciones de recursos realizadas con distintos conjuntos de metadatos). El mecanismo utilizado para ello debe ser neutral con respecto al área de aplicación y, al mismo tiempo, lo suficientemente flexible como para describir cualquier tipo de información [1]

En cierto modo, parece lógico que RDF naciera en el seno del W3C, ya que esta institución siempre mostró una relación especial con todo lo que tiene que ver con metadatos. De hecho, podemos afirmar que el grupo de trabajo que desarrolló RDF se nutrió de iniciativas de otros colectivos:

- Para empezar, hay reseñar las actividades desarrolladas por un destacado número de empresas (Blue Angel, Verity, IBM, Oracle) y el impulso dado por los

creadores de los principales navegadores; Netscape (Mozilla es capaz de leer RDF y de gestionar bookmarks basándose en él) y, en menor medida, Microsoft. Amaya, el navegador del W3C, permite añadir descripciones personalizadas a páginas Web y a imágenes.

- La aparición, en 1995, de PICS (Platform for Internet Content Selection). Esta infraestructura, que fue originalmente diseñada para ayudar a padres y docentes a controlar el acceso de niños a Internet mediante la asociación de etiquetas HTML, tiene, además, otros campos de actuación. Entre ellos destaca la protección de los derechos de autor y de propiedad intelectual.

2.1. Modelo de datos

De esta forma se conoce al modelo que utiliza RDF para describir recursos. En principio, recurso es entendido en toda la Recomendación W3C como cualquier objeto susceptible de ser identificado mediante un URI (Uniform Resource Locator) - esto incluiría una o varias páginas, una o varias imágenes, un servidor, una o varias animaciones, etc.-.

La clave para que RDF se desarrolle correctamente está en que su modelo de datos utiliza una sintaxis neutral para representar expresiones RDF.

Cualquier objeto se describe utilizando un conjunto de propiedades que se denominan descripciones (description) y que se etiquetan bajo la marca <rdf:description>. RDF representa las propiedades (y sus valores) de un objeto mediante un modelo formal que cuenta con cuatro elementos clave:

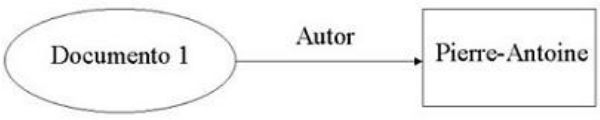
- Recurso: cualquier objeto Web susceptible de ser identificado unívocamente por un URI. Puede ser un documento HTML, una parte de una página, una colección de páginas, un sitio Web completo, una imagen...

- Propiedad: atributo de los recursos. Son aspectos específicos, características, atributos o relaciones utilizadas para describir recursos. Cada tipo de propiedad tiene sus valores específicos. Define los valores permitidos, los tipos de recursos que puede describir y las relaciones que existen entre las distintas propiedades. Corresponden a los pares tradicionales atributo-valor. Además representan las relaciones entre los distintos recursos de tal forma que este modelo puede parecer un esquema entidad-relación (Berners-Lee, Tim , 1997).

- Valor: la representación que toma la propiedad en sí misma.

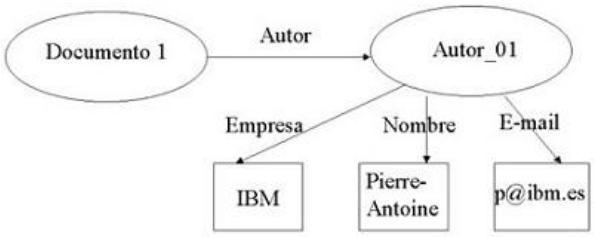
- Descripción: el conjunto que forma un recurso, un nombre de propiedad y el valor de esa propiedad.

Si obviamos el elemento description (al ser el que aglutina a los tres principales), nos encontramos con que la base del modelo RDF es un triple con tres nodos: un sujeto (el recurso) tiene un predicado (propiedad) con un objeto determinado (valor o literal [2]). Eso se podría representar en forma de gráfico de nodos y flechas. Los nodos representan los recursos y se dibujan con óvalos. Las flechas representan propiedades de los nodos y representan cadenas de literales, que pueden dibujarse como rectángulos [3] :



Este esquema se puede complicar con tantos elementos como le queramos añadir a

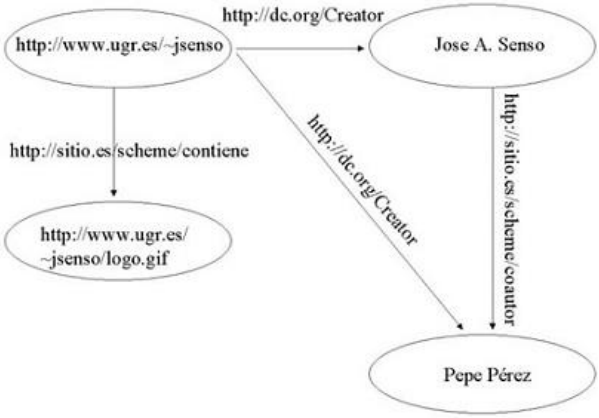
la descripción del recurso. Por ejemplo:



En este caso (Miller, Eric , 1998), la cadena de caracteres "Pierre-Antoine" ha sido sustituida por un identificador de recurso unívoco denominado Autor_01 que tiene asociados tres tipos de propiedades: empresa, nombre y e-mail. El hecho de utilizar un identificador (Autor_01) permite, por un lado, la asignación de propiedades a un sujeto de forma unívoca. Por otro, facilita la inclusión de diferentes conjuntos de propiedades.

Ambos gráficos siguen un modelo común: la descripción de nodos por medio de propiedades y la personalización de estos nodos a través de identificadores unívocos (en caso de RDF, los URIs). La semejanza que tiene este modelo con la programación orientada a objetos -especialmente con la metodología de análisis de declaración de objetos de Booch [4] - no es casual. Si bien es cierto que en RDF los objetos no son de programación, si no más bien de información (no pueden hacer cosas al no tener métodos asociados a ellos, además, el concepto de herencia de propiedades entre clases no es tan claro) (Hjelm, Johan , 2001).

La transformación de estos esquemas a RDF debe pasar por un etiquetado de URIs que permitan calificar a los nodos y a las flechas. Así, un recurso con más de un valor para una propiedad se representaría de esta forma:



En este ejemplo, el recurso http://www.ugr.es/~jsenso determina al autor Jose A. Senso y al coautor Pepe Pérez bajo el esquema de trabajo especificado en http://dc.org/Creator. Al mismo tiempo, y siguiendo otro esquema (explicitado por "http://sitio.es/scheme/coautor" , y por "http://sitio.es/scheme/contiene") describe de nuevo al coautor, así como a una de las imágenes de dicho sitio.

Por último, hay que hacer mención a una se las características más destacadas del modelo de datos de RDF, y es la uniformidad. El hecho de que el único vocabulario que se utiliza es el formado por las URIs permite el uso de una misma URI como nodo y como flecha. De esa forma es posible realizar autoreferencias.

2.2. RDF y XML

Con el fin de lograr sus objetivos, RDF utiliza el lenguaje XML (eXtensible Markup Language) como método para representar y "transportar" la información. Hay que tener muy presente que XML no es un lenguaje de etiquetado. Se trata de un lenguaje que establece un conjunto de reglas que permiten la creación de lenguajes de etiquetado.

XML únicamente muestra las normas a seguir sobre cómo se deben combinar las cadenas de caracteres, cómo se han de especificar las propiedades de los elementos y poco más. Para informar sobre el contenido de cada conjunto de datos, su interpretación, y establecer la forma más correcta de trabajar con ellos debe crearse una DTD (Document Type Definition) donde se plasme el esquema específico de trabajo (Hjelm, Johan , 2001). Por ese motivo RDF utiliza una DTD de XML para desarrollar sus etiquetas.

RDF se beneficia de XML por su flexibilidad a la hora de generar nuevos conjuntos de etiquetas, su orientación multiplataforma y por que proporciona el mecanismo semántico perfecto para expresar la descripción de cualquier tipo de recurso.

Uno de los puntos fuertes del XML es la posibilidad de generar árboles con los elementos y subelementos que forman el documento, a la vez que permite la representación completa del mismo añadiendo o eliminando etiquetas (Senso, Jose A. y Rosa, Antonio de la , 1999). No obstante, la complejidad que tiene la consulta a un árbol XML -generada por la gran cantidad de posibles caminos para llegar a un mapa lógico del documento- imposibilita que pueda ser utilizado como único sistema para describir un objeto (Berners-Lee, Tim , 1998). RDF, por el contrario, emplea una semántica que, inherentemente, aporta cierta coherencia en las búsquedas.

A continuación mencionaremos cómo afectan algunas características propias de XML al desarrollo de RDF, los problemas que presentan y las posibles soluciones propuestas hasta la fecha.

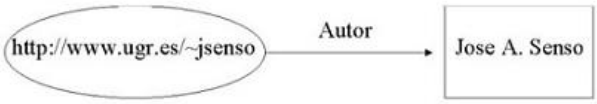
2.2.1. Namespaces

Como ya se ha comentado, XML permite que los desarrolladores generen lenguajes propios. Dado que entre los objetivos de XML se encuentran la multiplataforma y la universalidad, es fácil pensar que exista la necesidad de compartir estos lenguajes con otras personas que desarrollen proyectos similares en el Web. Tenemos un ejemplo claro en XSLT (eXtensible Style Language Transformations) [5]. Este lenguaje debe generar documentos XML bien formados que incluyan etiquetas XSLT. Esto hace necesario que exista un sistema que distinga claramente cuándo esos elementos XML son en realidad instrucciones XSLT y cuándo son simples elementos de salida [6].

Para solucionar este problema se acude a los namespaces, que permiten que cada elemento pueda ser entendido dentro de un entorno específico. Así, los elementos XML que conciernen a instrucciones de transformación XSLT de nuestro ejemplo estarán en el namespace con la URI http://www.w3.org/1999/XSL/Transform , mientras que los elementos XML de salida están en http://www.w3.org/1999/XSL/Format

También es posible definir namespaces para otros elementos, como por ejemplo el Dublin Core o para el elemento memo .

Veamos cómo quedaría el segundo ejemplo tras aplicarle las correspondientes etiquetas RDF:



xml version="1.0"?>
http://www.w3.org/1999/02/22-rdf-syntax-ns #
xmlns:dc="http://purl.org/dc/elements/1.1/">
http://www.ugr.es/~jsenso ">
Jose A. Senso

- El primer elemento () muestra que se está trabajando con XML e informa de la versión en concreto.
- A continuación se encuentra la declaración de namespace de RDF (http://www.w3.org/1999/02/22-rdf-syntax-ns#) es decir, se especifica la URI donde encontrar todas las etiquetas que se utilizarán en el documento XML y su significado.
- Por su parte, la segunda declaración de namespace (xmlns:dc="http://purl.org/dc/elements/1.1/">) indica que además de RDF, se empleará otro sistema de codificación, el Dublin Core, que tiene una URI concreta donde encontrar más información sobre las etiquetas utilizadas y que su prefijo será dc.
- El elemento declara la URI del recurso que se va a describir. Dado que se trata de un recurso externo a la descripción se utiliza el atributo about. Si no fuese así, el atributo ID sería más idóneo.
- La siguiente etiqueta (Jose A. Senso) contiene el valor de la propiedad creator según el sistema dc. En este ejemplo en concreto, el valor es una cadena de caracteres, pero también puede ser una llamada a otro recurso. Para ello bastará con utilizar su URI y la etiqueta correspondiente (rdf:resource="http...").
- El fichero XML se da por concluido una vez declarado el final de la descripción () y el final del código XML ().

Este ejemplo, además, sirve para mostrar que, tal y como ocurre con todos los documentos XML, el fichero RDF tiene una estructura inherente que puede ser visualizada por medio de un árbol.

Tal y como lo define Hjelm (Hjelm, Johan , 2001), cualquier aplicación que tratase de interpretar este documento debería estar formada por tres partes. La primera de ellas se encargaría de la capa semántica, que soporta al contenido. La segunda sería la capa sintáctica, que se encargaría de estructurar la información. Por último, la tercera sería la léxica, donde la semántica y la sintaxis se unen para definir datos concretos. RDF trabajaría dentro de las dos primeras, estableciendo las reglas gramaticales a utilizar con el fin de lograr una descripción coherente.

¿Por qué se utiliza XML y no HTML -como en el caso del Dublin Core-? La respuesta es bastante sencilla. Tal y como se apunta en trabajos anteriores de este autor (Rosa, Antonio de la y Senso, Jose A., 1999a, 1999b, HTML no es un formato de estructuración, si no más bien de representación. HTML permite aplicar una serie de etiquetas de formato a un texto concreto, pero no es capaz de establecer una estructuración con la información con la que trabaja.

2.2.2. URIs versus URLs

Como hemos podido observar, en la mayoría de los casos, las URIs que se utilizan suelen ser URLs. Los URIs fueron creados para ofrecer un mecanismo global y uniforme que permitiera identificar un recurso determinado accesible a través de una red.

Los URIs son universales, es decir, tienen una sintaxis básica con la que trabajar que es independiente del tipo de recuro al que se refiere (bien sea para identificarlo o para dirigirse a él). Se dividen en tres partes:

- protocolo: sirve para describir el mecanismo que se debe utilizar para acceder correctamente al recurso (una página Web, un servidor ftp...)
- hostmane: es decir, el nombre principal de la dirección (en el caso del Web, el asignado mediante el DNS). Junto a él es posible encontrar también la ruta de acceso que es necesario seguir -en ocasiones- para alcanzar el recurso
- consulta adicional: información adicional que permite realizar algún tipo de operación directamente con el recurso. Puede ser una consulta mediando un CGI (utilizando el signo ?) o una llamada a un anchor (#).

En este apartado queda constancia suficiente de la importancia de los URIs como elementos que permiten identificar recursos, namespaces, esquemas... Básicamente, una URI es una cadena de caracteres que sigue las directrices especificadas en el RFC 2396 (Berners-Lee, Tim et al, 1998). Teniendo en cuenta que esta RFC está basada en normas que definen el comportamiento del servicio DNS (Domain Name Service) [7] , es lógico pensar que también utilizará el mismo conjunto de caracteres (el ASCII de 7-bit). Es previsible que en próximas versiones del servicio de dominio se mejore este punto para dar más flexibilidad al direccionamiento.

En realidad el uso de URLs como URIs no está del todo bien visto por la comunidad relacionada con XML. Hay muchos desarrolladores que comparten la idea desarrollada por Champin (Champin, Pierre-Antoine, Euzenat, Jérôme, y Mille, Alain , 2000) de que las URLs no identifican siempre lo que dicen identificar. En muchos casos apuntan a recursos que no existen (o que no son capaces de localizar) y, además, tienen un período de vigencia muy limitado. Si bien es cierto que muchos de estos problemas son reales, en la mayoría de las ocasiones son fácilmente subsanables si se realizara un estudio de los vínculos incluidos en un sitio Web antes de dar el alta.

A modo de conclusión, podemos afirmar que los URI engloban a los URL y a los URN. Los URL describen una ubicación física, los URN un nombre único y los URI lo describen todo.

2.3. Sintaxis

La sintaxis RDF define, de forma simple, el modelo para la descripción de recursos. Este modelo es fundamental para poder almacenar información de forma eficiente, procesarla automáticamente e intercambiar datos entre aplicaciones.

Con el fin de describir los elementos sintácticos, RDF utiliza el sistema de notación EBNF (Extended Backups-Naur Form) [8] , el mismo que XML, de tal forma que asegura la plena compatibilidad entre lenguajes.

En realidad, RDF permite la utilización de dos tipos de sintaxis para realizar la codificación de características de un objeto. Por un lado está la sintaxis serializada, que aporta un conjunto reducido de expresiones para llevar a cabo la descripción. Por otro tenemos la abreviada, que añade un conjunto de entidades (con sus correspondientes atributos) a la serializada con el fin de completar descripciones de subconjuntos dependientes del principal. En la actualidad es posible encontrarse con ambas sintaxis dentro de un mismo documento RDF. A continuación repasaremos brevemente las características de cada una de ellas.

2.3.1. Sintaxis serializada

Es la encargada de establecer los parámetros con los que trabajará el elemento rdf:Description. Tal y como se comentó en el último ejemplo, este componente aporta la URI del recurso con el que se trabajará en RDF a continuación, especificando también su estado. Si se acompaña del atributo about se está indicando que el recurso existe y está localizable e identificable de forma clara. Si, por el contrario, el recurso no tiene asignado un URI, el atributo que se le asignará es ID.

La serialización en formato EBNF tiene la siguiente estructura [9] (Brickley, Dan y Guha, R. V., 2000):

Teniendo en cuenta que en esta sintaxis se utilizan tan sólo los elementos básicos para organizar el esqueleto de la descripción, no es de extrañar el reducido número de valores con los que se trabaja.

[1] RDF	::= ['< rdf :RDF>'] descripción* ['rdf:RDF>']
[2] description	::= '< rdf :Description' idAboutAttr? '>' propertyElt* 'rdf:Description>'
[3] idAboutAttr	::= idAttr | aboutAttr
[4] aboutAttr	::= 'about="' URI-reference '"'
[5] idAttr	::= 'ID="' IDsymbol '"'
[6] propertyElt	::= '<' propName '>' valor '
[7] propName	::= Qname
[8] value	::= descripción | cadena de caracteres
[9] resourceAttr	::= 'resource="' URI-reference '"'
[10] Qname	::= [NSprefix ':'] name
[11] URI-reference	::= cadena de caracteres, interpretada por un [URI]
[12] IDsymbol	::= (cualquier nombre legal en XML)
[13] name	::= (cualquier nombre legal en XML)

[14] NSprefix	::= (cualquier nombre legal en XML)
[15] string	::= (cualquier texto XML, con "<", ">", y "&")

El primero de ellos, que es común en ambas sintaxis, es el elemento RDF , que sirve para especificar el principio y el final de un documento XML estructurado en RDF. En principio se trata de un elemento opcional. Se aconseja no utilizarlo cuando la aplicación (software) con la que se está trabajando se ha configurado para contemplar el dato de que está trabajando con RDF.

<rdf:RDF>descripción</rdf:RDF> Una vez se ha dejado claro que se está usando con este lenguaje, el siguiente paso es el de "abrir" la descripción del recurso. Para ello se utiliza el elemento Description, que se encargará de aglutinar a todos los elementos RDF.

<rdf:Description>... </rdf:Description>

Este elemento puede utilizar dos atributos diferentes:

- about: determina el recurso al que apunta la descripción. Para ello se debe utilizar el identificador de recursos (URI) correspondiente. [10]
- ID: para aquellos recursos que todavía no existen, o no tienen asignado un URI. No puede haber un description con estos dos atributos a la vez (ya que uno indica un recurso existente, y el otro un recurso por existir), ni puede existir un mismo ID repetido dentro del mismo documento.

El tercer elemento que forma parte de esta sintaxis es propertyElt. Una sola description puede tener más de un propertyElt con el mismo nombre de propiedad, lo que significa que cada propertyElt añade una flecha al gráfico dentro del modelo de datos de RDF.

Cada propertyElt cuenta con el atributo resource para especificar que otros recursos (o cadenas de caracteres) son el valor de esta propiedad. Si se usa la opción del recurso, es necesario utilizar el URI, como ya viene siendo habitual. Para las cadenas de caracteres (strings) se han de tener en cuenta los convencionalismos usados en XML. [11] También puede ser empleado para incluir algún elemento procedente de un esquema diferente a RDF. Para ello será necesario especificarlo previamente con su namespace correspondiente. Por ejemplo:

<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf=" http://www.w3.org/1999/02/22-rdf-syntax-ns #"
xmlns:x=" http://www.ugr.es/~jsenso/schema/ ">
<rdf:Description about=" http://www.ugr.es/~jsenso/ ">
<x:Autor>Jose A. Senso</s:Autor>
</rdf:Description>
</rdf:RDF>

Aquí estamos diciendo que se va a valerse de la etiqueta "autor". Esta etiqueta no pertenece a RDF, sino que forma parte de un esquema "X" definido previamente. Si se quiere obtener más información sobre él tan sólo hay que conectarse a la URI especificada en el namespace "http://www.ugr.es/~jsenso/schema"). Como existe la posibilidad de que otros esquemas utilicen el término "autor", es necesario utilizar el identificador de esquema "X", que se encargará de disipar cualquier tipo de duda. El esquema "X" puede tener varias propiedades (autor, editor, lugar...), pero basta con especificar una sola vez su URI en el namespace para que tenga el efecto deseado. Eso sí, si se piensa utilizar más de un esquema es necesario establecer su identificador y su namespace en el lugar especificado para ello en RDF.

Teniendo en cuenta que la declaración de namespace se puede asociar a un elemento description concreto, e incluso con un elemento propertyElt particular, nos podríamos encontrar con casos como el siguiente:

<?xml version="1.0"?>
<RDF xmlns=" http://www.w3.org/1999/02/22-rdf-syntax-ns #">
<Description about=" http://www.ugr.es/~jsenso ">
<s:Autor xmlns:s=" http://www.ugr.es/~jsenso/schema ">Jose A. Senso</s:Autor>
</Description>
</RDF>

O su variante con la declaración de namespace anidada:

...

<Description about=" http://www.ugr.es/~jsenso ">

<Autor xmlns="http://www.ugr.es/~jsenso/schema/">Jose A. Senso</Autor>

...

Ambos casos son desaconsejables. El primero porque generaría unos ficheros innecesariamente largos y bastante difíciles de leer. El segundo porque requeriría que se repitiese la declaración de namespace en cada uno de los propertyElt que se añadieran a la description.

En definitiva, y a modo de conclusión, podemos observar cómo la sintaxis serializada muestra un modelo de transposición del recurso a RDF muy claro, con pocas complicaciones y bien estructurado. El problema lo encontraremos cuando se desee profundizar más en determinados detalles del recurso. Para eso se tendrá que utilizar la sintaxis abreviada.

2.3.2. Sintaxis abreviada

Esta sintaxis tiene como principal punto fuerte el poder interpretar DTDs de XML como modelos RDF. La Recomendación define tres formas de abreviación básica serializada:

La que se utiliza para propiedades no repetidas dentro de un elemento description.

Los valores de esas propiedades serán literales (cadenas de caracteres). Estas propiedades se pueden expresar como atributos XML del elemento description:

Cada propertyElt cuenta con el atributo resource para especificar que otros recursos (o cadenas de caracteres) son el valor de esta propiedad. Si se usa la opción del recurso, es necesario utilizar el URI, como ya viene siendo habitual. Para las cadenas de caracteres (strings) se han de tener en cuenta los convencionalismos usados en XML. [12] También puede ser empleado para incluir algún elemento procedente de un esquema diferente a RDF. Para ello será necesario especificarlo previamente con su namespace correspondiente. Por ejemplo:

Dado que Description no tiene otro contenido (namespace , por ejemplo), la propiedad "Autor" se escribe en forma de atributo XML (ya que así se debe haber expresado en la DTD correspondiente) y, además, se utiliza la misma etiqueta para que marque el fin de la propiedad (con el " />") omitiendo la etiqueta final de Description.

La flexibilidad de esta primera opción la podemos contemplar es los siguientes dos documentos, que ejemplifican dos formas diferentes de decir lo mismo en RDF:

< rdf :RDF>
< rdf :Description
about="http://www.ugr.es/~jsenso">
< x :Editor>Universidad de Granada</ x :Editor>
< x :Titulo>Página personal</ x :Titulo>
< x :Autor>Jose A. Senso</ x :Autor>
</ rdf :Description>
</ rdf :RDF>
< rdf :RDF>
< rdf :Description about="http://www.ugr.es/~jsenso"
x :Editor="Universidad de Granada "
x :Titulo="Página personal"
x :Autor="Jose A. Senso"/>
</ rdf :RDF>

La segunda forma abreviada se emplea sobre fases específicas donde el objeto es otro recurso, y el valor de cualquier propiedad está formado por strings (cadenas de caracteres). Para ello se emplea una propiedad típica de XML: convertir los elementos en atributos. En RDF lo que se hace en concreto es convertir las propiedades de Description en atributos de propertyElt.

```
< rdf :RDF>
< rdf :Description about=" http://www.ugr.es/~jsenso ">
< x :Autor rdf :resource="http://www.ugr.es/personal/56971" />
</ rdf :Description>

< rdf :Description about="http://www.ugr.es/personal/56971">
< v :Nombre>Jose A. Senso</ v :Nombre>
< v :Email>jsenso@ugr.es</ v :Email>
</ rdf :Description>
</ rdf :RDF>
```

Lo que se ha hecho aquí es establecer una primera relación entre un objeto y su identidad dentro del grupo al que pertenece. Una vez se ha establecido un mecanismo de identificación de ese objeto dentro del grupo (por medio del código de personal: 56971), se procede a desarrollar una descripción de su nombre y su dirección de correo electrónico, utilizando para ello lo explicado en la primera forma: aprovechar las propiedades expresadas como atributos XML. Esta relación podría quedar más clara formulada de otra forma donde se muestre claramente la relación arbórea [13]

```
<rdf :RDF>
< rdf :Description about="http://www.ugr.es/~jsenso">
< x :Autor>
< rdf :Description about="http://www.ugr.es/personal/56971">
< v :Nombre>Jose A.
Senso</ v :Nombre>

< v :Email>jsenso@ugr.es</ v :Email>
</ rdf :Description>
</ x :Autor>
</ rdf :Description>
</ rdf :RDF>
```

La última forma se aplica únicamente cuando el elemento Description tiene una propiedad type [14]. Cuando esto sucede, el valor de type puede utilizarse

directamente como un nombre de elemento. Usando la tercera forma abreviada, el ejemplo anterior quedaría así:

```
< rdf :RDF>
< rdf :Description about="http://www.ugr.es/~jsenso">
< x :Autor>
< x :Person about="http://www.ugr.es/personal/56971">
< v :Nombre>Jose A. Senso</ v :Nombre>

< v :Email>jsenso@ugr.es</ v :Email>
</ x :Person>
</ x :Autor>
</ rdf :Description>
</ rdf :RDF>
```

x :Person about="http://www.ugr.es/personal/56971"> y aglutinarían al valor del type que, desarrollado, sería:

```
< rdf :Description about=" http://www.ugr.es/personal/56971 ">
< rdf :type resource="http://www.ugr.es/~jsenso/schema/Person"/>
< v :Nombre>Jose A. Senso</ v :Nombre>
< v :Email>jsenso@ugr.es</ v :Email>
</ rdf :Description>
```

En este caso sí es necesario especificar el namespace utilizado, ya que la sintaxis no corresponde a la de XML. Por lo tanto, hay que poner xmlns:v="http://http://www.ugr.es/~jsenso/shema/

La serialización en formato EBNF mostrada en el apartado 2.3.1 se modificaría de la siguiente forma :

```
El apartado 2: [2] description ::= '<rdf:Description' idAboutAttr? '>' propertyElt* '</rdf:Description>'
```

Pasaría a quedar así (Brickley, Dan y Guha, R. V. , 2000):

```
[2a] description ::= '< rdf: Description' idAboutAttr? propAttr* '/>'| '< rdf: Description' idAboutAttr? propAttr*
```

'>'propertyElt* '</ rdf: Description>'| typedNode

El apartado 6:

[6] propertyElt ::= '<' propName '>' valor '</' propName '>'| '<' propName resourceAttr '/>'

ahora sería (Brickley, Dan y Guha, R. V. , 2000):

[6a] propertyElt ::= '<' propName '>' valor '</' propName '>' | '<' propName resourceAttr? propAttr* '/>'

Y además se añadirían dos elementos más (Brickley, Dan y Guha, R. V. , 2000):

El apartado 16:

[16] propAttr ::= propName '="' cadena de caracteres '"' (con comillas embebidas)

y el apartado 17:

[17] typedNode ::= '<' typeName idAboutAttr? propAttr* '/>' | '<' typeName idAboutAttr? propAttr* '>' property* '</' typeName '>'

Los elementos de la sintaxis de RDF serían por tanto: Description , Alternative , Bag , PropertyElt , Sequence , referencedItem ; mientras que los atributos tendrían los siguiente nombres: aboutAttr , aboutEachAttr , bagIDAttr , idAttr , Idsymbol , member , name , Nsprefix , parseLiteral , propName , Qname , resourceAttr , string , typedNode , URI-reference y value.

A continuación procederemos a la explicación de cada uno de ellos siguiendo el modelo propuesto por Hjelm (Hjelm, Johan , 2001):

Elemento	Explicación	Nombre	Descripción
descripción		RDF	Es el método de marcar el principio y el final de un documento RDF. Es opcional siempre y cuando se especifique el sistema de trabajo para la aplicación en concreto.
propertyElt element	IdAboutAttr es la descripción (con atributos about o ID) mientras que propertyElt	Description	Description aglutina a todos los elementos de RDF.

	es la propiedad.		
valor	Alternative se usa para especificar una lista de recursos que representan, de forma alternativa, el/los valor/es de la propiedad que se está utilizando.	Alternative	Puede ser utilizada para suministrar, por ejemplo, versiones en diferentes idiomas de un valor determinado de la descripción.
valor		Bag	Se trata de una lista desordenada de recursos (URIs) o de valores (cadenas de caracteres) sin especificar un orden determinado.
valor o	PropName es el nombre de la propiedad.	PropertyElt	Una única descripción puede contener más de un elemento PropertyElt con el mismo nombre de propiedad. Un esquema (especificado mediante un namespace y su identificador) es el encargado de definir su interpretación.
valores		sequence	Es una lista ordenada de recursos (con sus URIs) o de literales (cadenas de caracteres). Se utiliza para declarar los múltiples valores que puede tener una propiedad. Estos valores se pueden duplicar.
o valor	resourceAttr es el atributo de la lista de ejemplo.	ReferencedItem o inlineItem	Los contenedores de RDF utilizan estos elementos para expresar una numeración explícita que debe respetarse entre los objetos contenidos. Le asigna automáticamente las propiedades _1, _2...

En lo que respecta a los atributos tenemos:

Atributo	Definido como	Significado
aboutAttr	about=URI	Este atributo permite definir la identidad del recurso que se va a describir. Se refiere siempre a un recurso existente y fácilmente identificable.

aboutEachAttr	aboutEach=URI o aboutEachPrefix=cadena de caracteres	El valor de cualquier aboutEachAttr debe ser un contenedor (bag, alternative o sequence).
bagIDAttr	bagIF=Idsymbol	Especifica el ID del recurso contenedor. ID, al representar un recurso inline, debe tener una denominación unívoca dentro del contenedor.
idAttr	ID=IDsimbol	Si la descripción se refiere a un recurso inline, sin identificación posible, es necesario emplear este atributo.
Idsymbol	Cualquier símbolo que cumpla con la sintaxis XML	Por lo general se suele utilizar cualquier símbolo de Unicode.
member	ReferencedItem o inlineItem	Se usa para mostrar las identidades de los objetos contenidos en los Bags.
name	Cualquier nombre que cumpla con la sintaxis XML	Por lo general se suele utilizar cualquier símbolo de Unicode.
NSprefix	Cualquier NSprefix que se ajuste a la sintaxis XML para namespaces	
parseLiteral	ParseType="Literal" o resource	Se usa para especificar actuaciones del parser RDF en determinados elementos que pueden ser conflictivos.
propName	Qname	Un nombre de propiedad que debe ser escrito siguiendo las pautas de la sintaxis de XML.
Qname	Nsprefix:name	Un nombre cualificador debe estar escrito con la misma estructura que se utiliza para el namespace.
resourceAttr	resource=URI-reference	Especifica el estado del recurso por medio de su URI.
string	Cualquier texto XML con <, > y &	
typedNode	typeName idAboutAttr o bagIdAttr o propAttr /	Utilizado como mecanismo de abreviatura (ver apartado 2.3.2)
URI-reference	Cualquier URI que siga la norma descrita en el RFC 2396	
value	description | cadena de caracteres	El valor de este atributo puede ser una descripción o una cadena de caracteres que siga las pautas de XML.

3. Conclusiones

Tras lo aquí expuesto, este estudio es fácilmente deducible que RDF ofrece la solución técnica necesaria para realizar una descripción homogénea y estricta de los recursos sin necesidad de limitar las opciones de localización y recuperación.

Al mismo tiempo, la posibilidad de utilizar RDF junto a cualquier lenguaje de marcas derivado del SGML le aporta la característica de multiplataforma, lo que lo convierte en la herramienta ideal para crear un entorno de información integrada en el que el catálogo proporcione acceso tanto a los documentos tradicionales como a la información electrónica.

El uso de conjuntos de metadatos que faciliten la interoperatividad entre diversas bases de datos (como puede ser el uso de RDF y DCMI), la utilización de lenguajes de etiquetado más manejables que el SGML y menos simples que el HTML (XML), la aplicación de protocolos pensados para la recuperación de información (Z39.50), el empleo de técnicas de recuperación de información para generar servicios determinados (DSI, por ejemplo) así como el desarrollo del Web, hacen vislumbrar un futuro halagüeño a este sistema de metadatos que tiene como principales características el ser:

- neutral (al no estar ligado explícitamente a ningún otro sistema de metadatos),
- expresivo (las etiquetas son muy intuitivas y se adivina fácilmente su posible contenido),
- familiar (su base SGML lo convierte en asequible para las personas relacionadas con HTML)
- y simple de procesar (al ser texto ASCII).

Por último, no se puede obviar que detrás de RDF se encuentra una institución como el Consorcio W3, lo que le confiere cierto rango de estándar de facto.

4. Bibliografía

Berners-Lee, Tim. Metadata architecture: documents, metadata and links. [Página Web] 6 enero 1997. Consultado en: 1 abril 1999. http://www.w3.org/DesignIssues/Metadata.html

Berners-Lee, Tim. Why RDF is more than XML. [Página Web] septiembre 1998. Consultado en: 21 marzo 2000. http://www.w3.org/DesignIssues/RDF-XML.html

Berners-Lee, Tim et al. RFC 2396: Uniform Resource Identifiers (URI): Generic Syntax. agosto 1998. http://www.isi.edu/in-notes/rfc2396.txt

Brickley, Dan y Guha, R. V. Resource Description Framework (RDF) schema specification. [Página Web] 27 marzo 2000. Consultado en: 21 junio 2000. http://www.w3.org/TR/PR-rdf-schema

Champin, Pierre-Antoine (et al.). Why URLs are good URIs, and why they are not. [Página Web] 5 abril 2000. Consultado en: 10 mayo 2000. http://bat710.univ-lyon1.fr/~champin/urls/

Hjelm, Johan. Creating the semantic web in RDF. New York: Wiley Computer Publishing; John Wiley & sons, Inc., 2001.

Iannella, Renato. An idiot's guide to the Resource Description Framework. [Página

Web] 199? Consultado en: 13 enero 2001. http://archive.dstc.edu.au/RDU/reports/RDF-Idiot/

Lassila, Ora. Introduction to RDF metadata. [Página Web] 199? Consultado en: 23 febrero 2000. http://www.w3.org/TR/NOTE-rdf-simple-intro

Lassila, Ora y Swick, Ralph R. Resource Description Framework (RDF) Model and Syntax Specification. W3C Recommendation. 22 febrero 1999. http://www.w3.org/TR/1999/REC-rdf-syntax-19990222

Miller, Eric. "An introduction to the Resource Description Framework". En: D lib magazine , mayo 1998. http://www.dlib.org/dlib/may98/miller/05miller.html

Rosa, Antonio de la. "Lenguajes de marcas aplicados a la transformación de estructuras documentales". En: El profesional de la información, enero-febrero 2001, v. 10, n. 1-2, pp. 4-22.

Rosa, Antonio de la y Senso, Jose A. "XML como medio de normalización y desarrollo documental". En: Revista Española de Documentación Científica , octubre-diciembre 1999a, v. 22, n. 4, pp. 488-504.

Senso, Jose A. y Rosa, Antonio de la. "Especificaciones XML aplicadas a la Documentación". En: Fuentes i Pujol, Maria Eulàlia dir. Anuario de Biblioteconomía, Documentación e Información. Barcelona: Col·legi Oficial de Bibliotecaris-Documentalistes de Catalunya, 1999b, pp. 133-152.

5. Notas

[1] De ahí la importancia de considerar que cualquier recurso puede ser un objeto. En realidad para RDF -al igual que la mayoría de sistemas de metadatos que han triunfado en el Web- no importa el tipo de recurso con el que se esté trabajando, lo único que es fundamental es establecer un mecanismo que permita identificar unívocamente a ese recurso (ya objeto) dentro del entorno en el que se trabaja. Por este motivo se le da una especial importancia al uso de URIs en las descripciones. [volver]

[2] En la recomendación RDF, los blancos del gráfico (hacia donde apunta a flechas) pueden ser secuencias de texto en lugar de recursos; a esas cadenas de caracteres se denominan literales (literals). [volver]

[3] La dirección de la flecha es importante. La propiedad del nodo (el arco) debe empezar en el sujeto y apuntar siempre hacia el objeto de la frase (el documento 1 tiene como autor a Pierre-Antoine). [volver]

[4] Sistema de anotación para el desarrollo de sotware utilizado en programación orientada a objetos que facilita la redacción de especificaciones técnicas del programa que van a construir. Otros métodos son OMT y OOSE (Object-Oriented Sorftware Engineering). En la actualidad se empiezan a realizar en UML (Unified Modeling Language). [volver]

[5] Este lenguaje permite que el autor, o el receptor del documento XML, pueda especificar cómo se debe presentar el contenido (Rosa, Antonio de la, 2001). [volver]

[6] Incluso si ambos elementos tienen el mismo nombre. [volver]

[7] Este servicio se desarrolla por primera vez en la RFC1034 (www.ietf.org/rfc/rfc1034.txt?number=1034) y en la RFC 1035

(www.ietf.org/rfc/rfc1035.txt?number=1035) [volver]

[8] En realidad utiliza una versión reducida de este sistema. Dado que es el mismo que se emplea en la sintaxis de XML, para RDF se pensó que no era necesaria la inclusión de determinadas características sintácticas heredadas del lenguaje de etiquetado. Por esa razón no se comenta nada sobre las restricciones relativas a los documentos bien formados, el uso del espacio en blanco alrededor de los atributos y el signo igual (=) o la utilización de las comillas simples y dobles dentro de los valores de los atributos. [volver]

[9] Todos los ejemplos han sido extraídos directamente de la Recomendación W3C del modelo de datos y sintaxis RDF (Brickley, Dan y Guha, R. V. , 2000). [volver]

[10] La forma de realizar las llamadas a un recurso -o a partes de un recurso- se encuentran normalizadas en el RFC 2396 (Berners-Lee, Tim et al, 1998). [volver]

[11] En XML se presta especial atención a todos aquellos caracteres especiales que pueden dar lugar a equívocos, como los signos >, <, & o / dado que pueden ser confundidos con principios o finales de etiquetas. [volver]

[12] En XML se presta especial atención a todos aquellos caracteres especiales que pueden dar lugar a equívocos, como los signos >, <, & o / dado que pueden ser confundidos con principios o finales de etiquetas. [volver]

[13] En este ejemplo el atributo about de description se transforma en un atributo de resource en el elemento propertyElt. [volver]

[14] Generalmente se emplea para declarar el tipo de recurso sobre el que se está trabajando. [volver]

Last updated 05-06-2012

Universitat Pompeu Fabra. Departament de Comunicació. Grup de Recerca DigiDoc
Campus de la Comunicació. Roc Boronat, 138, despatx 53804. Barcelona 08018
Tels: 93 542 13 11. Correu electrònic: cristofol.rovira@upf.edu
Depòsit Legal B-49106-2002 - ISSN 1695-5498