Ingeniería de Aplicaciones para la Web Semántica

Clase 07

Los Esquemas RDF

Mg. A. G. Stankevicius

Segundo Cuatrimestre

2005

Copyright

Contenidos

- ¿Hace falta otro lenguaje?
- Jerarquías de clases y de propiedades.
- Sintáxis del lenguaje RDFS.
- Codificación XML para RDFS.
- Especificación formales de RDF y RDFS.
- Semántica axiomática para RDFS.

¿Hace falta otro lenguaje?

- RDF es un lenguaje que permite que describir un dominio empleando terminología propia.
  - No asume ni define semántica alguna para el dominio que se está decribiendo.
- Estos aspectos no cubiertos pueden ser especificados a través de RDFS:
  - Clases y propiedades.
  - Jerarquía de clases y herencia.
  - Jerarquía de propiedades.

Clases y sus instancias

- Debemos recordar las diferencias entre:
  - Los objetos concretos del dominio (e.g., el profesor Mengue Chón).
  - Los conjuntos de individuos que comparten ciertas características (e.g., los profesores).
- Los objetos concretos que pertenecen a una cierta clase de denominan instancias de esa clase.
- Esta relación queda explicitada mediante el atributo `rdf:type`.

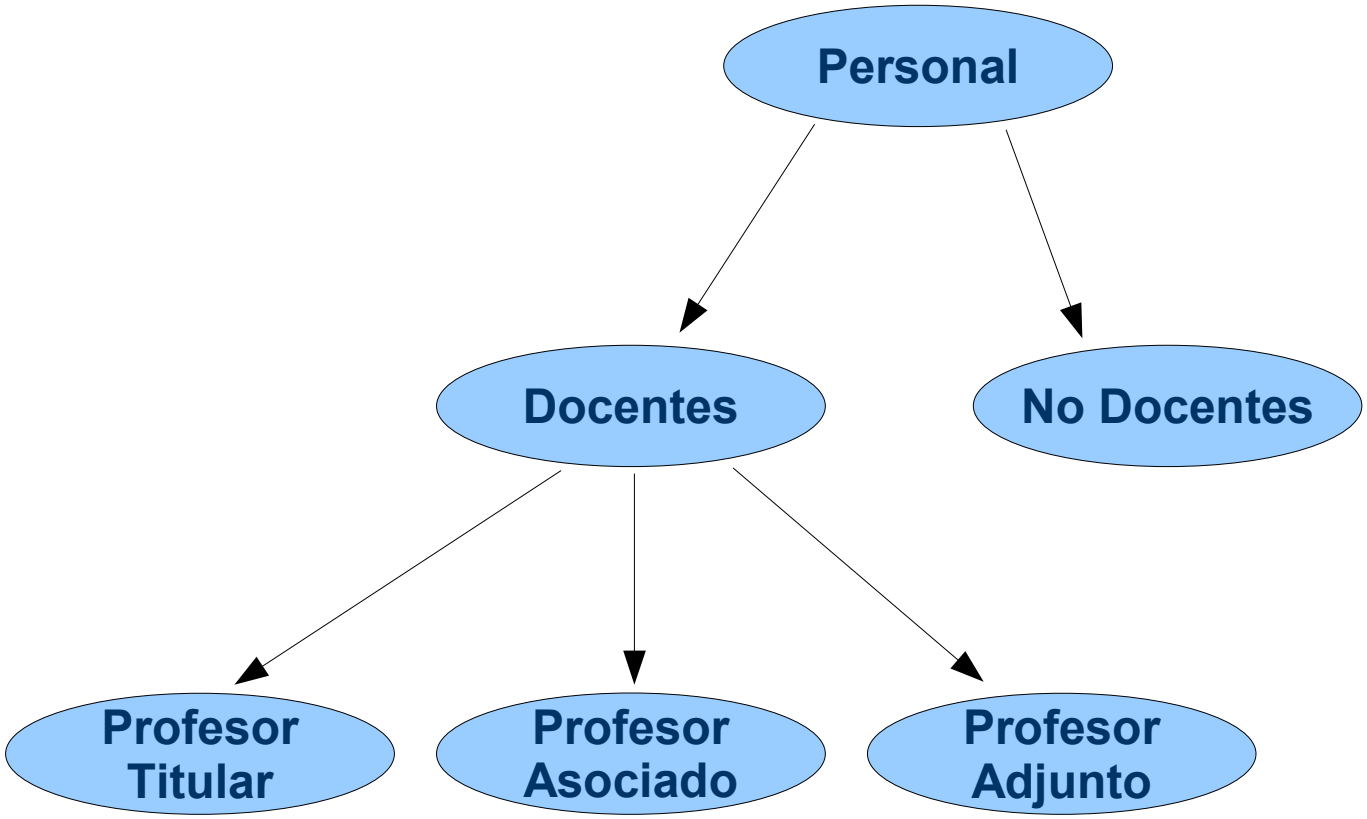
Clases y Jerarquías

- El especificar la jerarquía de clases propia del dominio en consideración aporta distintos beneficios.
- Al igual que en los lenguajes de programación, la información de tipado evita cometer errores triviales.
- Por caso, podemos evitar declaración del tipo *“el curso CS101 es dictado por el curso CS102”*.

Jerarquía de clases

- Las distintas clases identificadas se organizan en una jerarquía:
  - La clase **A** es un subclase de la clase **B** si toda instancia de **A** también instancia de **B**.
  - En este caso, **B** es la superclase de **A**.
- Notar que el grafo de una subclase no necesariamente tiene que ser un árbol.
- De la misma forma, una clase puede tener múltiples superclases.

Ejemplo de jerarquía



Herencia en las jerarquías de clases

- Supongamos que en el contexto del ejemplo anterior se cumplen que:
  - Los cursos sólo pueden ser dictados por los docentes.
  - Mengue Chón es Profesor Asociado.
- Observemos que Mengue Chón hereda la capacidad de dictar cursos.
- Los esquemas RDF capturan este comportamiento fijando la semantica asociada al concepto “es-subclase-de”.

Jerarquía de propiedades

- De manera análoga se puede especificar una estructuración jerárquica para las propiedades:
  - “dictado-por” es una subpropiedad de “cátedra”.
  - Si un curso **A** es dictado por el profesor titular **B**, entonces **A** cuenta en su cátedra al profesor titular **B**.
- La inversa no es necesariamente válida:
  - **B** podría ser un ayudante alumno.

RDF vs. RDFS

- RDFS describe las relaciones entre las clases y las propiedades mencionadas en un determinado dominio.
- RDF describe las relaciones entre objetos concretos, propiedades y sus valores asociados.
- Lo expresado mediante un documento RDF no puede ser capturado a través de los esquemas RDF, y vice-versa.

El lenguaje RDFS

- Las primitivas de modelado del lenguaje RDFS son caracterizadas mediante recursos y propiedades.
- Para declarar que “Profesor Titular” es una subclase de “Docente”, debemos:
  - Definir los recursos `profesorTitular`, `docente` y `subClassOf`.
  - Definir la propiedad `subClassOf`.
  - Formular la declaración `(profesorTitular, subClassOf, docente)`.

Sintáxis XML para RDFS

- Principales clases:
  - `rdfs:Resource`, la clase asociada a todos los recursos.
  - `rdfs:Class`, la clase asociada a todas las clases.
  - `rdfs:Literal`, la clase asociada a todos los literales.
  - `rdf:Property`, la clase asociada a todas las propiedades.
  - `rdf:Statement`, la clase asociada a todas las declaraciones reificadas.

Sintáxis XML para RDFS

- Principales atributos:
  - `rdf:type`, que relaciona recursos con su clase asociada (el recurso en cuestión queda declarado instancia de esa clase).
  - `rdfs:subClassOf`, que relaciona una clase con una de sus superclases (es decir, todas las instancias de esa clase serán instancia de su superclase).
  - `rdfs:subPropertyOf`, que relaciona una propiedad con una de sus superpropiedades.

Sintáxis XML para RDFS

- Principales atributos:
  - `rdfs:domain`, que especifica el dominio de una cierta propiedad (es decir, recursos de qué clase podrán aparecer como sujeto del cual la propiedad habla).
  - `rdfs:range`, que especifica el rango asociado a una cierta propiedad (es decir, recursos de qué clase podrán aparecer como valor asociado a esa propiedad).

Un ejemplo concreto

```
<rdfs:Class
  rdf:about="#profesorTitular">
   <rdfs:subClassOf rdf:resource="#docentes"/>
</rdfs:Class>
<rdf:Property rdf:ID="oficina">
   <rdfs:domain
     rdf:resource="#docentes"
   />
   <rdfs:range
     rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"
   />
</rdf:Property>
```

Relaciones entre las principales clases y atributos

- Tanto `rdfs:subClassOf` como `rdfs:subPropertyOf` se comportan de forma transitiva por definición.
- Considerando que las clases son recursos, `rdfs:Class` es al mismo tiempo subclase de `rdfs:Resource`.
- A su vez, `rdfs:Resource` es instancia de `rdfs:Class`, ya que `rdfs:Resource` es la clase de todos los recursos.
- Toda clase es instancia de `rdfs:Class`.

Propiedades auxiliares

- Otras propiedades útiles:
  - `rdfs:seeAlso` relaciona un recurso con otro recurso que lo explaya más en detalle.
  - `rdfs:isDefinedBy` es un subpropiedad de `rdfs:seeAlso` y relaciona un recurso con la ubicación en la cual figura su definición (usualmente un esquema RDF).
  - `rfds:comment` permite incluir comentarios en la especificación de los recursos.
  - `rdfs:label` asigna un nombre simple de recordar por los humanos.

Un ejemplo concreto

- Formalizando el ejemplo anterior:

```
<rdfs:Class rdf:ID="profesorTitular">
   <rdfs:comment>
     La clase de los Profesores Titulares.
     Todos los Profesores Titulares son a su
     vez Docentes.
   </rdfs:comment>
   <rdfs:subClassOf rdf:resource="#docentes"/>
</rdfs:Class>
...
```

Un ejemplo concreto (cont.)

```
<rdfs:Class rdf:ID="cursos">
   <rdfs:comment>
     La clase de todos los cursos.
   </rdfs:comment>
</rdfs:Class>
<rdf:Property rdf:ID="dictado-por">
   <rdfs:comment>
     Hereda su dominio (Cursos) y su ranfo
     (Docentes) de la superpropiedad "cátedra".
   </rdfs:comment>
   <rdfs:subPropertyOf rdf:resource="#cátedra"/>
</rdf:Property>
```

Un ejemplo concreto (cont.)

```
<rdf:Property rdf:ID="oficina">
   <rdfs:comment>
     It is a property of staff members
     and takes literals as values.
   </rdfs:comment>
   <rdfs:domain
       rdf:resource="#docentes"
   />
   <rdfs:range
       rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"
   />
</rdf:Property>
```

Especificación formal de los lenguajes RDF y RDFS

- Estamos en condiciones de inspeccionar la especificación formal de los lenguajes RDF y RDFS, ya que los mismos se encuentran definidos apelando al lenguaje RDFS.
- El lenguaje RDF se especifica en el siguiente espacio de nombres:
  - http://www.w3.org/1999/02/22-rdf-syntax-ns
- Y el lenguaje RDFS en:
  - http://www.w3.org/2000/01/rdf-schema

Especificación de RDF

```
<rdfs:Class rdf:ID="Statement"
   rdfs:comment="The class of triples
     consisting of a predicate, a subject and
     an object (that is, a reified statement)"
/>
<rdfs:Class rdf:ID="Property"
   rdfs:comment="The class of properties"
/>
<rdfs:Class rdf:ID="Bag"
   rdfs:comment="The class of unordered
     collections"
/>
```

Especificación de RDF

```
<rdf:Property rdf:ID="predicate"
   rdfs:comment="Identifies the property of a
     statement in reified form"
>
   <rdfs:domain rdf:resource="#Statement"/>
   <rdfs:range rdf:resource="#Property"/>
</rdf:Property>
```

Especificación de RDFS

```
<rdfs:Class rdf:ID="Resource"
   rdfs:comment="The most general class"
/>
<rdfs:Class rdf:ID="Class"
   rdfs:comment="The concept of classes. All
     classes are resources"
>
   <rdfs:subClassOf rdf:resource="#Resource"/>
</rdfs:Class>
```

Especificación de RDFS

```
<rdf:Property rdf:ID="subPropertyOf">
   <rdfs:domain rdf:resource="&rdf;Property"/>
   <rdfs:range rdf:resource="&rdf;Property"/>
</rdf:Property>

<rdf:Property rdf:ID="subClassOf">
   <rdfs:domain rdf:resource="#Class"/>
   <rdfs:range rdf:resource="#Class"/>
</rdf:Property>
```

Semántica vs. espacios de nombres

- Consideremos la especificación formal de `rdfs:subClassOf`:
  - El espacio de nombres indica que se aplica a clases y que tiene otra clase como valor asociado.
  - Su significado (“ser subclase de”) no queda capturado.
- Observemos que su significado no puede ser capturado en RDF.
- Hace falta una definición semántica externa.

Semántica axiomática para RDFS

- Para capturar la semántica de las primitivas de modelado de RDFS es posible aplicar un mapeo hacia la lógica de primer orden.
  - La semántica de la lógica especifica claramente la semántica de RDFS.
- Este mapeo permite establecer la base sobre la cual razonadores automáticos podrán manipular la información representada en estos lenguajes.

Convenciones

- Todas las primitivas de modelado en son representadas mediante constantes.
- Un reducido conjunto de predicados predefinidos capturan las relaciones entre estas constantes.
- Se adopta una lógica con igualdad.
- Las variables comienzan con '**?**'.
- Los axiomas se asumen implícitamente cuantificados de forma universal.

Axiomatización de listas

- Símbolos funcionales:
  - nil
  - $cons(x,\ l)$
  - $first(l)$
  - $rest(l)$
- Símbolos predicativos:
  - $item(x,\ l)$
  - $list(l)$

Predicados básicos

- $PropVal(P,\ R,\ V)$
  - Predicado con tres argumentos, que se usa para representar declaraciones RDF acerca del recurso **R**, la propiedad **P** y el valor **V**.
  - La terna RDF `(P,R,V)` se representa como $PropVal(P,\ R,\ V)$.
- $Type(R,\ T)$
  - Abreviatura de $PropVal(type,\ R,\ T)$.
  - Especifica que el recurso **R** es de tipo **T**.

Constantes representando primitivas de modelado

- Constantes adoptadas: $Class$, $Resource$, $Property$ y $Literal$.
- Todas las clases son instancias de la clasa $Class$.
  - $Type(Class,\ Class)$
  - $Type(Resource,\ Class)$
  - $Type(Property,\ Class)$
  - $Type(Literal,\ Class)$

Constantes representando primitivas de modelado

- La clase $Resource$ es la clase más general: todas las clases y todas las propiedades son a su vez recursos:
  - $Type(?p,\ Property) \rightarrow Type(?p,\ Resource)$
  - $Type(?c,\ Class) \rightarrow Type(?c,\ Resource)$
- El predicado de toda declaración RDF es una propiedad:
  - $PropVal(?p,\ ?r,\ ?v) \rightarrow Type(?p,\ Property)$

La propiedad “type”

- $type$ es una propiedad:
  - $PropVal(type,\ type,\ Property)$
- Esta propiedad se aplica a recursos (su dominio) y tiene como valor una clase (su rango):
  - $Type(?r,\ ?c) \rightarrow$

$$(Type(?r,\ Resource) \wedge Type(?c,\ Class))$$

La propiedad auxiliar “FuncProp”

- Diremos que **P** es una propiedad funcional si, y sólo si:
  - **P** es una propiedad.
  - No existen **X**, **Y** y **Z** tales que P(X,Y) y P(X,Z) se verifiquen, con **Y** ≠ **Z**.
- Formalmente:
  - $$Type(?p,\ FuncProp) \leftrightarrow (Type(?p,\ Property) \wedge$$
    $$\forall ?r\ \forall ?v1\ \forall ?v2\ (PropVal(?p,\ ?r,\ ?v1) \wedge$$
    $$PropVal(?p,\ ?r,\ ?v2) \rightarrow ?v1 = ?v2))$$

Semántica de los contenedores RDF

- Los contenedores RDF son esencialmente listas:
  - $Type(?c,\ Container) \rightarrow list(?c)$
- Existen tres tipos de contenedores:
  - $Type(?c,\ Container) \rightarrow$

    $(Type(?c,\ Bag) \vee\ Type(?c,\ Seq) \vee\ Type(?c,\ Alt))$
- El orden de los elementos puede o no importar:
  - $\neg(Type(?x,\ Bag) \wedge\ Type(?x,\ Seq))$

Semántica de los contenedores RDF

- Para cada número natural $n > 0$, se contempla el identificador "_n" para hacer referencia al enésimo elemento de un contenedor.
- Evidentemente, se trata de una propiedad funcional:
  - $Type(\_n, FuncProp)$
- Pero sólo se aplica a contenedores:
  - $PropVal(\_n, ?c, ?o) \rightarrow Type(?c, Container)$

Semántica de las subclases

- *subClassOf* es una propiedad:
  - $Type(subClassOf,\ Property)$
- Si una clase **A** es una subclase de la clase **B**, todas las instancias de la clase **A** son a su vez instancias de la clase **B**:
  - $PropVal(subClassOf,\ ?a,\ ?b) \leftrightarrow$

    $(Type(?a,\ Class) \wedge Type(?b,\ Class) \wedge$

    $\forall ?x\ (Type(?x,\ ?a) \rightarrow Type(?x,\ ?b)))$

Semántica de las subpropiedades

- **P** es una subpropiedad de **Q**, si Q(X, Y) se verifica toda vez que P(X, Y) se verifique.
- Formalmente:
  - $Type(subPropertyOf,\ Property)$
  - $PropVal(subPropertyOf,\ ?p,\ ?q) \leftrightarrow$

    $(Type(?p,\ Property) \land Type(?q,\ Property) \land$

    $\forall ?r\, \forall ?v\ (PropVal(?p,\ ?r,\ ?v) \rightarrow$

    $PropVal(?q,\ ?r,\ ?v)))$

Especificación de dominios y rangos

- Si el dominio de P es D, entonces para cada P(X,Y), X debe pertenecer a D:
  - $PropVal(domain,\ ?p,\ ?d) \rightarrow$

    $\forall ?x \forall ?y\ (PropVal(?p,\ ?x,\ ?y) \rightarrow Type(?x,\ ?d))$
- Si el rango de P es R, entonces para cada P(X,Y), Y debe pertenecer a R.
  - $PropVal(range,\ ?p,\ ?r) \rightarrow$

    $\forall ?x \forall ?y\ (PropVal(?p,\ ?x,\ ?y) \rightarrow Type(?y,\ ?r))$