

RDF (Resource Description Framework)

Modelo de datos para la descripción de *recursos* y relaciones entre recursos, es decir, *datos sobre recursos (metadatos)*

- Facilita la codificación, el intercambio y el reuso de datos
- Facilita la interoperabilidad entre aplicaciones y agentes mediante convenios sintácticos y semánticos

«**Recurso**» = cualquier cosa, identificada por una URIref (referencia URI), sobre la que hay datos, aunque no sea accesible directamente (personas, preferencias, objetos físicos...)

W3C Recommendation 10-Feb-2004: <http://www.w3.org/RDF/>

«RDF/Web Semántica \approx HTML/Web»

Sintaxis básica de RDF: sentencias = triplas

Como las «triplas Objeto-Atributo-Valor», pero con esta terminología:

- *sujeto*, o *recurso*, identificado por una URIref, o «anónimo»
- *predicado*, o *propiedad (rol)*, identificado por una URIref
- *objeto*, o *valor (actor del rol)*, identificado por una URIref, un literal o «anónimo»

<sujeto> <predicado> <objeto>.

«El *sujeto* tiene una propiedad *predicado* cuyo valor es *objeto*»

Todos son *recursos*: sujeto, predicado y objeto (salvo si es literal), *y hasta la misma sentencia*

<sujeito> <predicado> <objeto>.

En el lenguaje de FOL (First Order Logic)
y los de DL (Description Logics) se escribe:

predicado(Sujeto, Objeto)

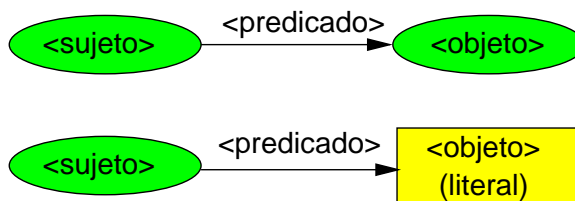
donde Sujeto y Objeto son *constantes* (recursos concretos).

El que las sentencias puedan considerarse también recursos
(«cosificación») hace que el modelo lógico sea de orden superior:

pred2(pred1(Suj1, Obj1), Obj2)

(lógica de predicados de segundo orden)

Grafos y serializaciones en Notation 3 y en XML



N3:

```
<URIfref_de_sujeto> <URIfref_de_predicado> <URIfref_de_objeto> .  
<URIfref_de_sujeto> <URIfref_de_predicado> "literal" .
```

XML:

```
<?xml version="1.0" encoding="iso-8859-1"?>  
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">  
  <rdf:Description rdf:about="URIfref_de_sujeto">  
    <URIfref_de_predicado rdf:resource="URIfref_de_objeto" />  
  </rdf:Description>  
  
  <rdf:Description rdf:about="URIfref_de_sujeto">  
    <URIfref_de_predicado>literal</URIfref_de_predicado>  
  </rdf:Description>  
</rdf:RDF>
```

Ejemplo de tripla



N3:

```
<http://www.example.org/index.html>
  <http://www.example.org/terms#autor>
    "Pepita Perez" .
```

XML (resultado de traducir con `cwm --n3 RDF01.n3 --rdf`):

```
<rdf:RDF xmlns="http://www.example.org/terms#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  <rdf:Description rdf:about="http://www.example.org/index.html">
    <autor>Pepita Perez</autor>
  </rdf:Description>
</rdf:RDF>
```

Píldora lógica: problema semántico en RDF

Los recursos son elementos del *universo del discurso*:

$$\text{recurso} \in \mathcal{U}$$

Una *función de interpretación semántica* debe asignar a los símbolos de constante elementos de \mathcal{U} , y a los símbolos que representan relaciones de grado n elementos de \mathcal{U}^n

$$I(\text{URIref}) = \text{recurso}$$

Problema de la definición semántica de RDF:

como las propiedades son recursos, propiedad $\in \mathcal{U}$

(en lugar de propiedad $\in \mathcal{U} \times \mathcal{U}$)

«autor» no debería interpretarse como un recurso, sino como lo que es: una relación entre dos recursos.

Herramientas

Validador de RDF en línea

- Se puede pegar en un cuadro de texto o pedirle que visite un URL
- Señala errores sintácticos; si no los hay da una traducción a triplas y, opcionalmente, un grafo (muy prolijos: URIs absolutos)
- <http://www.w3.org/RDF/Validator/>

IsaViz

- Entorno visual para edición de modelos RDF con una interfaz 2.5D
- Importa y exporta N3, NTriples y RDF/XML. Exporta PNG y SVG
- Aplicación Java/Swing/GSS
- <http://www.w3.org/2001/11/IsaViz/>

CWM («Closed World Machine»)

- Programa Python que analiza y traduce N3, NTriples y RDF/XML
- Almacena triplas en una base de datos que puede consultarse
- Contiene un motor de inferencias con encadenamiento hacia adelante
- <http://www.w3.org/2000/10/swap/doc/cwm>

Uso de vocabularios estándar: Dublin Core



<http://purl.org/dc/elements/1.1/>, ISO15836, 15 elementos:

- Relacionados con el recurso:
Date, Type, Format, Identifier
- Relacionados con el contenido del recurso:
Title, Subject, Description, Source, Language, Relation, Coverage
- Relacionados con la propiedad intelectual:
Creator, Publisher, Contributor, Rights

<http://purl.org/dc/terms/>
55 elementos, incluyendo los 15 anteriores.
Especificaciones de dominios y rangos.

Ejemplo con Dublin Core y *qnames*



N3:

```
@prefix ex: <http://www.example.org/> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
ex:index.html    dc:creator    "Pepita Pérez" .
```

XML:

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/">
  <rdf:Description rdf:about="http://www.example.org/index.html">
    <dc:creator>Pepita Pérez</dc:creator>
  </rdf:Description>
</rdf:RDF>
```

Qnames: nombres cualificados

Qname = prefijo asignado a un *namespace* + ":" + nombre local
(Prefijo asignado a un *namespace* + ":" = «vocabulario»)

Prefijos habituales:

Prefijo	URI
dc	http://purl.org/dc/elements/1.1/
dcterms	http://purl.org/dc/terms/
xsd	http://www.w3.org/2001/XMLSchema#
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#
rdfs	http://www.w3.org/2000/01/rdf-schema#
owl	http://www.w3.org/2002/07/owl#
ex	http://www.example.org/
exterm	http://www.example.org/terms#
exstaff	http://www.example.org/staffid/

Objetos no literales (recursos): `rdf:resource`



N3:

```
@prefix ex:      <http://www.example.org/> .
@prefix exstaff: <http://www.example.org/staffid/> .
@prefix dc:      <http://purl.org/dc/elements/1.1/> .
ex:index.html   dc:creator   exstaff:A007 .
```

XML:

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/">
  <rdf:Description rdf:about="http://www.example.org/index.html">
    <dc:creator rdf:resource="http://www.example.org/staffid/A007"/>
  </rdf:Description>
</rdf:RDF>
```

Abreviaturas en RDF/XML: `about` e ID

Como *valores* de los atributos `rdf:about` y `rdf:resource` no se pueden utilizar qnames, pero sí URIs relativos.

`rdf:about="#autor"` es relativo a la localización del documento en el que aparece, a menos que en el elemento `rdf:RDF` se haya incluido `xml:base="http://..."`

$$\text{URIref} = \text{xml:base} + \# \text{autor}$$

Si en lugar de `rdf:about="#autor"` se pone `rdf:ID="autor"`,

$$\text{URIref} = \text{xml:base} + \# + \text{autor}$$

`rdf:ID` es una abreviatura sintáctica para expresar que el valor del atributo es un identificador local. Por tanto:

$$\text{rdf:ID}="x" \equiv \text{rdf:about}="#x" \equiv \text{rdf:about}="URI_abs\#x"$$

Uso de entidades XML

Siempre se puede abreviar definiendo entidades XML:



```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
  <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  <!ENTITY dc "http://purl.org/dc/elements/1.1/">
  <!ENTITY ex "http://www.example.org/">
  <!ENTITY exstaff "http://www.example.org/staffid/">
]>
<rdf:RDF xmlns:rdf="&rdf;"
  xmlns:dc="&dc;"
  <rdf:Description rdf:about="&ex;index.html">
    <dc:creator rdf:resource="&exstaff;A007"/>
  </rdf:Description>
</rdf:RDF>
```

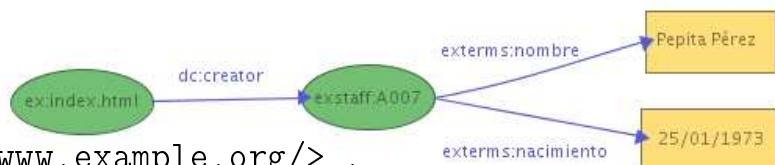
Objetos como sujetos

N3:

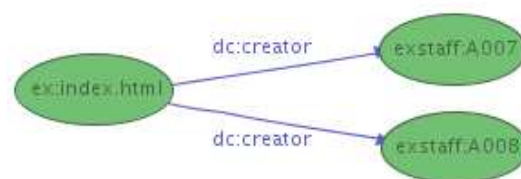
```
@prefix ex: <http://www.example.org/> .
@prefix exstaff: <http://www.example.org/staffid/> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix : <http://www.example.org/terms#> .
ex:index.html dc:creator exstaff:A007 .
exstaff:A007 :nacimiento "25/01/1973";
exstaff:A007 :nombre "Pepita Pérez" .
```

XML:

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns="http://www.example.org/terms#">
  <rdf:Description rdf:about="http://www.example.org/index.html">
    <dc:creator rdf:resource="http://www.example.org/staffid/A007"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.example.org/staffid/A007">
    <nombre>Pepita Pérez</nombre>
    <nacimiento>25/01/1973</nacimiento>
  </rdf:Description>
</rdf:RDF>
```



Predicado con varios objetos



N3:

```
@prefix ex: <http://www.example.org/> .
@prefix exstaff: <http://www.example.org/staffid/> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
ex:index.html dc:creator exstaff:A007 ,
exstaff:N723 .
```

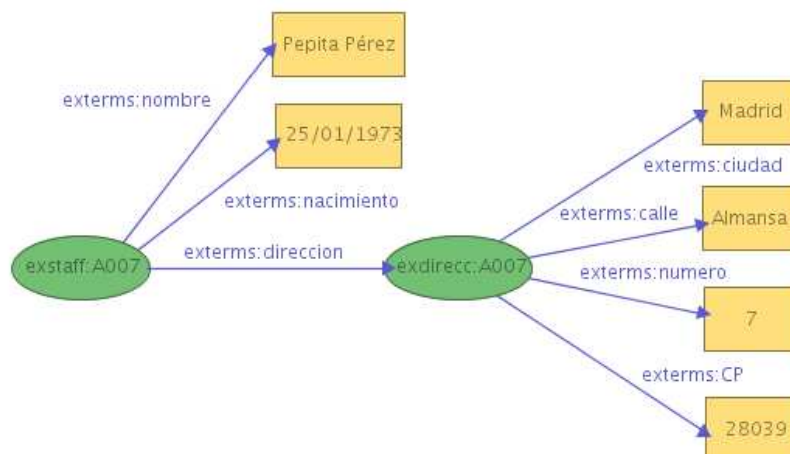
XML:

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:dc="http://purl.org/dc/elements/1.1/">
  <rdf:Description rdf:about="http://www.example.org/index.html">
    <dc:creator rdf:resource="http://www.example.org/staffid/A007"/>
    <dc:creator rdf:resource="staffid/N723"/>
  </rdf:Description>
</rdf:RDF>
```

Valores estructurados

Si el literal tiene un valor que ha de procesarse como una estructura, se sustituye el literal por un recurso nuevo sobre el cual se enuncian nuevas propiedades. Por ejemplo, «dirección»:

- como literal: "Almansa, 7, 28039 Madrid"
- como recurso con propiedades "calle", "número", "ciudad"...



Píldora lógica: relaciones de grado n

La introducción de un recurso nuevo es un artificio para poder representar relaciones de cualquier grado con triplas (que son relaciones binarias). En lugar de:

dirección(Pepita, Almansa, 7, 28039, Madrid)

representamos con:

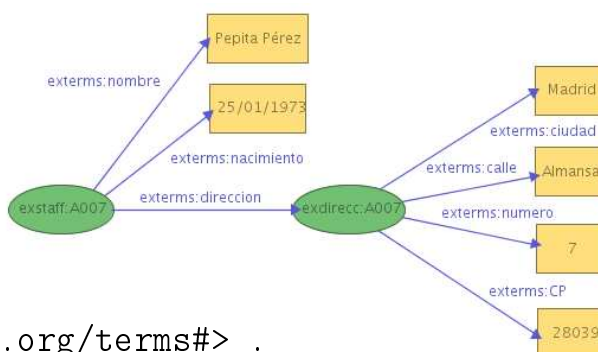
dirección(Pepita, Dir)

calle(Dir, Almansa)

num(Dir, 7) ...

En general, para representar una relación de grado $n > 2$ se selecciona uno de los elementos (*Pepita* en este caso) como sujeto de la relación original (*dirección*), se crea un nuevo recurso para representar el resto de la relación (*Dir* en este caso, o bien anónimo) y se le atribuyen propiedades que representan a los demás componentes de la relación original.

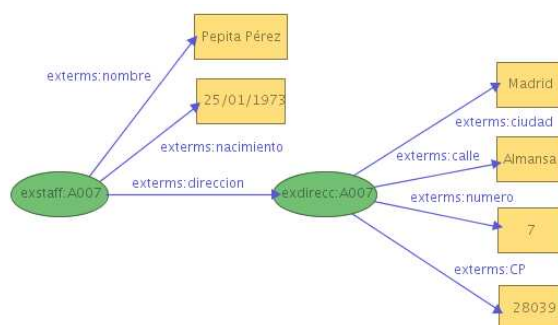
Valores estructurados: el ejemplo en N3



```
@prefix : <http://www.example.org/terms#> .
@prefix exstaff: <http://www.example.org/staffid/> .
@prefix exdirecc: <http://www.example.org/exdireccid/> .
exstaff:A007 :nombre "Pepita Pérez" ;
             :nacimiento "25/01/1973" ;
             :direccion exdirecc:A007 .
exdirecc:A007 :calle "Almansa" ;
              :numero "7" ;
              :CP "28039" ;
              :ciudad "Madrid" .
```

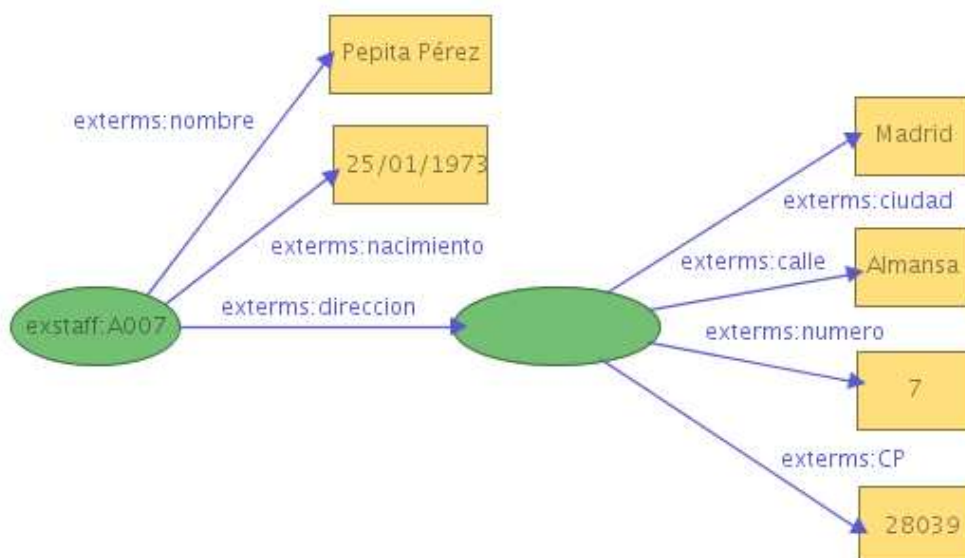
Valores estructurados: el ejemplo en XML

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns="http://www.example.org/terms#">
  <rdf:Description rdf:about="http://www.example.org/staffid/A007">
    <nombre>Pepita Pérez</nombre>
    <nacimiento>25/01/1973</nacimiento>
    <direccion>
      <rdf:Description
        rdf:about="http://www.example.org/exdireccid/A007">
        <calle>Almansa</calle>
        <numero>7</numero>
        <CP>28039</CP>
        <ciudad>Madrid</ciudad>
      </rdf:Description>
    </direccion>
  </rdf:Description>
</rdf:RDF>
```



Nodos en blanco, o «bnodes» (recursos anónimos)

¡Pero no nos interesa la identidad del nuevo recurso, sólo sus propiedades!

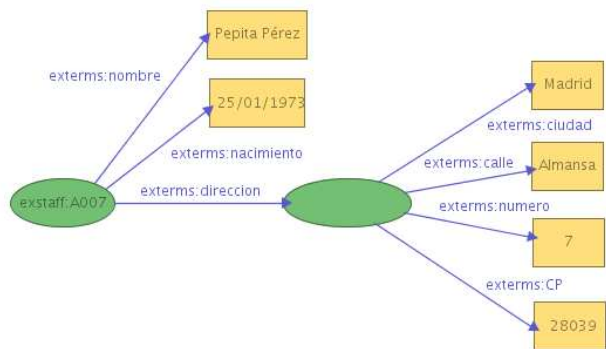


bnodes en N3

```
@prefix : <http://www.example.org/terms#> .
@prefix exstaff: <http://www.example.org/staffid/> .
exstaff:A007
  :nombre "Pepita Pérez" ;
  :nacimiento "25/01/1973" ;
  :direccion
    [ :calle "Almansa" ;
      :numero "7";
      :CP "28039" ;
      :ciudad "Madrid"
    ] .
```

O bien:

```
  :direccion :_b1.
:_b1 :calle "Almansa" ;
...
```



Píldora lógica: cuantificación existencial

En términos de lógica, los bnodes corresponden a variables cuantificadas existencialmente: «existe una dirección que tiene propiedades...».

```
exstaff:A007 :direccion [ :calle "Almansa"; ...]
```

es la implementación en N3 de la sentencia en FOL:

$$(\exists d)(\text{direccion}(\text{A007}, d) \wedge \text{calle}(d, \text{Almansa}) \wedge \dots)$$

Una sentencia puede tener bnodes como sujeto y objeto:

```
[ :nombrepila "Pepita." ] :direccion [ :calle "Almansa" ]
```

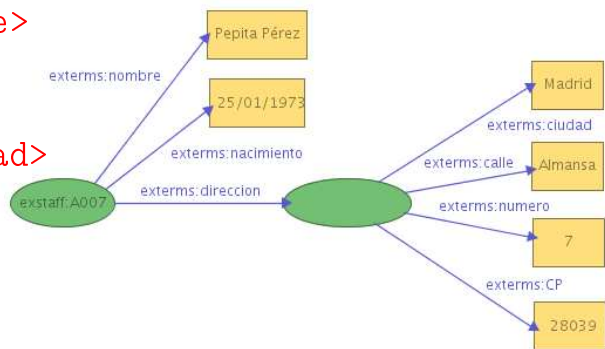
implementa la sentencia en FOL:

$$(\exists x)(\exists y)(\text{np}(x, \text{Pepita}) \wedge \text{direccion}(x, y) \wedge \text{calle}(y, \text{Almansa}))$$

bnodes en XML: forma «natural»

Elemento `rdf:Description` sin `rdf:about` ni `rdf:ID`

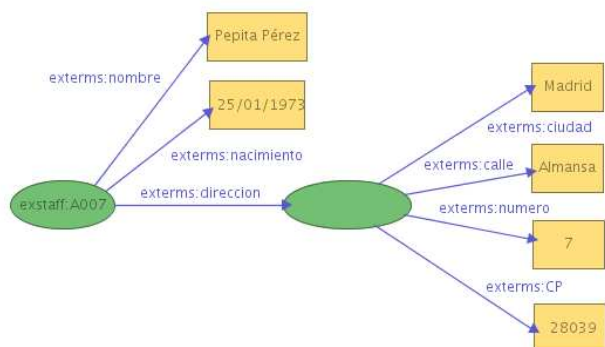
```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns="http://www.example.org/terms#">
  <rdf:Description rdf:about="http://www.example.org/staffid/A007">
    <nombre>Pepita Pérez</nombre>
    <nacimiento>25/01/1973</nacimiento>
    <direccion>
      <rdf:Description>
        <calle>Almansa</calle>
        <numero>7</numero>
        <CP>28039</CP>
        <ciudad>Madrid</ciudad>
      </rdf:Description>
    </direccion>
  </rdf:Description>
</rdf:RDF>
```



bnodes en XML: forma abreviada con `rdf:parseType="Resource"`

Se omite el `rdf:Description` del bnode

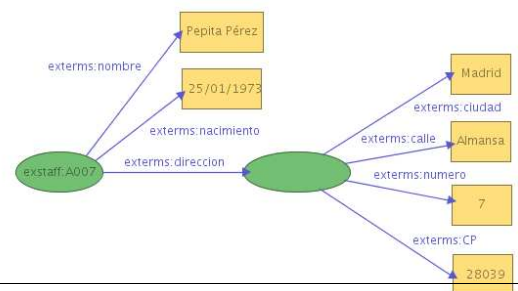
```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns="http://www.example.org/terms#">
  <rdf:Description rdf:about="http://www.example.org/staffid/A007">
    <nombre>Pepita Pérez</nombre>
    <nacimiento>25/01/1973</nacimiento>
    <direccion rdf:parseType="Resource">
      <calle>Almansa</calle>
      <numero>7</numero>
      <CP>28039</CP>
      <ciudad>Madrid</ciudad>
    </direccion>
  </rdf:Description>
</rdf:RDF>
```



bnodes en XML: forma extendida con `rdf:nodeID="un_nombre"`

Se identifica el bnode con un nombre local

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
        xmlns="http://www.example.org/terms#">
  <rdf:Description rdf:about="http://www.example.org/staffid/A007">
    <nombre>Pepita Pérez</nombre>
    <nacimiento>25/01/1973</nacimiento>
    <direccion rdf:nodeID="b1"/>
  </rdf:Description>
  <rdf:Description rdf:nodeID="b1">
    <calle>Almansa</calle>
    <numero>7</numero>
    <CP>28039</CP>
    <ciudad>Madrid</ciudad>
  </rdf:Description>
</rdf:RDF>
```



Ejemplo: vCard en RDF

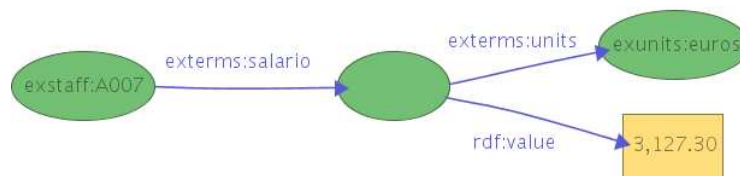
```
@prefix vCard: <http://www.w3.org/2001/vcard-rdf/3.0#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix gsi: <http://www.gsi.dit.upm.es/home/> .
gsi:gfer
  vCard:FN "Gregorio Fernandez" ;
  vCard:N [vCard:Family "Fernandez";
          vCard:Given "Gregorio";
          vCard:Prefix "Prof." ] ;
  vCard:TITLE "Ingeniero de Telecomunicacion" ;
  vCard:ROLE "Catedratico" ;
  vCard:ORG [vCard:Orgname
            "Universidad Politecnica de Madrid";
            vCard:Orgunit
            "Departamento de Ingenieria Telematica" ];
  vCard:TEL "+34 913367326".
#
#
gsi:jcg
  vCard:FN "Jose Carlos Cristobal" ;
#
gsi:mga
  vCard:FN "Mercedes Garijo" ;
#
```

Vocabulario vCard definido en RDFS. Un agente, teniendo en cuenta este vocabulario, «entiende» el significado de la triplas.

Valores y unidades: `rdf:value`

`rdf:value`: propiedad predefinida en RDF para señalar a un componente principal en la estructura. Por ejemplo, «salario»

- como literal: "3.127,30 euros"
- como recurso con propiedades "3.127,30" y "euros"



```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns="http://www.example.org/terms#">
  <rdf:Description rdf:about="http://www.example.org/staffid/A007">
    <salario rdf:parseType="Resource">
      <rdf:value>3.127,30</rdf:value>
      <units rdf:resource="http://www.example.org/units/euros"/>
    </salario>
  </rdf:Description>
</rdf:RDF>
```

Literales con tipo

`exterms:salario rdf:value "3.127,30"`

Pero «3.127,30» debe interpretarse como un número, no como una cadena.

Análogamente:

`exstaff:A007 exterms:nacimiento "25/01/1973"`

Se usa el sistema de tipos de XML Schema:

```
@prefix exstaff: <http://www.example.com/staffid/> .
@prefix exunits: <http://www.example.org/units/> .
@prefix rdf:    <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xsd:   <http://www.w3.org/2001/XMLSchema#> .
@prefix :     <http://www.example.org/terms#> .
exstaff:A007
  :nacimiento "1973-01-25"^^xsd:date ;
  :salario
    [ rdf:value "3,127.30"^^xsd:decimal ;
      :units exunits:euros
    ] .
```

Tipos de datos en RDF/XML

```
exstaff:A007
  :nacimiento "1973-01-25"^^xsd:date ;
  :salario
    [ rdf:value "3,127.30"^^xsd:decimal ;
      :units exunits:euros
    ] .
```

Traducción a RDF/XML:

```
<!DOCTYPE rdf:RDF [
  <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#">
]>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns="http://www.example.org/terms#">
  <rdf:Description rdf:about="http://www.example.com/staffid/A007/">
    <nacimiento rdf:datatype="&xsd:date">1973-01-25</nacimiento>
    <salario rdf:parseType="Resource">
      <rdf:value rdf:datatype="&xsd:decimal">3,127.30</rdf:value>
      <units rdf:resource="http://www.example.org/units/euros"/>
    </salario>
  </rdf:Description>
</rdf:RDF>
```

Recursos con tipo (pertenencia a una clase)

«a» en N3, «rdf:type» en XML (y en los grafos, y en N3):

Predicado predefinido en el vocabulario de RDF para expresar la pertenencia de un ejemplar a una clase (tipo)

- Sujeto (recurso): ejemplar de una clase
- Objeto: recurso que representa a esa clase

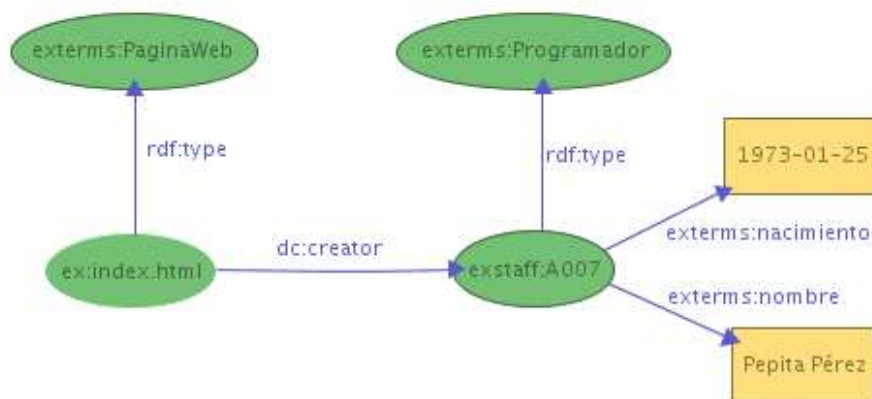
N3: ns1:nombreRec a ns2:NombreCl .

o bien: ns1:nombreRec rdf:type ns2:NombreCl .

RDF: <rdf:Description rdf:about="&ns1;nombreRec">
 <rdf:type rdf:resource="&ns2;NombreCl" />
 ...
</rdf:Description>

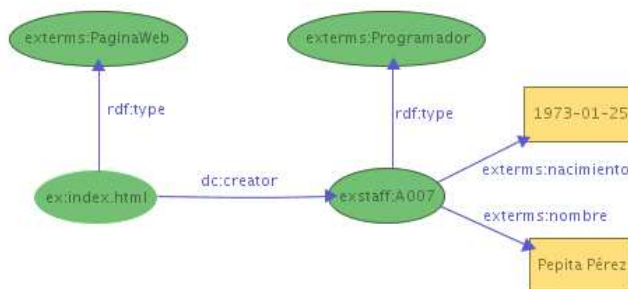
RDF abreviado (y habitual): <ns2:Clase rdf:about="&ns1;nombreRec">
 ...
</ns2:Clase>

Recursos con tipo: ejemplo en N3



```
ex:index.html a ex:PaginaWeb ;
              dc:creator ex:staff:A007 .
ex:staff:A007 a ex:Programador ;
              :nacimiento "1973-01-25"^^xsd:date ;
              :nombre "Pepita Pérez"^^xsd:string .
```

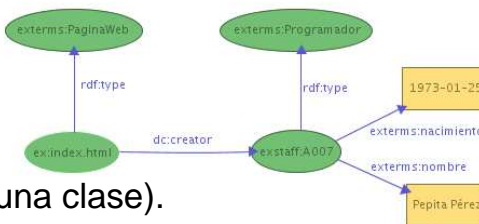
Recursos con tipo: ejemplo en RDF/XML



```
<rdf:Description rdf:about="http://www.example.org/index.html">
  <rdf:type rdf:resource="http://www.example.org/terms#PaginaWeb"/>
  <dc:creator rdf:resource="http://www.example.org/staffid/A007"/>
</rdf:Description>
<rdf:Description rdf:about="http://www.example.org/staffid/A007">
  <rdf:type rdf:resource="http://www.example.org/terms#Programador"/>
  <nombre rdf:datatype="&xsd:string">Pepita Pérez</nombre>
  <nacimiento rdf:datatype="&xsd:date">1973-01-25</nacimiento>
</rdf:Description>
```

Recursos con tipo: ejemplo en RDF/XML, forma abreviada

Se eliminan la propiedad `rdf:type` y su valor, y el elemento `rdf:Description` se sustituye por un elemento cuyo nombre es el qname correspondiente al valor de la propiedad `rdf:type` que se ha eliminado (una URIref de una clase).



```
<rdf:RDF xmlns="http://www.example.org/terms#"
  ...
  >
  <PaginaWeb rdf:about="http://www.example.org/index.html">
    <dc:creator rdf:resource="http://www.example.org/staffid/A007"/>
  </PaginaWeb>
  <Programador rdf:about="http://www.example.org/staffid/A007">
    <nombre rdf:datatype="&xsd:string">Pepita Pérez</nombre>
    <nacimiento rdf:datatype="&xsd:date">1973-01-25</nacimiento>
  </Programador>
  ...
</rdf:RDF>
```

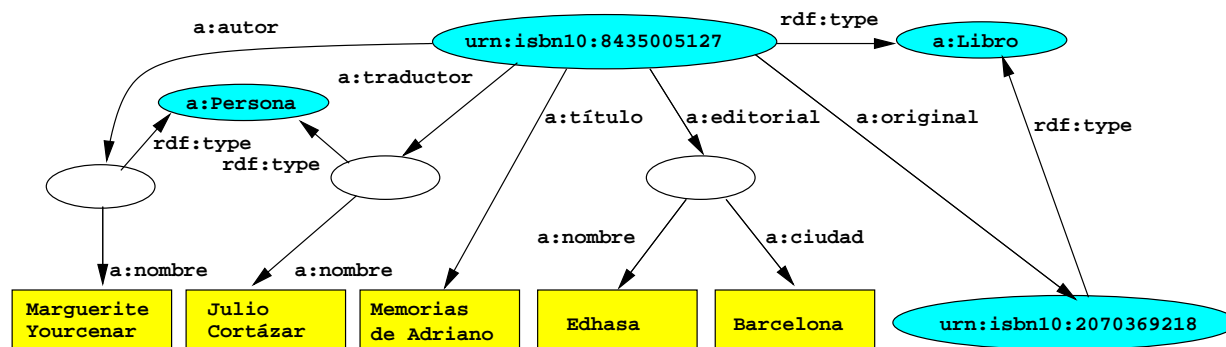
Fusión de datos distribuidos: ejemplo (1)

En la base de datos de la librería A:

Libros	ISBN10	Autor	Título	Traductor	Editorial	ISBN10-Orig
	8435005127	id_Yourcenar	Memorias de Adriano	id_Cortázar	id_Edh	2070369218

Personas	ID	Nombre	Editoriales	ID	Nombre	Ciudad
	id_Yourcenar	Marguerite Yourcenar		id_Edh	Edhasa	Barcelona
	id_Cortázar	Julio Cortázar				

Exportación a RDF (puede ser «al vuelo»):



Fusión de datos distribuidos: ejemplo (2)

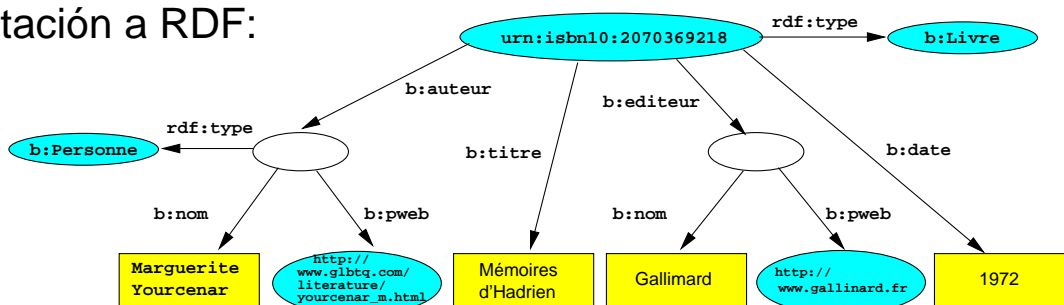
En la base de datos de la librería B:

Livres	ISBN10	Auteur	Titre	Editeur	Date
	2070369218	id_Your	Mémoires d'Hadrien	id_Gall	1972

Auteurs	ID	Nom	Page web
	id_Your	Marguerite Yourcenar	http://www.glbttq.com/literature/yourcenar_m.html

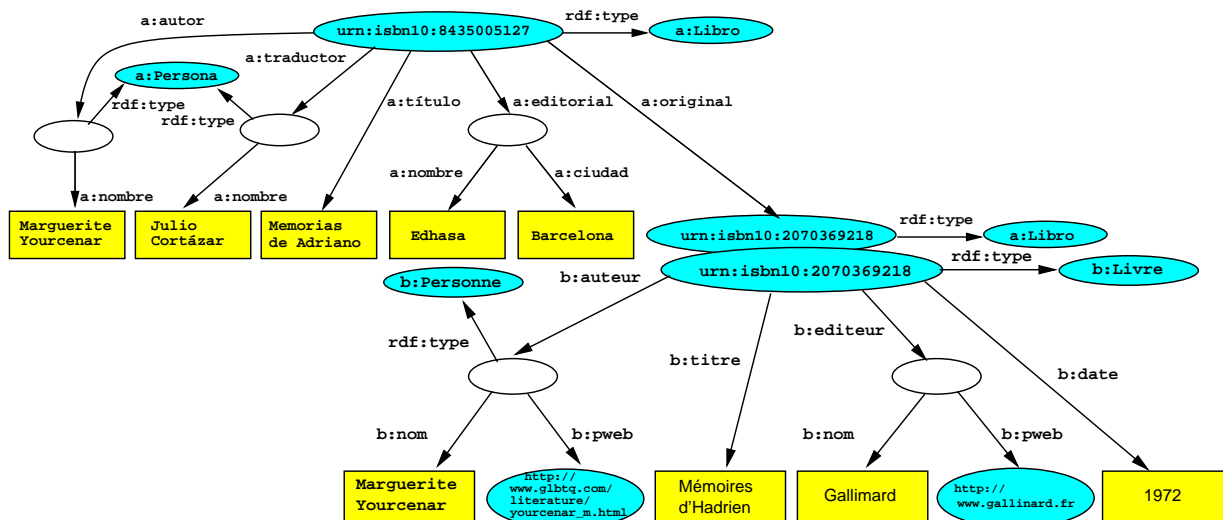
Editeurs	ID	Nom	Page web
	id_Gall	Gallimard	http://www.gallimard.fr

Exportación a RDF:



Fusión de datos distribuidos: ejemplo (3)

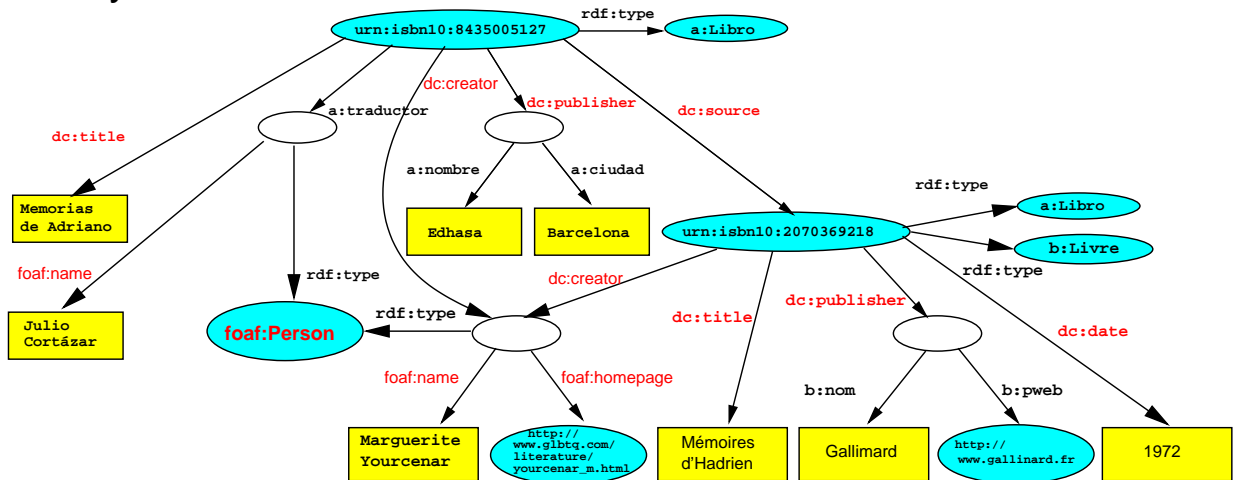
El mismo URI \rightsquigarrow el mismo recurso



Un usuario de la BD-A puede preguntar (si sabe idiomas...), por ejemplo, «díme el *titre* del original».

Fusión de datos distribuidos: ejemplo (4)

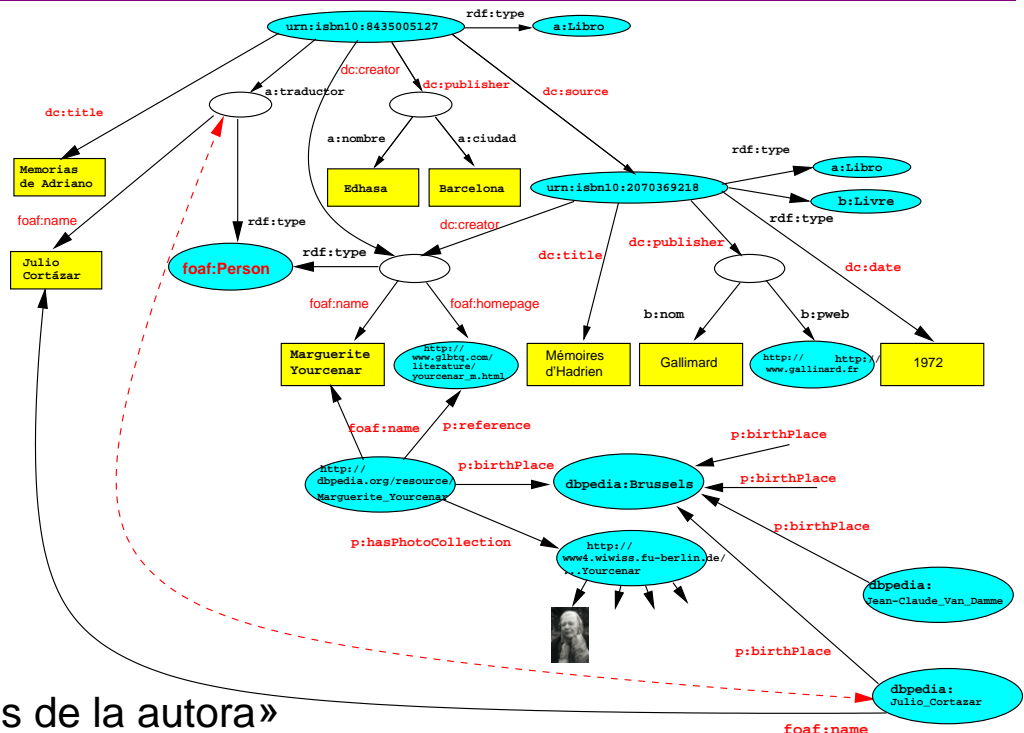
Si ambos sistemas utilizan vocabularios comunes, p. ej., Dublin Core y FOAF:



«Díme la página web del autor (autora)»...

Pero con el uso de una ontología como FOAF, un agente puede fusionar con muchas otras fuentes:

Más fusiones: dbpedia



«Fotografías de la autora»

«Personas que han nacido en la misma ciudad que la autora»

- **Contenedores:** clases y propiedades predefinidas en RDF para representar grupos de cosas, o *miembros* (recursos, bnodes o literales). Multiconjunto (`rdf:Bag`), secuencia (`rdf:Seq`) y alternativa (`rdf:Alt`).
- **Colecciones:** conjunto de miembros cerrado. Equivalente a una lista.
`rdf:List`, `rdf:first`, `rdf:rest`, `rdf:nil`
- **Cosificación** (reification): considerar como recurso a una sentencia (tripla).
`rdf:Statement`, `rdf:subject`, `rdf:predicate`, `rdf:object`

Ejemplo de aplicación de RDF: RSS 1.0

RSS: Rich Site Summary (RSS 0.91)
RDF Site Summary (RSS 0.9 y 1.0)
Really Simple Syndication (RSS 2.0)

Vocabularios para describir, distribuir y reusar información cambiante (noticias, bitácoras, agendas...)

- Las páginas web pueden hacer «sindicación» (redifusión de noticias)
- Agentes de usuario pueden leer, buscar y presentar de varias maneras los contenidos de sitios que estén descritos en RSS y hacer agregación de fuentes (*feeds*)
- RSS 2.0: el más sencillo, sólo para presentar enlaces. CNN, BBC, El País...
- Atom: IETF RFC 4287 (en desarrollo)
- RSS 1.0 usa RDF. Más adecuado para procesamiento por agentes

Ejemplo de RSS: <http://www.dit.upm.es/rss/2.html> (1)

```
<?xml version="1.0" encoding="iso-8859-1"?>
<rdf:RDF
  xmlns="http://purl.org/rss/1.0/"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:slash="http://purl.org/rss/1.0/modules/slash/"
  xmlns:dc="http://purl.org/dc/elements/1.1/">
  <channel rdf:about="http://www.dit.upm.es">
    <title>Noticias DIT-UPM</title>
    <description>Sistema de sindicación de Joomla!</description>
    <link>http://www.dit.upm.es</link>
    <image rdf:resource="http://www.dit.upm.es/images/M_images/joomla_rss.png" />
    <dc:date>2008-08-31T18:59:41+01:00</dc:date>
    <items>
      <rdf:Seq>
        <rdf:li rdf:resource="http://www.dit.upm.es/actividades/premio-al-desarrollo-de-a"
        <rdf:li rdf:resource="http://www.dit.upm.es/actividades/premios-a-tesis-doctorales"
      </rdf:Seq>
    </items>
  </channel>
  <image rdf:about="http://www.dit.upm.es/images/M_images/joomla_rss.png">
    <title>Potenciado por Joomla! | JoomlaSpanish premium pack!</title>
    <link>http://www.dit.upm.es</link>
    <url>http://www.dit.upm.es/images/M_images/joomla_rss.png</url>
  </image>
```

...



Ejemplo de RSS: <http://www.dit.upm.es/rss/2.html> (2)

Descripciones de los «items»:

```
<item rdf:about="http://www.dit.upm.es/actividades/premio-al-desarrollo-de-aplicacion"
  <dc:format>text/html</dc:format>
  <dc:date>2008-05-20T00:00:00+01:00</dc:date>
  <dc:source>http://www.dit.upm.es</dc:source>
  <title>Premio al desarrollo de aplicaciones móviles</title>
  <link>http://www.dit.upm.es/actividades/premio-al-desarrollo-de-aplicaciones-m-vile"
  <description>El equipo &ldquo;ALCE&rdquo;, formado por Carolina Garc&ia"
</item>

<item rdf:about="http://www.dit.upm.es/actividades/premios-a-tesis-doctorales-en-el-"
  <dc:format>text/html</dc:format>
  <dc:date>2008-05-20T12:34:26+01:00</dc:date>
  <dc:source>http://www.dit.upm.es</dc:source>
  <title>Premios a tesis doctorales en el DIT</title>
  <link>http://www.dit.upm.es/actividades/premios-a-tesis-doctorales-en-el-dit-3.html"
  <description>En la XXVIII Convocatoria de Premios &ldquo;Ingenieros de telecomu"
</item>
</rdf:RDF>
```



Píldora lógica: expresividad de RDF

- En RDF se pueden representar predicados monádicos y diádicos:
 - $\langle \text{Sujeto} \rangle \ \langle \text{Predicado} \rangle \ \langle \text{Objeto} \rangle \ . \ \text{es: } p(S, O)$
 - $\langle \text{Sujeto} \rangle \ a \ \langle \text{Clase} \rangle \ . \ \text{es: } c(S)$donde S y O son símbolos de *constantes*.
- Es decir, es **adecuado para la implementación de la «ABox»**: conjunto de *hechos* o *aserciones*
- Se pueden definir dominios concretos (mediante XML Schema)
- Lógica de orden superior por la cosificación:
 $\langle \text{Tripla} \rangle \ a \ \text{rdf:Statement} \ ;$
 $\text{rdf:subject } \langle S \rangle ; \text{rdf:predicate } \langle P \rangle ; \text{rdf:object } \langle O \rangle \ .$
 $\langle \text{Tripla} \rangle \ \langle p \text{ Sobre } T \rangle \ \langle O \text{ Sobre } T \rangle \ .$
es: $p \text{ Sobre } T(p(S, O), O \text{ Sobre } T)$
- *No se pueden definir vocabularios (ontologías)*: clases y subclases de recursos, propiedades, axiomas terminológicos...

Expresividad de RDF en términos prácticos

- Se pueden expresar relaciones binarias entre objetos:
X es el autor de Y, X tiene salario Z...
- Se puede expresar que los objetos pertenecen a clases:
X es un programador, Y es una página web...
- ¿Qué relaciones hay entre clases? La clase «Página web» es una subclase de la clase «Documento», «Autor» y «Programador» son subclases de «Persona»
- Los objetos heredan propiedades de las clases. ¿Cómo se definen estas propiedades, y a qué clases pueden aplicarse?