

Introducción a la Web Semántica

Jose Emilio Labra Gayo

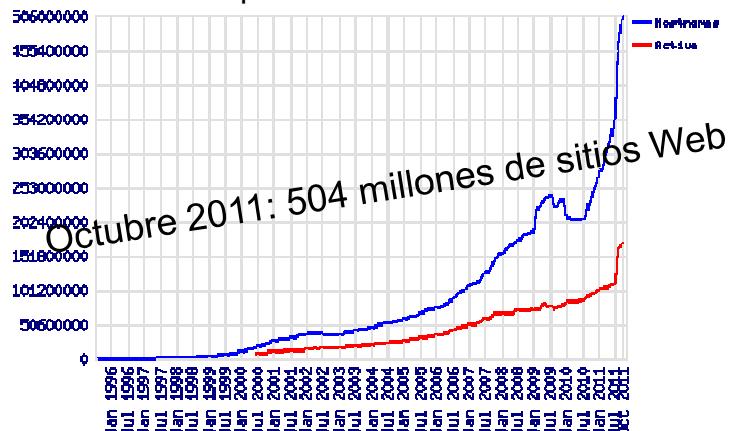
Departamento de Informática
Universidad de Oviedo

----- Departamento de Informática, UTFSM ----- 2012/2013



Evolución de la Web

Crecimiento casi exponencial



Fuente: Netcraft webserver survey

----- Departamento de Informática, UTFSM ----- 2012/2013

Datos multimedia

Dispositivos + baratos: Cámaras, móviles, ...

Facebook: 15.000 mill. fotos (2009)

Youtube: 144 mill. de vídeos (2009)

Flickr: 5.000 mill. fotos (2010)

y más.....

http://www.facebook.com/note.php?note_id=7639154919 | http://www.flickrpop.com/pop/flickr_interesting.html | <http://beernut.net/2008/03/14/how-to-find-out-the-number-of-friends-on-youtube/>

Internet de las cosas

Cada vez más dispositivos estarán conectados a Internet
Frigoríficos, coches, ...



Frigorífico con Internet

----- Departamento de Informática, UTFSM ----- 2012/2013

Y lo que falta...

Aumento de sensores y generadores de datos

Ejemplo: Acelerador de partículas LHC producirá 15 petabytes de datos/año



http://www.youtube.com/watch?v=sfEbMV295Kk&feature=player_embedded

----- Departamento de Informática, UTFSM ----- 2012/2013

Aún así...

Muchos datos no se están publicando
Reticencias para publicar datos
Razones para liberar datos
Incluso **exigir** datos abiertos



----- Departamento de Informática, UTFSM ----- 2012/2013

Razones para liberar datos

Facilitan la investigación
Tasa de descubrimiento se
acelera con mejores accesos a
los datos
Por el bien común de la
humanidad



----- Departamento de Informática, UTFSM ----- 2012/2013

Razones para liberar datos

Los sistemas abiertos facilitan las contribuciones externas



----- Departamento de Informática, UTFSM ----- 2012/2013

Razones para liberar datos

Eficiencia y calidad de los sistemas

Datos públicos no disponibles perjudican el desarrollo de sistemas

Ej. ¿Lista de municipios?



----- Departamento de Informática, UTFSM ----- 2012/2013

Razones para liberar datos

Trasparencia

- Fomentar participación
- Generar confianza
- Evaluar al gobierno



----- Departamento de Informática, UTFSM ----- 2012/2013

Razones para liberar datos

NO
IMAGE
AVAILABLE

Esta presentación hubiera sido muchísimo más aburrida si no hubiese tenido acceso a los datos parcial o totalmente abiertos de Google, Flickr, Wikipedia, Slideshare, etc. etc. etc.

----- Departamento de Informática, UTFSM ----- 2012/2013

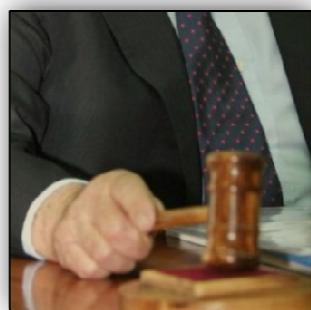


Si realmente quieres algo...

...déjalo libre

Como ciudadanos...

...también podemos demandar datos abiertos...



...demandar datos abiertos

Cuando pertenecen a la humanidad



----- Departamento de Informática, UTFSM ----- 2012/2013

...demandar datos abiertos

Hechos independientes y verificables ó de conocimiento común

Ejemplo: conocimiento científico

A chalkboard covered in handwritten mathematical equations, likely from a physics or mathematics class. The equations include various symbols like γ , α , β , and λ , and terms involving t , x , and y . The board is a visual representation of independent and verifiable scientific knowledge.

----- Departamento de Informática, UTFSM ----- 2012/2013

...demandar datos abiertos

Cuando han sido creados con
dinero público
Los hemos pagado con nuestros
impuestos
¡Son nuestros!



----- Departamento de Informática, UTFSM ----- 2012/2013



Universidad Técnica Federico Santa María
Departamento de Informática



OK, ¡vivan los datos abiertos!
pero...



¿Cómo publicarlos?

----- Departamento de Informática, UTFSM ----- 2012/2013

No basta con publicar datos...

El mayor reto = Integración

En general, el problema *no es informatizar* algo

El problema es **integrar** los sistemas

Interoperabilidad



----- Departamento de Informática, UTFSM ----- 2012/2013

Modelo de Estrellas*

- ★ **Publicar** los datos
(en cualquier formato)
- ★★ Utilizar **formato estructurado**
(Excel en lugar de imágenes escaneadas)
- ★★★ Usar formatos **no propietarios**
(CSV en lugar de Excel)
- ★★★★ Usar **URIs para identificar** datos
(otros sistemas puedan enlazar nuestros datos)
- ★★★★★ **Enlazar con otros** datos externos
(proporcionar contexto)

* Enunciado por Tim Berners-Lee en Gov 2.0 Expo 2010
<http://www.youtube.com/watch?v=gtaSJCFe0>

----- Departamento de Informática, UTFSM ----- 2012/2013



Formatos no estructurados

Formatos “caja negra”: Imágenes, vídeos, música, etc.

Formatos binarios: PDF, PS, etc.

Requieren técnicas de tratamiento de la señal,
reconocimiento de patrones, etc.



..... Departamento de Informática, UTFSM 2012/2013



Ejemplo: Servicio Público de Empleo

CONTRATOS DE TRABAJO REGISTRADOS SEGÚN SEXO Y SECTOR DE ACTIVIDAD ECONÓMICA												
ASTURIAS		TOTAL	TIPO DE CONTRATO						SECTORES			
			HOMBRES			MUJERES			AGRICULT.	INDUSTRIA	CONSTRUC.	
			INIC. INDEF.	INIC. TEMPORAL	CONVERT. INDEF.	INIC. INDEF.	INIC. TEMPORAL	CONVERT. INDEF.				
ALLANDE		31		18		3	10		3		9	16
ALLER		84	1	49	3	1	29	1		13	19	52
AMIEVA		7		3			4		1			6
AVILES		1.816	54	790	48	28	858	38	34	248	173	1.361
BELMONTE DE MIRANDA		25		3			22		1	1		23
BIMENES		15		7	1		7	0		3	5	7
BOAL		6		2		1	3					6
CABRALES		20		6	2	2	10	0		1	3	16
CABRANES		10	1	1		1	6	1		1		9

http://www.sepe.es/contenidos/cifras/datos_estadisticos/municipios/



..... Departamento de Informática, UTFSM 2012/2013



Formatos estructurados

Los datos tienen una estructura

Ejemplo: Hojas de cálculo

Problema con formatos propietarios

Requieren herramientas que no son públicas



Departamento de Informática, UTFSM 2012/2013



Ejemplo: Servicio Público de empleo

Portapapeles		Fuente	Alineación	Número	Estilos	Celdas	Modificar
B1	B2	B3	B4	B5	B6	B7	B8
CONTRATOS DE TRABAJO REGISTRADOS SEGÚN SEXO Y SECTOR DE ACTIVIDAD ECONÓMICA							
ASTURIAS							
MUNICIPIOS	TOTAL	TIPO DE CONTRATO				SECTORES	
		HOMBRES	MUJERES			AGRICULT.	INDUSTRIA
		INC. INDEF.	INC. INDEF.	CONVENT.	INC. INDEF.	INC. INDEF.	CONVENT. INDEF.
1 ALTAVERA	25	15	5	3	31	3	3
2 ALLEN	44	11	4	3	31	13	15
3 ALMERIA	7	5	2	4	4	1	6
4 ALONSO DE PALACIOS	186	54	70	45	285	56	244
5 ALQUINTAN Y ARANDA	15	7	1	2	1	1	1
6 BIMENDE	4	2	1	1	6	2	2
7 BES	65	35	20	15	90	1	5
8 BORRAS	16	5	2	1	6	1	5
9 CARBANES	16	5	1	1	6	1	5
10 CARRIZAL	4	2	1	1	4	1	3
11 CANGAS DE ONIS	55	20	15	41	4	1	14
12 CANGAS DEL NARCEA	135	50	50	72	45	4	35
13 CANTABRIA	4	2	1	1	4	1	2
14 CANTERA	111	100	45	56	144	4	35
15 CASO	5	2	1	2	4	1	2
16 CARRIZUELA	164	51	51	51	65	5	20
17 CARRIZUELA	44	15	15	24	55	5	15
18 CARRIZUELA	21	11	4	9	10	1	6
19 CARRIZUELA	59	25	25	25	70	7	45
20 CANTERA DE ASTURIAS	55	155	121	156	176	7	124
21 CARRIZUELA	64	15	50	22	35	1	25
22 CARRIZUELA	22	20	20	2	2	1	1
23 CARRIZUELA	30	15	15	14	34	3	16
24 GUON	1.624	110	2.204	93	1.63	2.342	146
25 GUON	92	52	54	1	55	4	12
26 GUON	95	25	25	4	2	42	6



http://www.sepe.es/contenidos/cifras/datos_estadisticos/municipios/9

Departamento de Informática, UTFSM 2012/2013



Formatos no propietarios

Utilizar formatos abiertos estructurados

Ejemplos: CSV, HTML

Problema: Contenido depende del contexto

..... Departamento de Informática, UTESM 2012/2013



CSV

“Comma separated values” valores delimitados por comas

Departamento de Informática, UTFSM 2012/2013



HTML

HTML pensado para representar información que se visualiza en el navegador

El procesamiento puede requerir “screen scrapping”

..... Departamento de Informática, UTFSM 2012/2013



URLs para identificar datos

Utilizar una URI para identificar un dato

Diferentes representaciones para cada tipo de dato

Negociación de contenido



Departamento de Informática, UTFSM 2012/2013



Negociación de contenido

El protocolo HTTP permite al cliente informar al servidor qué tipo de contenido prefiere

Servidor puede devolver representaciones diferentes según preferencias del cliente

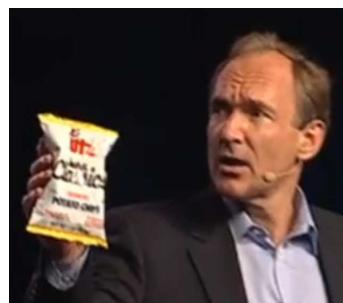


----- Departamento de Informática, UTFSM ----- 2012/2013



¿Varias representaciones para lo mismo?

Ejemplo: Códigos de barras



----- Departamento de Informática, UTFSM ----- 2012/2013



XML

XML permite representar información estructurada

Los documentos pueden validarse (XML Schema)

El significado de las etiquetas depende de la aplicación

```
<?xml version="1.0" ?>
<informe>
  <empleado año="2009" mes="01">
    <municipio nombre="Allande">
      <total>31</total>
      <sectores>
        <agricultura>13</agricultura>
        <industria>2</industria>
        <construcción>4</construcción>
        <servicios>2</servicios>
      </sectores>
    </municipio>
  </empleado>
  <empleado año="2009" mes="01">
    <municipio nombre="Oviedo">
      <total>3456</total>
      <sectores>
        <agricultura>2000</agricultura>
        <industria>410</industria>
        <construcción>40</construcción>
        <servicios>1006</servicios>
      </sectores>
    </municipio>
  </empleado>
```

----- Departamento de Informática, UTFSM ----- 2012/2013



Linked Open Data

Identificar datos mediante URIs

Usar URIs *dereferenciables*

Dar información útil al dereferenciar una URI

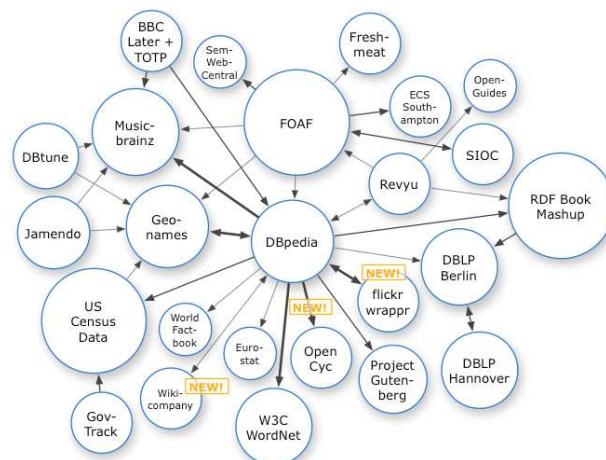
Enlazar con otras URIs



----- Departamento de Informática, UTFSM ----- 2012/2013



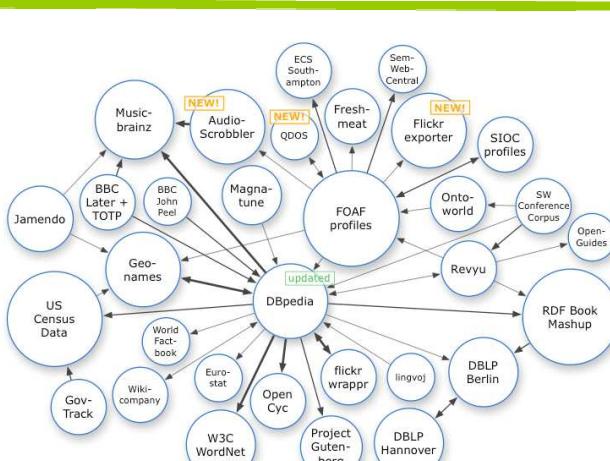
Linking Open Data (2007)



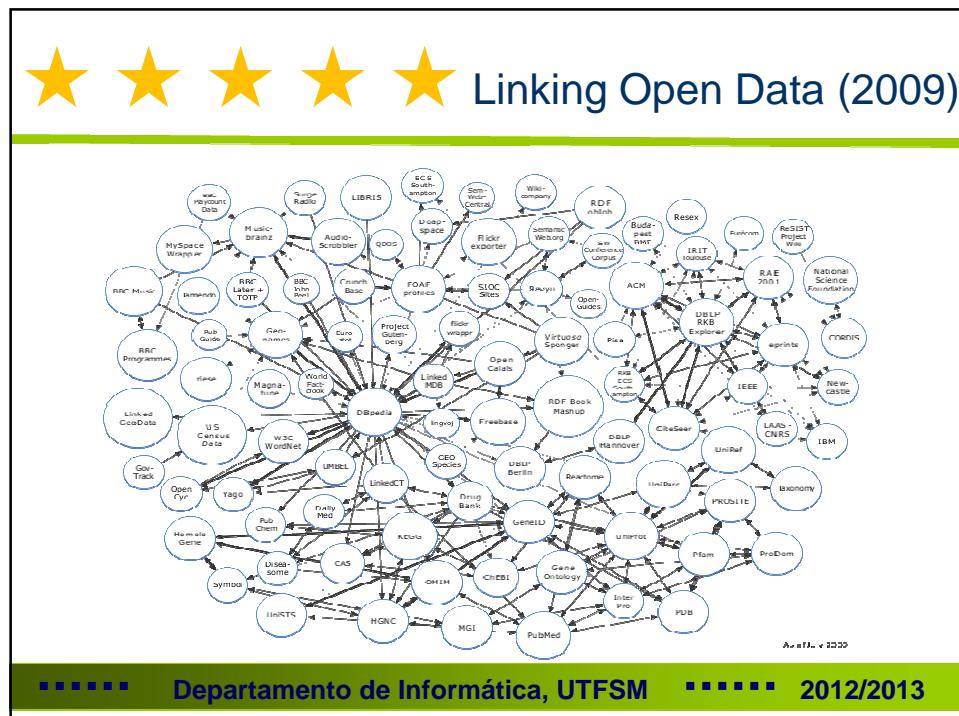
----- Departamento de Informática, UTSFSM ----- 2012/2013



Linking Open Data (2008)



----- Departamento de Informática, UTSFSM ----- 2012/2013



Web sintáctica

Web actual = Web sintáctica

Web de documentos

Normalmente representados en HTML

Enlaces entre documentos mediante `...`

Enlaces sin significado

El usuario no quiere documentos, quiere datos

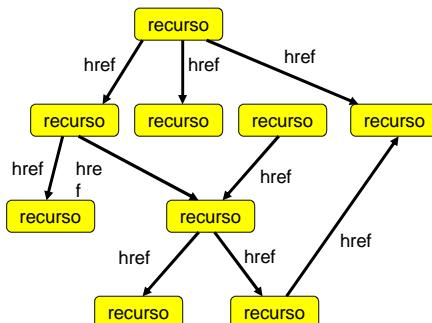
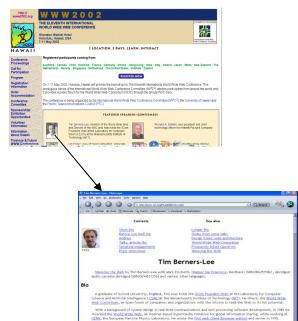
Ejemplo:

Teléfono de Juan vs Página web de Juan

----- Departamento de Informática, UTFSM ----- 2012/2013

Web sintáctica

Recursos enlazados entre sí (Grafo dirigido)



Ordenadores realizan la presentación visual (tarea fácil)

Personas navegan e interpretan el contenido (tarea difícil)

¿Sería posible que los ordenadores hiciesen algo más?

----- Departamento de Informática, UTFSM ----- 2012/2013

Problemas de la Web Sintáctica

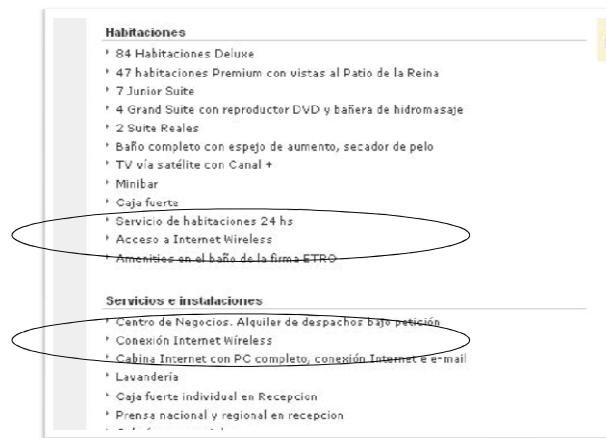
Ejemplos de tareas difíciles en la Web
Sintáctica

----- Departamento de Informática, UTFSM ----- 2012/2013

Representar información estructurada

Ejemplo 1:

Buscar un hotel con wi-fi gratuito en una ciudad



Habitaciones

- ✓ 84 Habitaciones Deluxe
- ✓ 47 habitaciones Premium con vistas al Patio de la Reina
- ✓ 7 Junior Suite
- ✓ 4 Grand Suite con reproductor DVD y bañera de hidromasaje
- ✓ 2 Suite Reales
- ✓ Baño completo con espejo de aumento, secador de pelo
- ✓ TV vía satélite con Canal +
- ✓ Minibar
- ✓ Caja fuerte
- ✓ Servicio de habitaciones 24 hs
- ✓ Acceso a Internet Wireless
- ✓ Amenidades en el baño de la firma STRO

Servicios e instalaciones

- ✓ Centro de Negocios. Alquiler de despachos bajo petición
- ✓ Conexión Internet Wireless
- ✓ Cabin Internet con PC completo, conexión Internet e e-mail
- ✓ Lavandería
- ✓ Caja fuerte individual en Recepción
- ✓ Prensa nacional y regional en recepción

----- Departamento de Informática, UTFSM ----- 2012/2013

Representar información estructurada

Ejemplo 2:

Organizar un viaje (Valladolid - Lanzarote?)



----- Departamento de Informática, UTFSM ----- 2012/2013

Material Multimedia

Ejemplos:

Fotos/vídeos con ciertas características

Información sobre un cuadro

Canciones



¿Otras obras del mismo autor?

----- Departamento de Informática, UTFSM ----- 2012/2013

Tareas difíciles en la Web Sintáctica

Buscar información sobre la Universidad de **Beihang** en China...

..... Departamento de Informática, UTFSM 2012/2013

Tareas difíciles en la Web Sintáctica

Otras tareas:

¿Alguien que se autoproclamó rey de los Estados Unidos?

¿Un pájaro que utilice el oído para orientarse y que no sea un murciélagos?

..... Departamento de Informática, UTFSM 2012/2013

Tareas difíciles en la Web Sintáctica

Búsquedas complejas
Información estructurada
Información multimedia: imágenes, vídeos, audio
Información en otros idiomas
Información imprecisa
Búsquedas conceptuales
Encontrar y utilizar “servicios web”
Delegar tareas complejas a agentes de la Web
Organizar viajes
Buscar y comparar noticias
Realizar encargos (ofertas, alertas, etc.)

----- Departamento de Informática, UTFSM ----- 2012/2013

Web Semántica

Propuesta de la **Web semántica** (Tim Berners-Lee):

“**disponer datos en la Web definidos y enlazados** de forma que puedan ser **utilizados por las máquinas** no solamente para visualizarlos sino también para:

automatizar tareas,
integrar y
reutilizar datos entre aplicaciones”

Web de datos en lugar de Web de documentos

----- Departamento de Informática, UTFSM ----- 2012/2013

Web Semántica

Características de la Web que deben tenerse en cuenta...

No centralizada: problemas para garantizar integridad de la información)

Información Dinámica: puede cambiar la información e incluso el conocimiento sobre esa información

Mucha información: El sistema no puede pretender acaparar toda la información

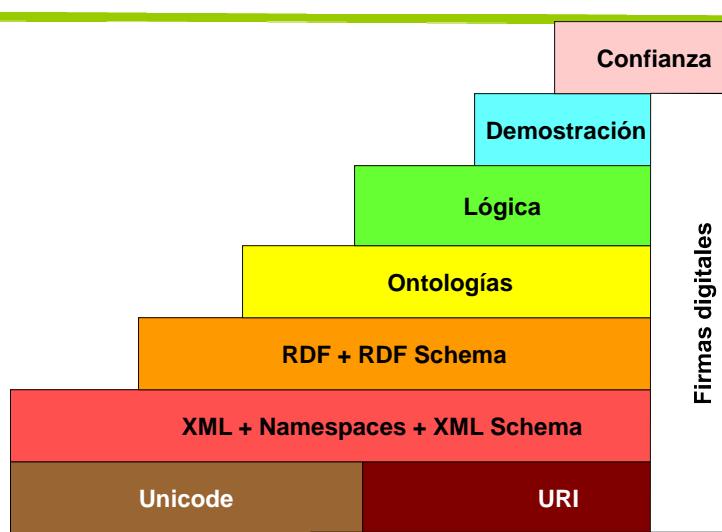
Es abierta: Muchos sistemas anteriores usaban la *Closed World Assumption*

Principio CCC
Cualquiera puede decir Cualquier cosa sobre Cualquier tema

En inglés: Principio AAA: Anyone can say Anything about Any topic
Fuente: Semantic Web for the Working Ontologist, D. Allemang, J. Hendler

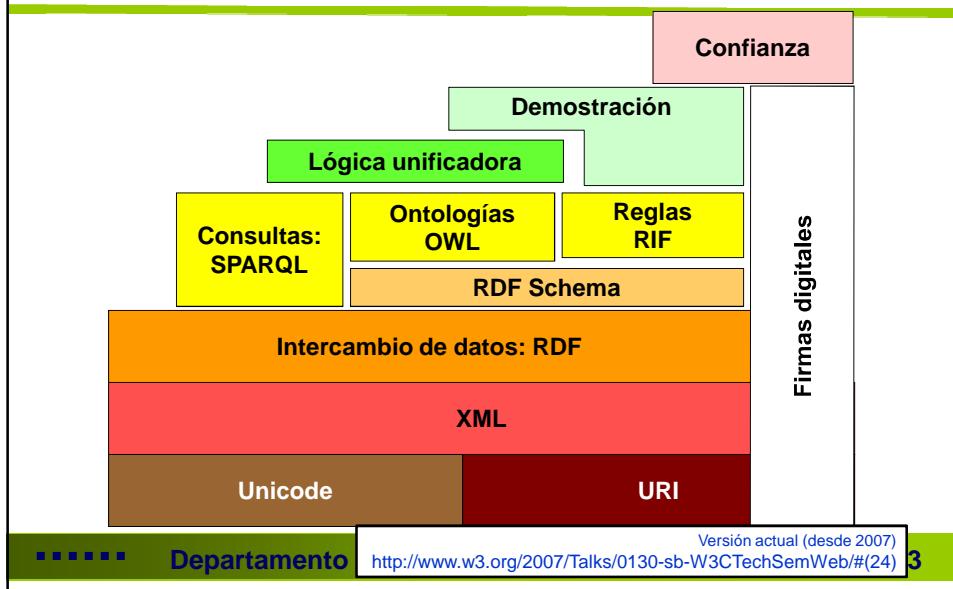
----- Departamento de Informática, UTFSM ----- 2012/2013

Tarta de la Web



----- Departamento de Informática, UTFSM ----- 2012/2013

Cambios en la tarta...



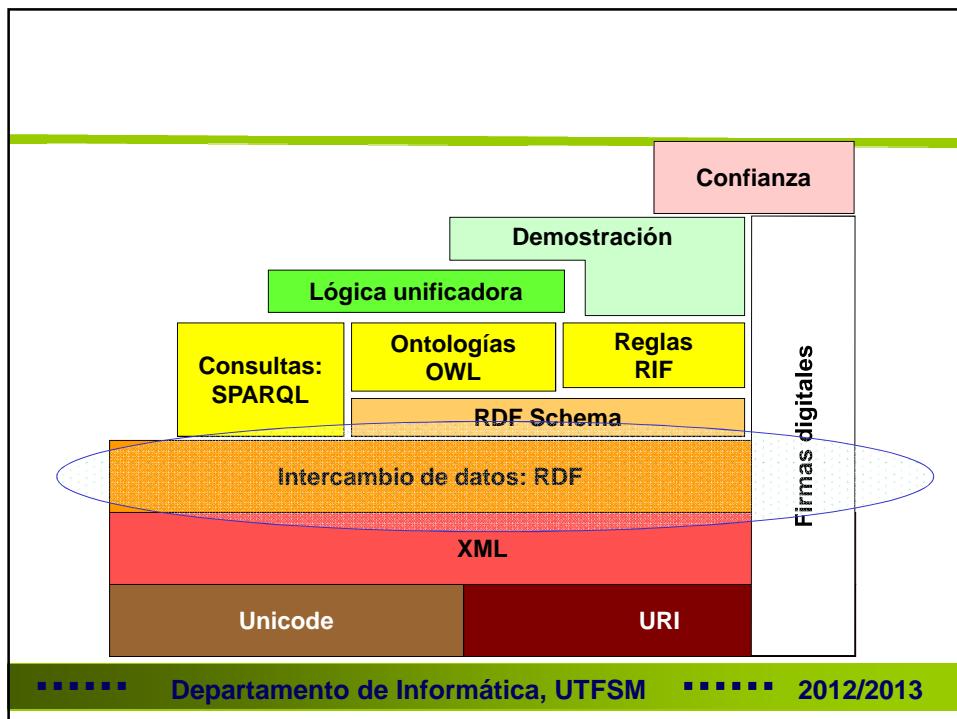
The slide features logos for the Magíster en Tecnologías de la Información (ITI) and the Universidad Técnica Federico Santa María (UTFSM).

Universidad Técnica Federico Santa María
Departamento de Informática

RDF

Jose Emilio Labra Gayo
Departamento de Informática
Universidad de Oviedo

----- Departamento de Informática, UTFSM ----- 2012/2013



RDF

Resource Description Framework (1998)

Descripción de recursos

Recurso = se identifica con URI

Tripletas: Sujeto → Predicado → Objeto



----- Departamento de Informática, UTFSM ----- 2012/2013



Triplets RDF

http://biology.uniovi.es

http://purl.org/dc/terms/creator

http://uniovi.es/people#Juan

Sujeto

Puede ser:

URI

Nodo anónimo (bNode)

Predicado

Identificado por URI

Objeto

Valor de una propiedad

Puede ser:

URI

Literal

Nodo anónimo

Abreviar URLs mediante espacios de nombres

Ejemplos:

dc: http://purl.org/dc/elements/1.1/

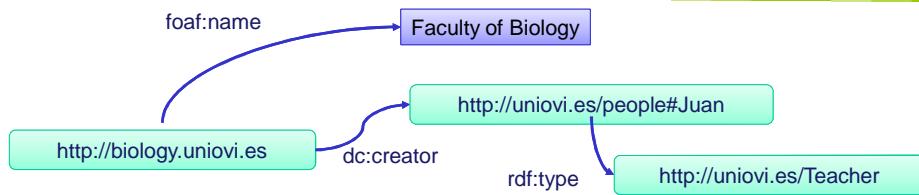
foaf: http://xmlns.com/foaf/0.1/

rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns#

----- Departamento de Informática, UTFSM ----- 2012/2013



Grafo RDF



Puede representarse en Turtle

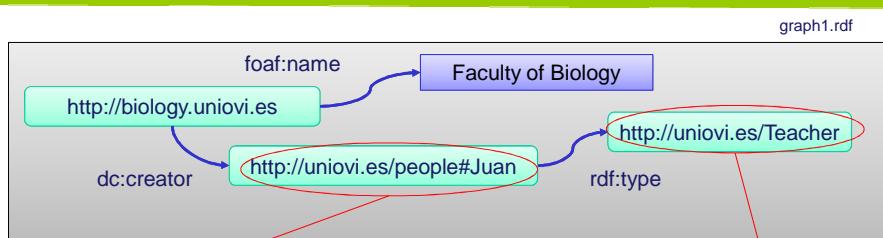
```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .  
@prefix foaf: <http://xmlns.com/foaf/0.1/> .  
@prefix dc: <http://purl.org/dc/terms/> .
```

```
<http://biology.uniovi.es>      dc:creator   <http://uniovi.es/people#Juan>.  
<http://biology.uniovi.es>      foaf:name    "Faculty of Biology".  
<http://uniovi.es/people#Juan>  rdf:type     <http://uniovi.es/Teacher> .
```

----- Departamento de Informática, UTFSM ----- 2012/2013



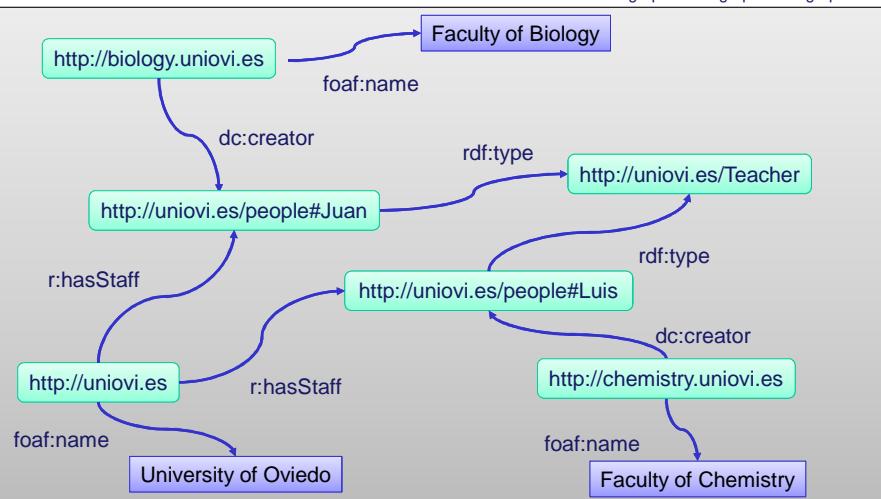
RDF es composicional





RDF es composicional

graph1.rdf + graph2.rdf+ graph3.rdf



..... Departamento de Informática, UTFSM 2012/2013



URIs y Espacios de nombres

Declarando espacios de nombres se facilita la declaración de URIs

@prefix x: <url> declara x como representante de url

Las referencias x:n equivalen a url:n

<> se refiere al documento actual

Puede declararse el espacio de nombres por defecto mediante

@prefix : <url>

jena.rdfcompare compara si 2 grafos son equivalentes

..... Departamento de Informática, UTFSM 2012/2013



Propiedad type

La propiedad type

<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>

declara el tipo al que pertenece un recurso

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.  
@prefix e: <http://www.ejemplo.org#> .  
  
e:Jose rdf:type e:Persona.  
e:Juan rdf:type e:Persona.
```

`rdf:type` puede simplificarse como a

```
@prefix e: <http://www.ejemplo.org#> .  
  
e:Jose a e:Persona.  
e:Juan a e:Persona.
```

----- Departamento de Informática, UTFSM ----- 2012/2013



Otros Espacios de nombres populares

Alias	URL	Nombre	Ejemplos
rdf:	http://www.w3.org/1999/02/22-rdf-syntax-ns#	RDF	type, subject, predicate, object,...
rdfs:	http://www.w3.org/2000/01/rdf-schema#	RDF Schema	domain, range Class, Property subClassOf,...
owl:	http://www.w3.org/2002/07/owl#	OWL Ontologías	intersectionOf unionOf, ...
dc:	http://purl.org/dc/elements/1.1/	Dublin Core	author, date, creator, ...
foaf	http://xmlns.com/foaf/01/	FOAF Friend of a Friend	name, knows, etc.
skos:	http://www.w3.org/2004/02/skos/core# http://www.w3.org/2008/05/skos#	SKOS Simple Knowledge Organization System	broader, narrower,

La página <http://prefix.cc> permite recuperar la URI del prefijo más habitual

----- Departamento de Informática, UTFSM ----- 2012/2013

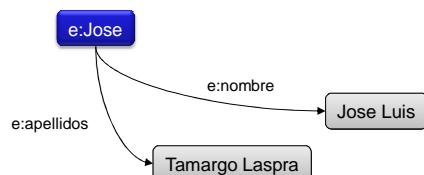


Literales

El valor puede ser una URI o un literal

```
@prefix e: <http://www.ejemplo.org#> .
```

```
e:Jose e:nombre "Jose Luis" .  
e:Jose e:apellidos "Tamargo Laspra" .
```



----- Departamento de Informática, UTSFSM ----- 2012/2013



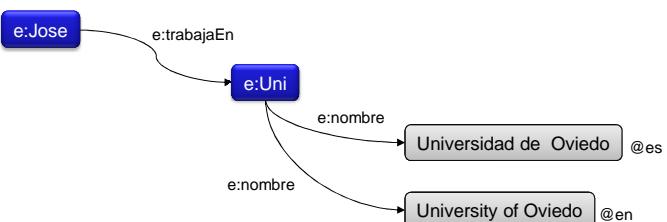
Literales con idioma

Es posible asociar un idioma al valor del literal

Sintaxis @idioma

```
@prefix e: <http://www.ejemplo.org#> .
```

```
e:Jose e:trabajaEn e:uni .  
e:uni e:nombre "Universidad de Oviedo" @es .  
e:uni e:nombre "University of Oviedo"@en .
```



----- Departamento de Informática, UTSFSM ----- 2012/2013



Literales con tipo

Se puede declarar el tipo de datos de un literal

Permite indicar cómo analizar el valor

```
@prefix e: <http://www.ejemplo.org#> .  
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.  
  
e:Jose e:edad "23"^^xsd:integer .
```

El grafo anterior es equivalente al siguiente

```
@prefix e: <http://www.ejemplo.org#> .  
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.  
  
e:Jose e:edad "0023"^^xsd:integer .
```

Simplificaciones en Turtle

```
true = "true"^^xsd:boolean  
3 = "3"^^xsd:integer  
4.2 = "4.2"^^xsd:decimal
```

----- Departamento de Informática, UTFSM ----- 2012/2013



Notación Turtle Simplificar descripciones (;

Mediante ; pueden declararse varias descripciones de propiedades a un recurso

```
e:Jose e:conoceA e:Luis ;  
e:esPadreDe e:Ana .
```

≡

```
e:Jose e:conoceA e:Luis .  
e:Jose e:esPadreDe e:Ana .
```

----- Departamento de Informática, UTFSM ----- 2012/2013



Notación Turtle Simplificar descripciones (,)

Mediante , pueden declararse varios valores para una propiedad de un recurso

e:Pepe e:conoceA e:Juan .
e:Pepe e:conoceA e:Luis .

≡

e:Pepe e:conoceA e:Juan , e:Luis.

----- Departamento de Informática, UTFSM ----- 2012/2013



Ejercicio

Simplificar el siguiente documento:

```
<http://www.ejemplo.org#Pepe> <http://www.ejemplo.org#conoceA> <http://www.ejemplo.org#Juan> .  
<http://www.ejemplo.org#Pepe> <http://www.ejemplo.org#conoceA> <http://www.ejemplo.org#Luis> .  
<http://www.ejemplo.org#Pepe> <http://www.ejemplo.org#conoceA> <http://www.ejemplo.org#Ana> .  
<http://www.ejemplo.org#Pepe> <http://www.ejemplo.org#esPadreDe> <http://www.ejemplo.org#Quique> .  
<http://www.ejemplo.org#Pepe> <http://www.ejemplo.org#esPadreDe> <http://www.ejemplo.org#Eva> .  
<http://www.ejemplo.org#Juan> <http://www.ejemplo.org#conoceA> <http://www.ejemplo.org#Luis> .  
<http://www.ejemplo.org#Juan> <http://www.ejemplo.org#esPadreDe> <http://www.ejemplo.org#Mar> .  
<http://www.ejemplo.org#Luis> <http://www.ejemplo.org#conoceA> <http://www.ejemplo.org#Ana> .  
<http://www.ejemplo.org#Luis> <http://www.ejemplo.org#conoceA> <http://www.ejemplo.org#Sandra> .
```

NOTA

jena.rdfcompare comparar si 2 grafos son equivalentes

----- Departamento de Informática, UTFSM ----- 2012/2013



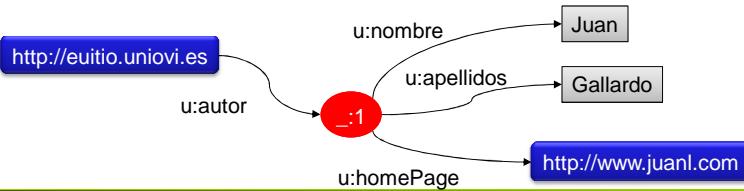
Nodos anónimos (blank nodes)

Los nodos blancos son nodos que no tienen asociada una URI

Permite hacer descripciones sobre elementos de los que no se conoce su URI

En N3 se identifican mediante _:identificador

```
<http://euitio.uniovi.es> u:autor _:1 .  
 _:1 u:nombre "Juan".  
 _:1 u:apellidos "Gallardo".  
 _:1 u:homePage <http://juan.com>
```



----- Departamento de Informática, UTFSM ----- 2012/2013



Nodos anónimos (blank nodes)

Puede haber varios nodos anónimos en una descripción

Cada nodo tendrá su propio identificador

Los identificadores de nodos anónimos son locales al contexto en el que se definen

grafo1.ttl

```
<http://uniovi.es> u:autor _:1 .  
 _:1 u:nombre "Juan".  
<http://unileon.es> u:autor _:2 .  
 _:2 u:nombre "Jose".
```

grafo2.ttl

```
<http://upc.es> u:autor _:1 .  
 _:1 u:nombre "Santi".
```

----- Departamento de Informática, UTFSM ----- 2012/2013



Ejercicio 1 (paginas)

Representar el siguiente conocimiento:

La página <http://www.uniovi.es> ha sido realizada por *Juan Gallardo*, el cual tiene por correo electrónico juan@uniovi.es y tiene 26 años.

Sin embargo, la página <http://www.euitio.uniovi.es> ha sido realizada por *Isabel Castilla* y *Juan Gallardo*. El correo electrónico de *Isabel* es isa@uniovi.es y tiene 25 años.

----- Departamento de Informática, UTFSM ----- 2012/2013

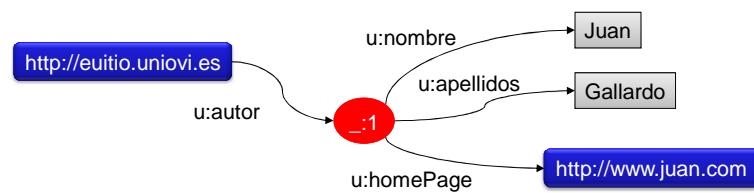


Nodos anónimos (blank nodes)

[] representa un nodo anónimo

Las declaraciones realizadas dentro de [] hacen referencia a dicho nodo anónimo.

```
<http://euitio.uniovi.es> u:autor
  [ u:nombre "Juan" ;
    u:apellidos "Gallardo";
    u:homePage <http://juan.com> ].
```



----- Departamento de Informática, UTFSM ----- 2012/2013



Ejercicio - Tabla

Representar información de una tabla

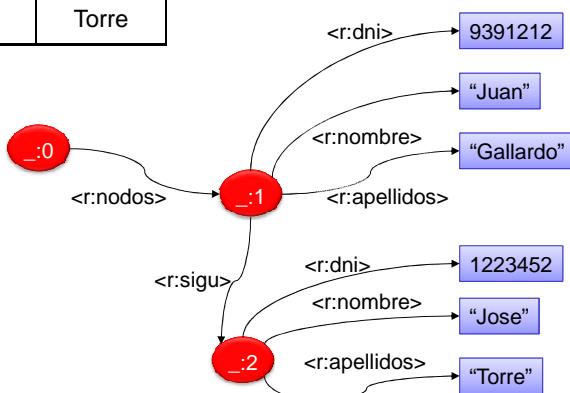
DNI	Nombre	Apellidos
9391212	Juan	Gallardo
1223452	Jose	Torre

----- Departamento de Informática, UTFSM ----- 2012/2013



Solución

DNI	Nombre	Apellidos
9391212	Juan	Gallardo
1223452	Jose	Torre



----- Departamento de Informática, UTFSM ----- 2012/2013



Ejercicio: Tabla con Motes

- Añadir motes a la tabla anterior.

Jose Torre también es conocido como "Pepe" y como "Pepín"

NOTA: En una tabla de bases de datos, requeriría celdas con valores múltiples y con valores nulos

DNI	Nombre	Apellidos	Mote
9391212	Juan	Gallardo	?
1223452	Jose	Torre	Pepe, Pepín

----- Departamento de Informática, UTFSM ----- 2012/2013



Ejercicio

Representar los siguientes grafos por separado y luego mezclarlos

@prefix r: <<http://ejemplo.org#>>.

_:1 r:dni 9999.
_:1 r:nombre "Juan".
_:1 r:esAmigoDe _:2 .

_:2 r:dni 8888 .
_:2 r:nombre "Jose".
_:2 r:esAmigoDe _:3 .

_:3 r:dni 7777 .

@prefix r: <<http://ejemplo.org#>>.

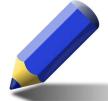
_:1 r:dni 7777.
_:1 r:nombre "Isabel" .
_:1 r:esAmigoDe _:2 .

_:2 r:dni 6666 .
_:2 r:nombre "Quique" .
_:2 r:esAmigoDe _:3 .

_:3 r:dni 9999 .

NOTA: Los nodos anónimos son locales

----- Departamento de Informática, UTFSM ----- 2012/2013



Ejercicio

¿Cuál de los siguientes grafos es equivalente a:

```
@prefix : <>.  
:a:p "1" .  
:a:p _:1 .  
:a:p _:2 .  
_:1 :q "A" .  
_:2 :r "B" .
```

```
@prefix : <>.  
:a:p "2" .  
:a:p _:1 .  
:a:p _:2 .  
_:1 :q "A" .  
_:2 :r "B" .
```

```
@prefix : <>.  
:a:p "1" .  
:a:p _:2 .  
:a:p _:1 .  
_:2 :q "A" .  
_:1 :r "B" .
```

```
@prefix : <>.  
:a:p "1" .  
:a:p _:1 .  
:a:p _:2 .  
_:1 :q "B" .  
_:2 :r "A" .
```

```
@prefix : <>.  
:a:p "1" .  
:a:p _:2 .  
:a:p _:2 .  
_:1 :q "B" .  
_:1 :r "A" .
```

----- Departamento de Informática, UTFSM ----- 2012/2013



Sintaxis RDF/XML

RDF/XML = Sintaxis XML para representar grafos RDF

```
@prefix e: <http://www.ejemplo.org#>.
```

```
e:Juan e:nombre "Juan".  
e:Juan e:apellidos "Gallardo".
```

```
<rdf:RDF  
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"  
    xmlns:e="http://www.ejemplo.org#">  
    <rdf:Description rdf:about="http://www.ejemplo.org#Juan">  
        <e:nombre>Juan</e:nombre>  
        <e:apellidos>Gallardo</e:apellidos>  
    </rdf:Description>  
</rdf:RDF>
```

----- Departamento de Informática, UTFSM ----- 2012/2013



Sintaxis RDF/XML

rdf:Description captura una o varias tripletas

```
@prefix e: <http://www.ejemplo.org#>.  
  
e:Juan e:nombre "Juan".  
e:Juan e:conoceA e:Pepe.  
  
e:Pepe e:nombre "Jose".
```

```
<rdf:RDF  
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"  
    xmlns:e="http://www.ejemplo.org#">  
    <rdf:Description rdf:about="http://www.ejemplo.org#Juan">  
        <e:conoceA rdf:resource="http://www.ejemplo.org#Pepe" />  
        <e:nombre>Juan</e:nombre>  
    </rdf:Description>  
    <rdf:Description rdf:about="http://www.ejemplo.org#Pepe">  
        <e:nombre>Jose</e:nombre>  
    </rdf:Description>  
</rdf:RDF>
```

***** Departamento de Informática, UTFSM ***** 2012/2013



Sintaxis RDF/XML

Modelo en cebolla

```
<rdf:RDF  
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"  
    xmlns:e="http://www.ejemplo.org#">  
    <rdf:Description rdf:about="http://www.ejemplo.org#Juan">  
        <e:conoceA>  
            <rdf:Description rdf:about="http://www.ejemplo.org#Pepe">  
                <e:nombre>Jose</e:nombre>  
            </rdf:Description>  
        </e:conoceA>  
        <e:nombre>Juan</e:nombre>  
    </rdf:Description>  
</rdf:RDF>
```

```
@prefix e: <http://www.ejemplo.org#>.  
  
e:Juan e:nombre "Juan".  
e:Juan e:conoceA e:Pepe.  
e:Pepe e:nombre "Jose".
```

***** Departamento de Informática, UTFSM ***** 2012/2013



Sintaxis RDF/XML

rdf:ID permite hacer referencia a un nodo local

Toma como base la URL del documento base

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:e="http://www.ejemplo.org#"
  xml:base="http://www.ejemplo.org#">
  <rdf:Description rdf:ID="Juan">
    <e:conoceA rdf:resource="http://www.ejemplo.org#Pepe" />
    <e:nombre>Juan</e:nombre>
  </rdf:Description>
  <rdf:Description rdf:ID="Pepe">
    <e:nombre>Jose</e:nombre>
  </rdf:Description>
</rdf:RDF>
```

----- Departamento de Informática, UTFSM ----- 2012/2013



Sintaxis RDF/XML

Varias reglas para simplificar las expresiones

```
<rdf:Description rdf:about="http://www.ejemplo.org#Juan">
  <rdf:type rdf:resource="http://www.ejemplo.org#Persona" />
  <e:nombre>Juan</e:nombre>
</rdf:Description>
```

La declaración de "type" puede incluirse en la etiqueta

```
<e:Persona rdf:about="http://www.ejemplo.org#Juan">
  <e:nombre>Juan</e:nombre>
</e:Persona>
```

Si las propiedades no se repiten, pueden incluirse como atributos

```
<e:Persona rdf:about="http://www.ejemplo.org#Juan" e:nombre="Juan" />
```

----- Departamento de Informática, UTFSM ----- 2012/2013



Nodos anónimos en RDF/XML

```
@prefix u: <http://uniovi.es#>
<http://euitio.uniovi.es> u:autor _:1 .
_:1 u:nombre "Juan".
_:1 u:apellidos "Gallardo".
_:1 u:homePage <http://juan.com>
```

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:u="http://uniovi.es#"
  <rdf:Description rdf:about="http://euitio.uniovi.es">
    <u:autor rdf:parseType="Resource">
      <u:homePage rdf:resource="http://juanlopez.com"/>
      <u:apellidos>Lopez</u:apellidos>
      <u:nombre>Juan</u:nombre>
    </u:autor>
  </rdf:Description>
</rdf:RDF>
```

013



RDF: Contenedores

Tipos de contenedores

Bag: Conjunto no ordenado (permite duplicados)
Seq: Lista ordenada (permite duplicados)
Alt: Valor único alternativo (elección de un elemento del contenedor)

Los elementos se indican con `<rdf:n>` ó con `<rdf:li>`

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:e="http://ejemplos.org#"
  <rdf:Description rdf:about="http://ejemplos.org#Logica">
    <e:tieneAlumnos>
      <rdf:Bag>
        <rdf:_1 rdf:resource="http://ejemplos.org#Juan"/>
        <rdf:_2 rdf:resource="http://ejemplos.org#Luis"/>
        <rdf:_3 rdf:resource="http://ejemplos.org#Marcos"/>
      </rdf:Bag>
    </e:tieneAlumnos>
  </rdf:Description>
</rdf:RDF>
```

Turtle

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix e: <http://ejemplos.org#>.
e:Logica e:tieneAlumnos _:1 .
_:1 a rdf:Bag .
_:1 rdf:_1 e:Juan .
_:1 rdf:_2 e:Luis .
_:1 rdf:_3 e:Marcos .
```

***** Departamento de



RDF: Colecciones

Listas de elementos permiten definir colecciones cerradas

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
           xmlns:e="http://www.ejemplos.org#">

<rdf:Description rdf:about="http://www.ejemplos.org#Logica">
  <e:tieneAlumnos rdf:parseType="Collection">
    <rdf:Description rdf:about="http://www.ejemplos.org#Juan" />
    <rdf:Description rdf:about="http://www.ejemplos.org#Luis" />
    <rdf:Description rdf:about="http://www.ejemplos.org#Marcos" />
  </e:tieneAlumnos>
</rdf:Description>
</rdf:RDF>
```

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix e: <http://www.ejemplos.org#>.
```

Turtle

```
e:Logica e:tieneAlumnos _:1 .
_:1 rdf:first e:Juan .
_:1 rdf:rest _:2 .
_:2 rdf:first e:Luis .
_:2 rdf:rest _:3 .
_:3 rdf:first e:Marcos .
_:3 rdf:rest rdf:nil .
```

En Turtle pueden simplificarse como:

```
e:Logica e:tieneAlumnos (e:Juan e:Luis e:Marcos).
```

..... Departamento de Informática, UTSMS 2012/2013

2012/2013



RDF: Reificación

Permite definir sentencias sobre sentencias (orden superior)

Ej. *El sitio Web de Uniovi dice que Labra es el profesor de Lógica*

Las sentencias se representan con el tipo predefinido `rdf:Statement`

Los atributos de `rdf:Statement` son: `rdf:subject`, `rdf:predicate` y `rdf:object`

Es posible añadir otros atributos a las sentencias

```
@prefix e: <http://www.ejemplos.org#>.
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.

e:uniovi  e:dice      e:d1 .
e:d1       a          rdf:Statement .
e:d1       rdf:subject   e:Labra .
e:d1       rdf:predicate e:esProfesorDe .
e:d1       rdf:object     e:Logica .
```

..... Departamento de Informática, UTSMS 2012/2013



RDF: Reificación

Sintaxis RDF/XML

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:e="http://www.ejemplos.org#">
  <rdf:Description rdf:about="http://www.ejemplos.org#EUITIO">
    <e:dice>
      <rdf:Statement rdf:about="http://www.ejemplos.org#d1">
        <rdf:subject rdf:resource="http://www.ejemplos.org#Labra"/>
        <rdf:predicate rdf:resource="http://www.ejemplos.org#esProfesorDe"/>
        <rdf:object rdf:resource="http://www.ejemplos.org#Logica"/>
      </rdf:Statement>
    </e:dice>
  </rdf:Description>
</rdf:RDF>
```

e:EUITIO	e:dice	e:d1 .
e:d1	a	rdf:Statement .
e:d1	rdf:subject	e:Labra .
e:d1	rdf:predicate	e:esProfesorDe .
e:d1	rdf:object	e:Logica .

----- Departamento de Informática, UTFSM ----- 2012/2013



RDF: Reificación

Es posible añadir más información al enunciado reificado

Ejemplo:

La EUITIO declara en 2008 que Labra es profesor de Lógica.

e:EUITIO	e:dice	e:d1 .
e:d1	e:fecha	2008.
e:d1	a	rdf:Statement .
e:d1	rdf:subject	e:Labra .
e:d1	rdf:predicate	e:esProfesorDe .
e:d1	rdf:object	e:Logica .

----- Departamento de Informática, UTFSM ----- 2012/2013



RDF: Tipos de Datos

RDF/XML permite declarar tipos de datos

En general se utilizan los tipos de XML Schema

Podrían utilizarse otros tipos de datos

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:e="http://www.ejemplo.org#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#>
  <rdf:Description rdf:about="http://www.ejemplo.org#Pepe">
    <e:edad rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">23</e:edad>
  </rdf:Description>
</rdf:RDF>
```

TRUCO: Se puede ahorrar escribir la URI entera declarando una entidad

```
<!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:e="http://www.ejemplo.org#"
           <rdf:Description rdf:about="http://www.ejemplo.org#Pepe">
             <e:edad rdf:datatype="&xsd;integer">23</e:edad>
           </rdf:Description>
</rdf:RDF>
```

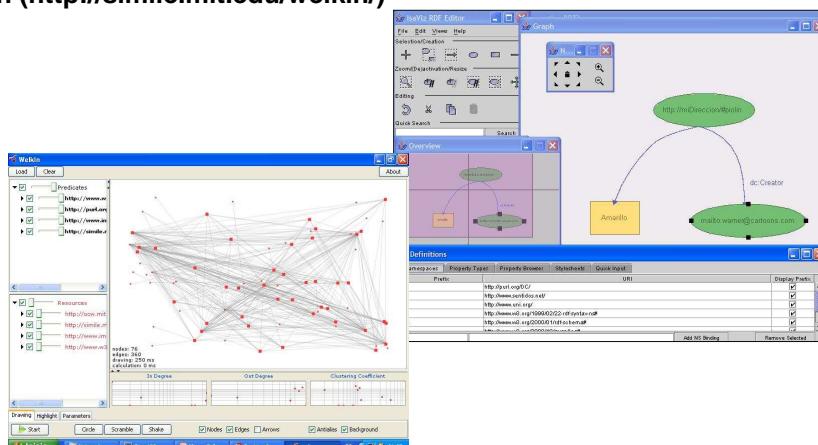
013



Herramientas para visualizar RDF

IsaViz (<http://www.w3.org/2001/11/IsaViz>)

Welkin (<http://simile.mit.edu/welkin/>)



----- Departamento de Informática, UTFSM ----- 2012/2013

Aplicaciones de RDF

----- Departamento de Informática, UTFSM ----- 2012/2013

Aplicaciones de RDF: RSS

RSS 1.0 es un vocabulario de RDF

Creación de resúmenes de sitios Web (syndication)

NOTA: Existe RSS 0.92, 0.93 y 2.0 que no se basa en RDF

```
<rdf:RDF>
  - <channel rdf:about="http://www.webweavertech.com/ovidiu/weblog">
    <title>Ovidiu Predescu's Weblog</title>
    <link>http://www.webweavertech.com/ovidiu/weblog/</link>
    <description>Technology ramblings</description>
    <dc:language>en-us</dc:language>
    <dc:creator>ovidiu</dc:creator>
    <dc:date>2004-03-02T20:17:36-08:00</dc:date>
    <admin:generatorAgent rdf:resource="http://www.movabletype.org/?n=2.64"/>
    <admin:errorReportsTo rdf:resource="mailto:ovidiu@webweavertech.com"/>
    <sy:updatePeriod>hourly</sy:updatePeriod>
    <sy:updateFrequency>1</sy:updateFrequency>
    <sy:updateBase>2000-01-01T12:00+00:00</sy:updateBase>
    - <items>
      - <rdf:Seq>
        <rdf:li rdf:resource="http://www.webweavertech.com/ovidiu/weblog/archives/000318.html"/>
        <rdf:li rdf:resource="http://www.webweavertech.com/ovidiu/weblog/archives/000317.html"/>
        <rdf:li rdf:resource="http://www.webweavertech.com/ovidiu/weblog/archives/000316.html"/>
        <rdf:li rdf:resource="http://www.webweavertech.com/ovidiu/weblog/archives/000314.html"/>
        <rdf:li rdf:resource="http://www.webweavertech.com/ovidiu/weblog/archives/000307.html"/>
    </rdf:Seq>
```

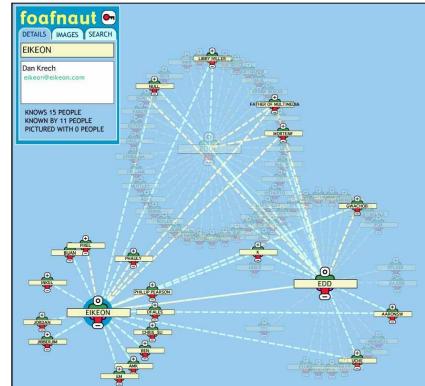
----- Departamento de Informática, UTFSM ----- 2012/2013

Aplicaciones de RDF: FOAF

FOAF = Friend of a Friend (<http://rdfweb.org>)

Vocabulario para definir páginas Personales: redes sociales

FOAFNaut: Usa RDF, SVG, SMIL, etc.



----- Departamento de Informática, UTFSM ----- 2012/2013

DBpedia

A screenshot of a Mozilla Firefox browser window displaying the DBpedia page for 'Spain'. The URL in the address bar is 'http://dbpedia.org/resource/Spain'. The page content includes a brief description of Spain as a country in Southern Europe, followed by a detailed table of properties and values. The table lists various facts about Spain, such as its birthplace (Amaya, Montero), deathplace (Tess, Crosby, Bing, Crosby), and economic data like GDP and HDI. The table is organized into two columns: 'Property' and 'Value'. The entire screenshot is framed by a green border at the bottom.

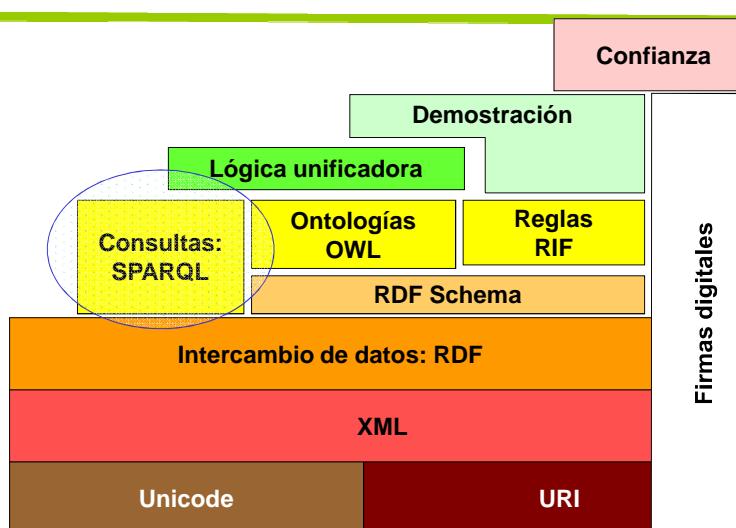
----- Departamento de Informática, UTFSM ----- 2012/2013

SPARQL

Jose Emilio Labra Gayo
Dept. Informática
Universidad de Oviedo

----- Departamento de Informática, UTFSM ----- 2012/2013

SPARQL



----- Departamento de Informática, UTFSM ----- 2012/2013

SPARQL

Los ficheros RDF pueden considerarse bases de datos de tripletas

SPARQL (Abril 2006) es un lenguaje de consulta para datos RDF

Similar a SQL para RDF

Lenguaje de consultas

Basado en RDQL

Modelo = patrones sobre grafos

También describe un protocolo de transporte

SPARQL 1.1 (2011, borrador)

Actualizaciones, consultas federadas, etc.

----- Departamento de Informática, UTFSM ----- 2012/2013

SPARQL Sintaxis Turtle

Sintaxis similar a N3

URIs entre < >

<<http://www.uniovi.es>>

Prefijos para espacios de nombres

PREFIX x: <<http://www.alumnos.org>>

x:profesor

Nodos anónimos

:nombre ó []

Literales entre " "

"Jose Emilio"

"234"^^xsd:integer

Variables empiezan por ?

?nombre

Comentarios empiezan por #

esto es un comentario

Nota: En N3 se ponía @prefix

----- Departamento de Informática, UTFSM ----- 2012/2013

RDF

RDF = Modelo de grafo
Diferentes sintaxis: N3, Turtle, RDF/XML

Ejemplo en Turtle

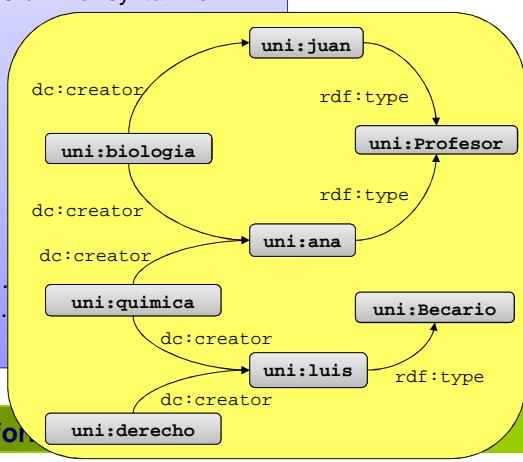
```
@prefix dc: <http://purl.org/dc/terms/> .  
@prefix uni: <http://uniovi.es/> .  
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .  
  
uni:biologia dc:creator uni:juan .  
uni:biologia dc:creator uni:ana .  
uni:ana rdf:type uni:Profesor .  
uni:juan rdf:type uni:Profesor .  
uni:quimica dc:creator uni:ana .  
uni:quimica dc:creator uni:luis .  
uni:luis rdf:type uni:Bebecario .
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Grafo RDF

Ejemplo en Turtle

```
@prefix dc: <http://purl.org/dc/terms/> .  
@prefix uni: <http://uniovi.es/> .  
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .  
  
uni:biologia dc:creator uni:juan .  
uni:biologia dc:creator uni:ana .  
  
uni:quimica dc:creator uni:ana .  
uni:quimica dc:creator uni:luis .  
  
uni:derecho dc:creator uni:luis .  
  
uni:ana rdf:type uni:Profesor .  
uni:juan rdf:type uni:Profesor .  
uni:luis rdf:type uni:Bebecario .
```



----- Departamento de Infor

Consulta RDF

Buscar páginas creadas por un profesor

```
PREFIX dc: <http://purl.org/dc/terms/>
PREFIX uni: <http://uniovi.es/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

SELECT ?p ?c WHERE {
  ?p dc:creator ?c .
  ?c rdf:type uni:Profesor .
}
```

----- Departamento de Informática, UTFSM ----- 2012/2013

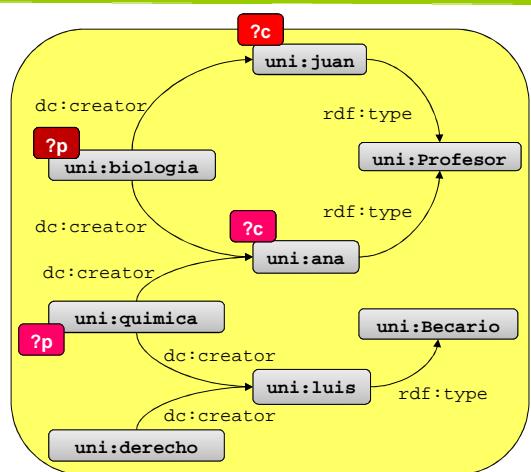
Encaje de grafos

```
SELECT ?p ?c WHERE {
  ?p dc:creator ?c .
  ?c rdf:type uni:Profesor .
}
```



Resultados

?p	?c
uni:biologia	uni:juan
uni:biologia	uni:ana
uni:quimica	uni:ana



----- Departamento de Informática, UTFSM ----- 2012/2013



Ejercicio Test

¿Cuál sería la respuesta de la consulta SPARQL ante el fichero N3 siguiente?

```
@prefix : <http://www.pp.org#>.  
:  
:a :p :b.  
:a :p :c.  
:b :q "M".  
:b :q "N".
```

```
PREFIX e: <http://www.pp.org#>  
SELECT ?z WHERE {  
?x e:p ?y.  
?y e:q ?z.  
}
```

M

N

b

c

M

:a

:b

:c

----- Departamento de Informática, UTFSM ----- 2012/2013



Ejercicio Test

¿Cuál sería la respuesta de la consulta SPARQL ante el fichero N3 siguiente?

```
@prefix : <http://www.pp.org#>.  
:  
:a :p :b.  
:a :p :c.  
:b :q "M".  
:b :q "N".
```

```
PREFIX e: <http://www.pp.org#>  
SELECT ?x WHERE {  
e:a ?x e:c.  
}
```

?x

e:p

e:q

e:b

----- Departamento de Informática, UTFSM ----- 2012/2013

Filtros

FILTER añade restricciones a los valores encajados

```
@prefix e: <http://ejemplo.org#>.  
  
e:Pepe e:nombre "Jose" .  
e:Pepe e:edad 31 .  
  
e:Juan e:nombre "Juan" .  
e:Juan e:edad 12 .  
  
e:Ana e:nombre "Ana" .  
e:Ana e:edad 25.
```

```
PREFIX e: <http://ejemplo.org#>  
  
SELECT ?n ?e WHERE {  
?x e:nombre ?n .  
?x e:edad ?e  
FILTER (?e > 18)  
}
```

n	e
"Ana"	25
"Jose"	31

----- Departamento de Informática, UTFSM ----- 2012/2013

Operadores en los Filtros

FILTER utiliza funciones y operadores de XPath 2.0

Tipos de datos: Boolean, Integer, Float, dataTime, etc.

Operadores habituales: >, <, >=, <=, =, !=, ||, &&

```
PREFIX e: <http://ejemplo.org#>  
  
SELECT ?n ?e WHERE {  
?x e:nombre ?n .  
?x e:edad ?e  
FILTER (?e > 30 || ?e < 18)  
}
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Conversión/creación de tipos de datos

str(arg) : convierte el argumento a una cadena

NOTA: Las URIs deben convertirse a cadenas si se quieren tratar como tales

lang(arg): devuelve el idioma del literal

datatype(arg): devuelve el tipo de datos del literal

uri(arg), iri(arg): convierten el argumento a una URI/IRI

bnode(arg): genera un nodo anónimo

strdt(literal,tipo): genera un literal con un tipo de datos

STRDT("123","xsd:integer") = "123"^^<xsd:integer>

strlang(literal,tipo): genera un literal con un idioma dado

strlang("University", "en") = "University"@ "en"

----- Departamento de Informática, UTFSM ----- 2012/2013

Funciones de comprobación de tipos

isNumeric(arg) = true si el argumento es un número

isBlank(arg) = true si el argumento es un nodo anónimo

isLiteral(arg) = true si el argumento es un literal

isIRI(arg) = true si el argumento es una IRI

bound(arg) = true si el argumento tiene un valor

exists(patrón) = true si se cumple el patrón

not exists(patrón) = true si no se cumple el patrón

if(cond,expr1,expr2) = si se cumple cond, devuelve expr1, si no, devuelve expr2

coalesce(e1,e2,...)= devuelve la primer expresión que se evalúa sin error

----- Departamento de Informática, UTFSM ----- 2012/2013

Ejemplo

Filtrar las notas numéricas

```
@prefix : <http://ejemplo.org#> .  
_:1 :nombre "Juan" .  
_:1 :nota 8.5 .  
  
_:2 :nombre "Luis" .  
_:2 :nota "No presentado" .  
  
_:3 :nombre "Ana" .  
_:3 :nota 6.0 .
```

```
PREFIX : <http://ejemplo.org#>  
  
SELECT ?n WHERE {  
?x :nota ?n .  
FILTER (isNumeric(?n))  
}
```

n
6.0
8.5

----- Departamento de Informática, UTFSM ----- 2012/2013

Funciones con cadenas

`strlen(str)` = longitud de str
`ucase(str)` convierte a mayúsculas
`lcase(str)` convierte a minúsculas
`substr(str,inicio,tam?)` = subcadena a partir de inicio de tamaño tam
 `substr('camino',3,2)='mi'`
`strstarts(str1,str2)` = true si str1 comienza con str2
`strends(str1,str2)` = true si str1 finaliza con str2
`contains(str1,str2)` = true si str1 contiene str2
`encode_for_uri(str)` = resultado de codificar str
`concat(str1,...strN)` = concatenación de cadenas
`langMatches(str,lang)` = true si encaja el idioma
`regex(str,patrón,flags)` = true si encaja la expresión regular

----- Departamento de Informática, UTFSM ----- 2012/2013

Ejemplo

```
@prefix : <http://ejemplo.org#>.  
  
_:1 :nombre "Juan" .  
_:1 :apellidos "Gallardo" .  
  
_:2 :nombre "Julio" .  
_:2 :apellidos "Zamora" .  
  
_:3 :nombre "Luis" .  
_:3 :apellidos "Castro" .
```

```
PREFIX : <http://ejemplo.org#>  
  
SELECT  
(concat(?nombre, ' ',?apells) AS ?persona)  
WHERE  
{  
?x :nombre ?nombre .  
?x :apellidos ?apells .  
FILTER (contains(ucase(?nombre),'L'))  
}
```

persona
"Luis Castro"
"Julio Zamora"

----- Departamento de Informática, UTFSM ----- 2012/2013

Regex

REGEX invoca el encaje de expresiones regulares

Utiliza la función de XPath 2.0

regex(?Expresión, ?Patrón [, ?Flags])

?Expresión = expresión a encajar

?Patrón = expresión regular con la que se encaja

?Flags = opciones para el encaje

```
PREFIX e: <http://ejemplo.org#>  
  
SELECT ?n ?e WHERE {  
?x e:nombre ?n .  
?x e:edad ?e  
FILTER regex(?n,"A","i")  
}
```

Selecciona los nombres que contengan la "A" ó la "a"

----- Departamento de Informática, UTFSM ----- 2012/2013

Regex

Expresiones regulares

`^` = Inicio de cadena
`$` = Fin de la cadena
`.` = Cualquier carácter
`\d` = dígito
`?` = opcional, `*` = 0 ó más, `+` = 1 ó más
`X{n}` = encaja X n veces
`X{m,n}` = encaja X de m a n veces

Flags:

`i` = insensible mayúsculas/minúsculas
`m` = múltiples líneas
`s` = línea simple
`x` = elimina espacios en blanco

----- Departamento de Informática, UTFSM ----- 2012/2013



Ejercicio

El siguiente documento contiene una lista de países

<http://www.di.uniovi.es/~labra/cursos/XML/europa.ttl>

1. Mostrar países cuyo nombre empieza por 'A'
2. Mostrar países cuyo nombre termina por 'a'
3. Mostrar países cuyo nombre empieza por 'A' y termina por 'a'
4. Mostrar países cuyo pib es mayor que 20000
5. Mostrar países cuyo pib es mayor que 20000 y su población menor de 40 millones

----- Departamento de Informática, UTFSM ----- 2012/2013

Funciones numéricas

`abs(n)` = valor absoluto
`floor(n)` = redondear nº hacia abajo
`round(n)` = redondear nº
`ceil(n)` = redondear nº hacia arriba
`rand()` = nº aleatorio entre 0 y 1

----- Departamento de Informática, UTFSM ----- 2012/2013

Funciones con fechas

`now()` = devuelve el instante actual
`year(i)` = devuelve el año de un instante de tiempo i
 `year("2011-01-10T14:45:13.815-05:00"^^xsd:dateTime)` = 2011
`month(i)` = devuelve el mes de i
 `month("2011-01-10T14:45:13.815-05:00"^^xsd:dateTime)` = 1
`day(i)` = devuelve el día de i
 `day("2011-01-10T14:45:13.815-05:00"^^xsd:dateTime)` = 10
`hours(i)` = devuelve la hora de i
 `hours("2011-01-10T14:45:13.815-05:00"^^xsd:dateTime)` = 14
`minutes(i)` = devuelve los minutos de i
 `minutes("2011-01-10T14:45:13.815-05:00"^^xsd:dateTime)` = 45
`seconds(i)` = devuelve los segundos de i
 `seconds("2011-01-10T14:45:13.815-05:00"^^xsd:dateTime)` = 13.815
`timezone(i)` = devuelve la zona temporal de i
 `timezone("2011-01-10T14:45:13.815-05:00"^^xsd:dateTime)` = -PT5H
`tz(i)` = devuelve la zona temporal de i
 `tz("2011-01-10T14:45:13.815-05:00"^^xsd:dateTime)` = -5

----- Departamento de Informática, UTFSM ----- 2012/2013

Funciones HASH

md5(str) = aplica el algoritmo MD5 a str
sha1(str), sha224(str), sha256(str), sha384(str),
sha512(str) = calculan el valor HASH de str utilizando las variaciones correspondientes del algoritmo SHA

```
@prefix : <http://ejemplo.org#>.  
  
@prefix : <http://ejemplo.org#> .  
  
_:1 :nombre "Juan" .  
_:1 :email "juan@uni.com" .  
  
_:2 :nombre "Luis" .  
_:2 :email "luis@uni.com" .
```

```
PREFIX : <http://ejemplo.org#>  
  
SELECT ?nombre  
      (SHA1(?email) AS ?sha1Email)  
WHERE {  
?x :nombre ?nombre .  
?x :email ?email .  
}
```

nombre	sha1Email
"Luis"	"c3fbebedb973ffbf0b319f152562b31552f03f157"
"Juan"	"bb402e5fe4d9ccbc8895caf0bae12122f1b0d2fa"

----- Departamento de Informática, UTFSM 2012/2013

Unión de grafos

UNION combina resultados de varios grafos

```
@prefix e: <http://ejemplo.org#>.  
@prefix foaf: <http://xmlns.com/foaf/0.1/>.  
  
e:Pepe e:nombre "Jose" .  
e:Pepe e:edad 31 .  
e:Pepe e:conoceA e:Juan .  
  
e:Juan foaf:name "Juan" .  
e:Juan e:edad 25 .  
e:Juan e:conoceA e:Ana .  
  
e:Ana foaf:name "Ana" .  
e:Ana e:nombre "Ana Mary" .
```

```
SELECT ?n  
WHERE {  
?x foaf:name ?n }  
UNION  
?y p:nombre ?n  
}
```

n
"Ana"
"Juan"
"Ana Mary"
"Jose"

----- Departamento de Informática, UTFSM 2012/2013

Encajes opcionales

OPTIONAL permite obtener valores en triplets sin fallar cuando éstas no existan

```
@prefix e: <http://ejemplo.org#>.  
@prefix foaf: <http://xmlns.com/foaf/0.1/>.  
  
e:Pepe e:nombre "Jose".  
e:Pepe e:edad 31.  
  
e:Juan e:nombre "Juan".  
  
e:Ana e:nombre "Ana".  
e:Ana e:edad 13.
```

```
PREFIX e: <http://ejemplo.org#>  
  
SELECT ?n ?e WHERE {  
?x e:nombre ?n .  
OPTIONAL { ?x e:edad ?e }  
}
```

n	e
"Ana"	13
"Juan"	
"Jose"	31

----- Departamento de Informática, U

2/2013

Especificar grafos de entrada

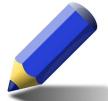
FROM indica la URL de la que proceden los datos

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>  
SELECT ?n  
FROM <http://www.di.uniovi.es/~labra/labraFoaf.rdf>  
WHERE { ?x foaf:name ?n }
```

Si se incluyen varios conjuntos de entrada se realiza la mezcla de los grafos resultantes

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>  
SELECT ?n  
FROM <http://www.di.uniovi.es/~labra/labraFoaf.rdf>  
FROM <http://www.w3.org/People/Berners-Lee/card>  
WHERE {  
?x foaf:name ?n  
}
```

----- Departamento de Informática, UTFSM ----- 2012/2013



Ejercicio

Modelizar las siguientes tablas en 2 ficheros Turtle diferentes

DNI	Nombre	Apellidos
9999	Juan	Gallardo
8888	Jose	Torre
7777	Ana	Cascos

DNI	Nota
9999	3
7777	5

Construir una consulta que permita visualizar la nota de cada alumno junto con su nombre y apellidos

----- Departamento de Informática, UTFSM ----- 2012/2013

Grafos con nombre

FROM NAMED asigna un nombre al grafo de entrada
GRAPH encaja con el nombre del grafo que corresponda

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?n ?g
FROM NAMED <http://www.w3.org/People/Berners-Lee/card>
FROM NAMED <http://www.di.uniovi.es/~labra/labraFoaf.rdf>
WHERE {
  GRAPH ?g { ?x foaf:name ?n }
}
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Control de los resultados

DISTINCT elimina valores duplicados

ORDER BY permite especificar el orden de los resultados (puede especificarse ASC, DESC...)

LIMIT n indica el número de resultados

OFFSET m indica a partir de qué resultado empezar a contar

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT DISTINCT ?n
WHERE { ?x foaf:knows ?y .
        ?y foaf:name ?n . }
ORDER BY ASC(?n)
LIMIT 5
OFFSET 10
```

----- Departamento de Informática, UTFSM ----- 2012/2013

CONSTRUCT

Permite crear un grafo de salida

```
@prefix : <http://ejemplo.org#> .
_:1 :nombre "Juan" .
_:1 :amigoDe _:2, _:3 .

_:2 :nombre "Luis" .
_:2 :amigoDe _:1 .

_:3 :nombre "Ana" .
_:3 :amigoDe _:1.
```

```
PREFIX : <http://ejemplo.org#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

CONSTRUCT {
  ?x foaf:name ?n .
  ?x foaf:knows ?y .
}
WHERE {
  ?x :nombre ?n .
  ?x :amigoDe ?y .
}
```

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .

_:bl foaf:knows [ foaf:knows _:bl ;
                   foaf:name "Luis" ]
      ;
     foaf:knows [ foaf:knows _:bl ;
                   foaf:name "Ana" ]
      ;
     foaf:name "Juan" .
```

----- Departamento

3

ASK

ASK devuelve sí o no

Puede ser útil para chequeo de errores

```
@prefix : <http://ejemplo.org#> .  
_:1 :nombre "Juan" .  
_:1 :nota 8.5 .  
  
_:2 :nombre "Luis" .  
_:2 :nota "No presentado" .  
  
_:3 :nombre "Ana" .  
_:3 :nota 6.0 .
```

```
PREFIX : <http://ejemplo.org#>  
ASK WHERE {  
?x :nota ?n .  
FILTER ( ! isNumeric(?n))  
}
```

Yes

----- Departamento de Informática, UTFSM ----- 2012/2013

DESCRIBE

Devuelve una descripción RDF de uno o varios nodos

```
@prefix : <http://ejemplo.org#> .  
_:1 :nombre "Juan" .  
_:1 :nota 8.5 .  
  
_:2 :nombre "Luis" .  
_:2 :nota "No presentado" .  
  
_:3 :nombre "Ana" .  
_:3 :nota 6.0 .
```

```
PREFIX : <http://ejemplo.org#>  
DESCRIBE ?x WHERE  
{  
?x :nota ?n .  
FILTER(?n = 6.0)  
}
```

```
@prefix : <http://ejemplo.org#> .  
[ ] :nombre "Ana" ;  
:nota 6.0 .
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Asignaciones

BIND **expr AS v** = Asigna el valor de **expr** a la variable **v**

```
@prefix e: <http://ejemplo.org#>.  
  
_:1 e:nombre "Manzanas" .  
_:1 e:cantidad 3 .  
_:1 e:precio 3 .  
  
_:2 e:nombre "Peras" .  
_:2 e:cantidad 2 .  
_:2 e:precio 2 .  
  
_:3 e:nombre "Naranjas" .  
_:3 e:cantidad 4 .  
_:3 e:precio 1 .
```

PREFIX e: <http://ejemplo.org#>

```
SELECT ?n ?precioTotal  
WHERE {  
?x e:nombre ?n .  
?x e:cantidad ?cantidad .  
?x e:precio ?precio .  
BIND ((?cantidad * ?precio) AS ?precioTotal)  
}
```

n	precioTotal
"Naranjas"	4
"Peras"	4
"Manzanas"	9

----- Departamento de Informática, UTFSM ----- 2012/2013

Asignaciones en SELECT

Es posible realizar la asignación directamente

```
@prefix e: <http://ejemplo.org#>.  
  
_:1 e:nombre "Manzanas" .  
_:1 e:cantidad 3 .  
_:1 e:precio 3 .  
  
_:2 e:nombre "Peras" .  
_:2 e:cantidad 2 .  
_:2 e:precio 2 .  
  
_:3 e:nombre "Naranjas" .  
_:3 e:cantidad 4 .  
_:3 e:precio 1 .
```

PREFIX e: <http://ejemplo.org#>

```
SELECT ?n ((?cantidad * ?precio) AS ?precioTotal)  
WHERE {  
?x e:nombre ?n .  
?x e:cantidad ?cantidad .  
?x e:precio ?precio .  
}
```

n	precioTotal
"Naranjas"	4
"Peras"	4
"Manzanas"	9

----- Departamento de Informática, UTFSM ----- 2012/2013

Funciones de agregación: AVG, SUM, COUNT, SAMPLE

```
@prefix e: <http://ejemplo.org#>.  
  
e:Pepe e:nombre "Jose" .  
e:Pepe e:edad 31 .  
  
e:Juan e:nombre "Juan" .  
e:Juan e:edad 12 .  
  
e:Ana e:nombre "Ana" .  
e:Ana e:edad 25.
```

```
PREFIX e: <http://ejemplo.org#>  
  
SELECT (AVG(?edad) AS ?media)  
(SUM(?edad) AS ?suma)  
(COUNT(?edad) AS ?cuenta)  
(SAMPLE(?edad) AS ?muestra)  
  
WHERE  
{  
    ?n e:edad ?edad .  
}
```

media	suma	cuenta	muestra
22.66	68	3	25

----- Departamento de Informática, UTFSM ----- 2012/2013

Funciones de agregación: MAX, MIN

```
@prefix e: <http://ejemplo.org#>.  
  
e:Pepe e:nombre "Jose" .  
e:Pepe e:edad 31 .  
  
e:Juan e:nombre "Juan" .  
e:Juan e:edad 12 .  
  
e:Ana e:nombre "Ana" .  
e:Ana e:edad 25.
```

```
PREFIX e: <http://ejemplo.org#>  
  
SELECT (MAX(?edad) as ?mayor)  
(MIN(?edad) as ?menor)  
  
WHERE  
{  
    ?n e:edad ?edad .  
}
```

mayor	menor
31	12

----- Departamento de Informática, UTFSM ----- 2012/2013

Funciones de agregación GROUP_CONCAT

```
@prefix e: <http://ejemplo.org#>.  
  
e:Pepe e:nombre "Jose".  
e:Pepe e:edad 31 .  
  
e:Juan e:nombre "Juan".  
e:Juan e:edad 12 .  
  
e:Ana e:nombre "Ana".  
e:Ana e:edad 25.
```

```
PREFIX e: <http://ejemplo.org#>  
  
SELECT (GROUP_CONCAT(?edad;  
SEPARATOR=',')  
as ?edades) where  
{  
?n e:edad ?edad .  
}
```

edades
=
25, 12, 31

----- Departamento de Informática, UTFSM ----- 2012/2013

Agrupaciones: GROUP_BY

GROUP BY permite agrupar conjuntos de resultados

```
@prefix e: <http://ejemplo.org#>.  
  
.1 e:nombre "Ana".  
.1 e:edad 18 .  
.1 e:notas 8 .  
  
.2 e:nombre "Juan".  
.2 e:edad 20 .  
.2 e:notas 7 .  
  
.3 e:nombre "Luis".  
.3 e:edad 18 .  
.3 e:notas 5 .  
  
.4 e:nombre "Mario".  
.4 e:edad 19 .  
.4 e:notas 6 .  
  
.5 e:nombre "Carlos".  
.5 e:edad 20 .  
.5 e:notas 9 .
```

```
PREFIX e: <http://ejemplo.org#>  
  
SELECT (AVG(?nota) AS ?mediaNota) ?edad  
WHERE {  
?x e:nombre ?n .  
?x e:edad ?edad .  
?x e:notas ?nota .  
}  
GROUP BY ?edad
```

mediaNota	edad
6.5	18
8.0	20
6.0	19

----- Departamento de Informática, UTFSM ----- 2012/2013

Agrupaciones: HAVING

HAVING permite filtrar los grupos que cumplan una condición

```
@prefix e: <http://ejemplo.org#>.  
  
_:1 e:nombre "Ana".  
_:1 e:edad 18 .  
_:1 e:notas 8 .  
  
_:2 e:nombre "Juan".  
_:2 e:edad 20 .  
_:2 e:notas 7 .  
  
_:3 e:nombre "Luis".  
_:3 e:edad 18 .  
_:3 e:notas 5 .  
  
_:4 e:nombre "Mario".  
_:4 e:edad 19 .  
_:4 e:notas 6 .  
  
_:5 e:nombre "Carlos".  
_:5 e:edad 20 .  
_:5 e:notas 9 .
```

```
PREFIX e: <http://ejemplo.org#>  
  
SELECT (AVG(?nota) AS ?mediaNota) ?edad  
WHERE {  
?x e:nombre ?n .  
?x e:edad ?edad .  
?x e:notas ?nota .  
}  
GROUP BY ?edad  
HAVING (?mediaNota < 8)
```

mediaNota	edad
6.5	18
6.0	19

Departamento de Informática

2013

Subconsultas

Es posible realizar consultas dentro de consultas

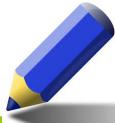
```
@prefix e: <http://ejemplo.org#>.  
  
_:1 e:nombre "Ana".  
_:1 e:edad 18 .  
_:1 e:notas 8 .  
  
_:2 e:nombre "Juan".  
_:2 e:edad 20 .  
_:2 e:notas 7 .  
  
_:3 e:nombre "Luis".  
_:3 e:edad 18 .  
_:3 e:notas 5 .  
  
_:4 e:nombre "Mario".  
_:4 e:edad 19 .  
_:4 e:notas 6 .  
  
_:5 e:nombre "Carlos".  
_:5 e:edad 20 .  
_:5 e:notas 9 .
```

```
PREFIX e: <http://ejemplo.org#>  
  
SELECT ?nombre ?nota  
      (?nota - ?notaMedia AS ?desv)  
WHERE {  
?x e:nombre ?nombre .  
?x e:notas ?nota .  
{  
SELECT (AVG(?nota) AS ?notaMedia) WHERE {  
?x e:notas ?nota .  
}  
}  
}
```

nombre	nota	desv
"Carlos"	9	2
"Mario"	6	-1
"Luis"	5	-2
"Juan"	7	0
"Ana"	8	1

Departamento de Informática

2013



Ejercicio

El siguiente documento contiene una lista de países
<http://www.di.uniovi.es/~labra/cursos/XML/europa.ttl>

1. Mostrar el país con mayor PIB
2. Mostrar el PIB medio
3. Mostrar países cuyo PIB es mayor que el PIB medio
4. Mostrar países de una población similar a la de España cuyo PIB esté por encima

----- Departamento de Informática, UTFSM ----- 2012/2013

Caminos de propiedades

La URI que identifica la propiedad puede contener una expresión regular

p	Encaja con la propiedad p
(e)	Camino agrupado entre paréntesis
^e	Camino inverso de e
!p	No encaja con la propiedad p
e1 / e2	Camino e1 seguido de e2
e1 e2	Camino e1 ó e2
e*	0 ó más apariciones de e
e+	1 ó más apariciones de e
e?	0 ó 1 aparición de e
e{n}	n apariciones de e
e{m,n}	Entre m y n apariciones de e
e{n,}	n ó más apariciones de e
e{,n}	Entre 0 y n apariciones de e

----- Departamento de Informática, UTFSM ----- 2012/2013

Caminos de propiedades

```
@prefix e: <http://ejemplo.org#>.  
@prefix foaf: <http://xmlns.com/foaf/0.1/>.  
  
e:Pepe e:nombre "Jose".  
  
e:Juan foaf:name "Juan".  
  
e:Ana foaf:name "Ana".  
e:Ana e:nombre "Ana Mary".
```

```
PREFIX e: <http://ejemplo.org#>  
PREFIX foaf: <http://xmlns.com/foaf/0.1/>  
  
SELECT ?n  
WHERE {  
    ?x (foaf:name | e:nombre) ?n  
}
```

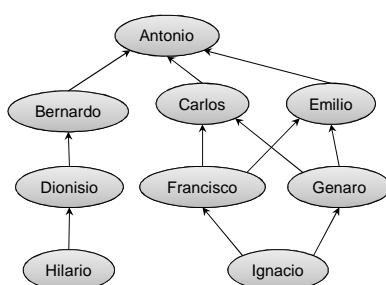
n

"Ana"
"Ana Mary"
"Jose"
"Juan"

***** Departamento de Informática, UTFSM ***** 2012/2013

Caminos de propiedades

```
e:Ignacio e:conoceA e:Francisco, e:Genaro.  
e:Francisco e:conoceA e:Carlos, e:Emilio .  
e:Genaro e:conoceA e:Carlos, e:Emilio .  
e:Carlos e:conoceA e:Antonio .  
e:Emilio e:conoceA e:Antonio .  
e:Hilario e:conoceA e:Dionisio .  
e:Dionisio e:conoceA e:Bernardo .  
e:Bernardo e:conoceA e:Antonio .
```



```
PREFIX e: <http://ejemplo.org#>  
  
SELECT ?p  
{  
    ?p e:conoceA+ e:Antonio.  
}
```

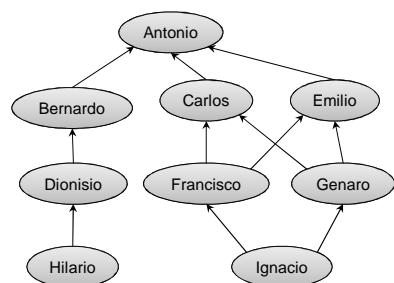
| p |
=====|
| e:Bernardo |
| e:Dionisio |
| e:Hilario |
| e:Emilio |
| e:Genaro |
| e:Ignacio |
| e:Francisco |
| e:Ignacio |
| e:Carlos |
| e:Genaro |
| e:Ignacio |
| e:Francisco |
| e:Ignacio |

***** Departamento de Informática, UTFSM *****

Caminos de propiedades

```
e:Ignacio e:conoceA e:Francisco, e:Genaro.  
e:Francisco e:conoceA e:Carlos, e:Emilio .  
e:Genaro e:conoceA e:Carlos, e:Emilio .  
e:Carlos e:conoceA e:Antonio .  
e:Emilio e:conoceA e:Antonio .  
e:Hilario e:conoceA e:Dionisio .  
e:Dionisio e:conoceA e:Bernardo .  
e:Bernardo e:conoceA e:Antonio .
```

```
PREFIX e: <http://ejemplo.org#>  
  
SELECT ?p  
{  
?p e:conoceA{2} e:Antonio.  
}
```



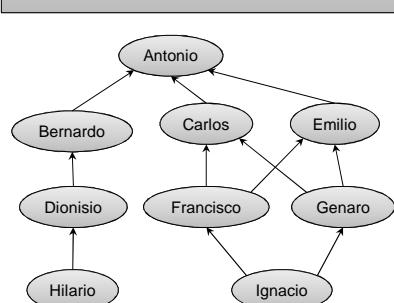
```
| p  
=====|  
| e:Dionisio |  
| e:Genaro |  
| e:Francisco |  
| e:Genaro |  
| e:Francisco |
```

----- Departamento de Informática, UTSFSM ----- 2012/2013

Caminos de propiedades

```
e:Ignacio e:conoceA e:Francisco, e:Genaro.  
e:Francisco e:conoceA e:Carlos, e:Emilio .  
e:Genaro e:conoceA e:Carlos, e:Emilio .  
e:Carlos e:conoceA e:Antonio .  
e:Emilio e:conoceA e:Antonio .  
e:Hilario e:conoceA e:Dionisio .  
e:Dionisio e:conoceA e:Bernardo .  
e:Bernardo e:conoceA e:Antonio .
```

```
PREFIX e: <http://ejemplo.org#>  
  
SELECT DISTINCT ?p  
{  
?p e:conoceA/e:conoceA e:Antonio.  
}
```



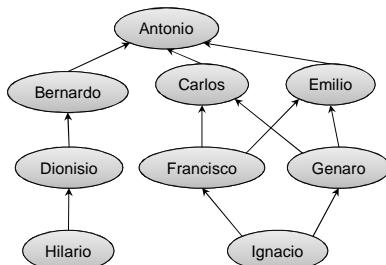
```
| p  
=====|  
| e:Dionisio |  
| e:Genaro |  
| e:Francisco |  
| e:Genaro |  
| e:Francisco |
```

----- Departamento de Informática, UTSFSM ----- 2012/2013

Caminos de propiedades

```
e:Ignacio e:conoceA e:Francisco, e:Genaro.  
e:Francisco e:conoceA e:Carlos, e:Emilio .  
e:Genaro e:conoceA e:Carlos, e:Emilio .  
e:Carlos e:conoceA e:Antonio .  
e:Emilio e:conoceA e:Antonio .  
e:Hilario e:conoceA e:Dionisio .  
e:Dionisio e:conoceA e:Bernardo .  
e:Bernardo e:conoceA e:Antonio .
```

```
PREFIX e: <http://ejemplo.org#>  
  
SELECT ?p  
{  
?p e:conoceA/^e:conoceA e:Francisco.  
FILTER (?p != e:Francisco)  
}
```



```
-----  
| p |  
=====|  
| e:Genaro |  
| e:Genaro |  
-----
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Inserción

INSERT DATA permite insertar triplets en un grafo

```
PREFIX e: <http://ejemplo.org#>  
  
INSERT DATA {  
  
e:ana e:nombre "Ana".  
e:ana e:edad 18 .  
e:ana e:notas 8 .  
  
e:juan e:nombre "Juan Manuel".  
e:juan e:edad 20 .  
e:juan e:notas 7 .  
  
e:luis e:nombre "Luis".  
e:luis e:edad 18 .  
e:luis e:notas 5 .  
}
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Inserción

INSERT permite insertar triplets en un grafo.

Requiere una cláusula WHERE

```
PREFIX e: <http://ejemplo.org#>

INSERT {
  ?p e:nombreNota "Notable".
}
WHERE {
  ?p e:nota ?nota .
  FILTER (?nota >= 7 && ?nota < 9)
}
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Carga de grafo

LOAD permite cargar todas las triplets existentes en una URI

```
LOAD <http://www.di.uniovi.es/~labra/labraFoaf.rdf>
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Borrado

DELETE DATA permite eliminar tripletas de un grafo

```
PREFIX e: <http://ejemplo.org#>  
  
DELETE DATA  
{  
    e:luis e:not 5 .  
}
```

NOTA: DELETE DATA No admite variables

----- Departamento de Informática, UTFSM ----- 2012/2013

Borrado

DELETE WHERE permite eliminar tripletas de un grafo especificando una condición

```
PREFIX e: <http://ejemplo.org#>  
  
DELETE  
{ ?x e:nota ?nota . }  
WHERE  
{  
    ?x e:nota ?nota .  
    FILTER (?nota >= 8)  
}
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Actualización

INSERT/DELETE permite actualizar tripletas de un grafo

Ejemplo: incrementar la edad

```
PREFIX e: <http://ejemplo.org#>

DELETE { ?x e:edad ?edad }
INSERT { ?x e:edad ?edad1 }
WHERE {
  ?x e:edad ?edad .
  BIND((?edad + 1) AS ?edad1)
}
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Borrado total

CLEAR borra todas las tripletas

Puede indicarse el conjunto de datos

CLEAR g = Borra grafo g

CLEAR DEFAULT = Borra grafo actual

CLEAR ALL = Borra todos los grafos

----- Departamento de Informática, UTFSM ----- 2012/2013

Acceso a servicios remotos

SERVICE uri = indica un endpoint SPARQL

```
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT ?nombre WHERE {
  SERVICE <http://dbpedia.org/sparql> {
    SELECT ?nombre WHERE {
      ?pais rdf:type dbo:Country .
      ?pais rdfs:label ?nombre .
      FILTER (lang(?nombre)='es')
    }
  }
}
```

Lista de terminales SPARQL
<http://esw.w3.org/topic/SparqlEndpoints>

----- Departamento de Informática, UTFSM ----- 2012/2013

Consultas federadas

Realizar consultas combinando resultados

DBpedia: <http://dbpedia.org>
IMDB: <http://data.linkedmdb.org>

```
PREFIX imdb: <http://data.linkedmdb.org/resource/movie/>
PREFIX dcterms: <http://purl.org/dc/terms/>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT * {
  { SERVICE <http://dbpedia.org/sparql>
    { SELECT ?fechaNacim ?nombreMujer WHERE {
        ?actor rdfs:label "Javier Bardem"@en ;
        dbo:birthDate ?fechaNacim ;
        dbo:spouse ?mujerURI .
        ?mujerURI rdfs:label ?nombreMujer .
        FILTER ( lang(?nombreMujer) = "en" )
      }
    }
  { SERVICE <http://data.linkedmdb.org/sparql>
    { SELECT ?peli ?fechaPeli WHERE {
        ?actor imdb:actor_name "Javier Bardem".
        ?movie imdb:actor ?actor ;
          dcterms:title ?peli ;
          dcterms:date ?fechaPeli .
      }
    }
  }
}
```

----- Departamento de In



Ejercicio

Buscar en la DBPedia películas españolas, mostrando el título, el nombre del director y el año en que se hicieron.

Combinar la información con otros terminales SPARQL

----- Departamento de Informática, UTFSM ----- 2012/2013

Patrón de negación por fallo en SPARQL

Combinando FILTER, OPTIONAL y !BOUND se puede simular la negación por fallo

Ejemplo: Buscar personas que no están casadas.

```
@prefix : <http://ej.org#>.  
  
:Pepe :estaCasadoCon :Ana .  
:Pepe :nombre "Jose" .  
  
:Luis :estaCasadoCon :Marta .  
:Luis :nombre "Luis" .  
  
:Carlos :nombre "Carlos" .
```

```
PREFIX : <http://ej.org#>  
SELECT ?n WHERE {  
?x :nombre ?n  
OPTIONAL {?x :estaCasadoCon ?y }  
FILTER (!BOUND(?y ))  
}
```

¿Realmente muestra los que no están casados?

----- Departamento de Informática, UTFSM ----- 2012/2013

Creación de Base de Datos RDF

Fuseki permite trabajar con datos RDF como una base de datos

Es posible realizar consultas SPARQL

Arrancar servidor

```
> mkdir dirDatos  
> java -jar fuseki-sys.jar --update --loc=dirDatos /datos
```

Acceso en puerto: <http://localhost:3030>

----- Departamento de Informática, UTFSM ----- 2012/2013



Universidad Técnica Federico Santa María
Departamento de Informática



Aplicaciones basadas en RDF

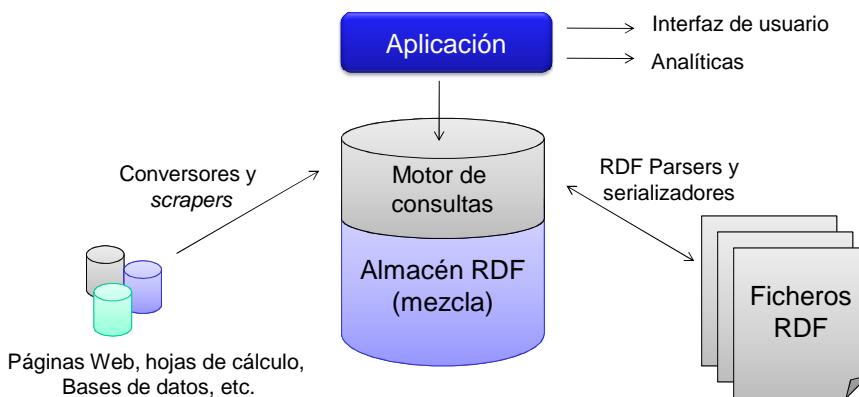
Jose Emilio Labra Gayo

Departamento de Informática
Universidad de Oviedo

----- Departamento de Informática, UTFSM ----- 2012/2013

Componentes de Aplicación

Arquitectura básica de aplicación RDF



----- Departamento de Informática, UTFSM ----- 2012/2013

Conversores y Scrapers

Conversores: Convierte contenido de otros formatos a RDF

Existen herramientas específicas para diferentes dominios

Ejemplos: <http://esw.w3.org/topic/ConverterToRdf>

Scraper (~rascador) obtiene RDF a partir de datos HTML

Técnicas de IA y reconocimiento de lenguaje natural

Ejemplo: [Solvent](http://simile.mit.edu/wiki/Solvent) (<http://simile.mit.edu/wiki/Solvent>)

----- Departamento de Informática, UTFSM ----- 2012/2013

Parsers y serializadores

Parsers: Analizan documentos RDF

Obtienen representación del modelo RDF (Grafo)

Jena (<http://jena.sourceforge.net/>) librería Java con diversas utilidades

Serializadores: Generan documentos RDF

NOTA: el documento resultante de leer/escribir el mismo grafo puede ser diferente

----- Departamento de Informática, UTFSM ----- 2012/2013

Almacén RDF

Operación básica: Mezclar modelos RDF

Integración de información en RDF

Métodos de almacenamiento:

BD Nativas en RDF

Ejemplo: Sesame (<http://www.openrdf.org/>)

BD relacionales con soporte para RDF

Ejemplo: Oracle 11 da soporte a RDF

----- Departamento de Informática, UTFSM ----- 2012/2013

Aplicaciones RDF

Varias librerías para diversos lenguajes

Jena (Java)

SemWeb (<http://razor.occams.info/code/semweb/>) para .Net

Librería Redland (<http://librdf.org/>) escrita en C y con adaptadores para Python, Ruby, PHP, etc.

Portales basados en RDF

RDF puede aumentar la flexibilidad del modelo de datos

Ejemplos:

<http://www.w3.org/2001/sw/sweo/public/UseCases/>

----- Departamento de Informática, UTFSM ----- 2012/2013

Motor de consultas: SPARQL

SPARQL (2006): Lenguaje de consulta y protocolo de acceso

≈ SQL para RDF

Basado en encaje de patrones sobre grafos

Extensiones no estándar para actualización

Terminales SPARQL (endpoints) permiten hacer consultas a una URI

----- Departamento de Informática, UTFSM ----- 2012/2013

RDF en HTML?

----- Departamento de Informática, UTFSM ----- 2012/2013

Anotación páginas HTML mediante RDF

Problema: Incluir descripciones RDF en páginas HTML

La sintaxis RDF/XML impide la validación de HTML

Soluciones:

Incluir RDF como comentarios

Difícil de generar con las herramientas XML

Fácil para el usuario

Un comentario no deja de ser un comentario

Extender XHTML para incluir RDF

2 formas:

Añadirlo sin más ⇒ XHTML no válido

Extender la DTD de XHTML

Utilizar <link> para enlazar a un fichero RDF externo

Problema: mantenimiento de 2 ficheros independientes

----- Departamento de Informática, UTFSM ----- 2012/2013

Incluir RDF como comentarios

```
<div id="f-lastmod"> This page was last modified 12:26, 11 July 2006.</div>
<div id="f-copyright">This wiki is licensed to the public under a
<a href="http://creativecommons.org/licenses/by/3.0/">
  class="external"
  title="http://creativecommons.org/licenses/by/3.0/"
  rel="nofollow">Creative Commons Attribution 3.0</a> license.<br/>
  Your use of this wiki is governed by the
  <a href="/CcWiki:Terms_of_Use">Terms of Use</a>.
<!-- rdf:RDF xmlns="http://web.resource.org/cc/" -->
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#",
  <Work rdf:about="">
    <license rdf:resource="http://creativecommons.org/licenses/by/2.5/" />
    <dc:type rdf:resource="http://purl.org/dc/dcmitype/Text" />
  </Work>
  <License rdf:about="http://creativecommons.org/licenses/by/2.5/">
    <permits rdf:resource="http://web.resource.org/cc/Reproduction"/>
    <permits rdf:resource="http://web.resource.org/cc/Distribution"/>
    <requires rdf:resource="http://web.resource.org/cc/Notice"/>
    <requires rdf:resource="http://web.resource.org/cc/Attribution"/>
    <permits rdf:resource="http://web.resource.org/cc/DerivativeWorks"/>
  </License>
</rdf:RDF> -->
</div>
<div id="f-about">
<a href="/CcWiki:About" title="CcWiki:About">About CcWiki</a>
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Añadir RDF en HTML

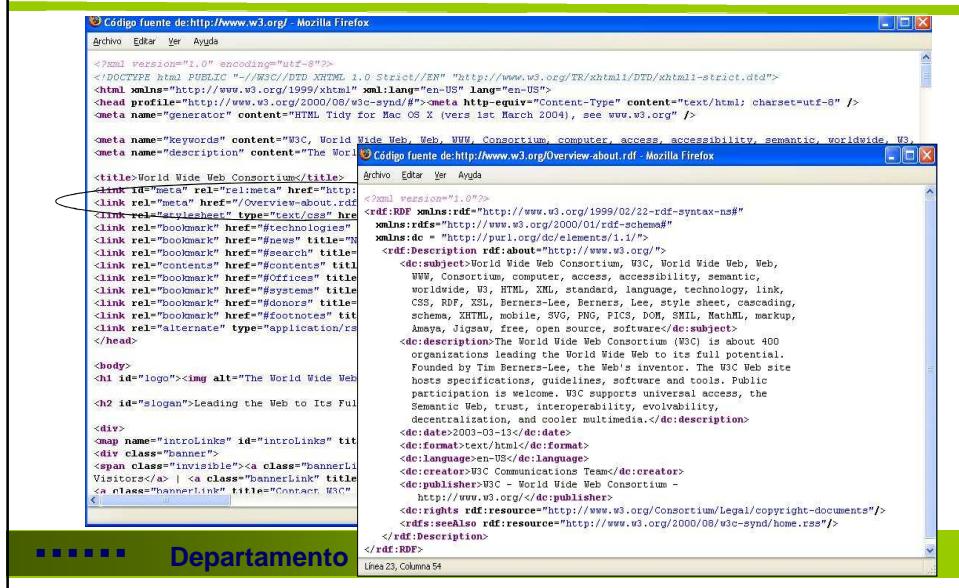
```
<!DOCTYPE html SYSTEM
  "http://infomesh.net/2002/m12n/test/rdf.txt">

<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xml:lang="en" >
<head>
<title>Pagina de Libros</title>
<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.libros.net"
                   dc:subject="Literatura"/>
</rdf:RDF>

</head>
...
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Utilizar <link> para enlazar a un fichero



The screenshot displays two Mozilla Firefox windows. The left window shows the source code of the W3C homepage (<http://www.w3.org/>). The right window shows the RDF representation of the same page, generated by an RDF Tidy tool. The RDF code is based on the source code, showing how each element is linked to its corresponding resource.

Microformatos

Añadir semántica a HTML reutilizando características ya existentes

Objetivo: facilitar procesamiento automático de datos incluidos en páginas Web

Ejemplos: Información sobre eventos, contactos, lugares, etc.

Desarrollo no estándar.

Aportaciones colaborativas en wiki: www.microformats.org

Método: utilizar atributos **class**, **rel** y **rev**

Ejemplo: "Oficina situada en coordenadas 23.4, -1.3" podría codificarse como:

Oficina situada en coordenadas

```
<span class="geo">
  <span class="latitude">23.4</span> y
  <span class="longitude">-1.8</span>
</span>
```

Utiliza la especificación geo 2013

Microformatos

Información de contacto utilizando hCard:

Sin microformatos:

```
<div>
  <div>Jose Labra</div>
  <div>Universidad de Oviedo</div>
  <div>+34-985103394</div>
  <a href="http://www.di.uniovi.es/~labra">
    http://www.di.uniovi.es/~labra</a>
</div>
```

Con microformatos:
hCard

```
<div class="vcard">
  <div class="fn">Jose Labra</div>
  <div class="org">Universidad de Oviedo</div>
  <div class="tel">+34-985103394</div>
  <a class="url" href="http://www.di.uniovi.es/~labra">
    http://www.di.uniovi.es/~labra</a>
</div>
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Microformatos

Diversas propuestas:

XFN (XHTML Friends Network) permite describir relaciones

hCard información de personas y organizaciones (basado en vCard)

Incluye adr para direcciones postales y geo para lugares geográficos

hCalendar describe eventos. Se basa en iCalendar

hAtom permite anotar ficheros Atom

hProduct permite describir productos

hResume describe el curriculum vitae de una persona

hReview permite definir revisiones y valoraciones

XOXO describe listas

Rel-license describe el uso de valores de licencias

----- Departamento de Informática, UTFSM ----- 2012/2013

RDFa

RDFa, propuesto en 2004 para añadir semántica a documentos XHTML.

Recomendación W3c (2008)

Inspirado en microformatos

Se codifican tripletas RDF mediante atributos de HTML

RDFa 1.1 borrador en 2011

Intenta facilitar la creación de documentos

Admite prefijos de espacios de nombres

Permite crear perfiles y vocabularios

----- Departamento de Informática, UTFSM ----- 2012/2013

RDFa

Sujeto: se especifica mediante atributo `about`

Predicado: mediante `property`, `rel`, `rev`

Objetos (URIs) mediante `href`, `resource` ó `src`

Literales: mediante `content` el contenido propio del elemento. Atributo opcional `datatype` para tipo de datos

Prefix: permite declarar prefijos de espacios de nombres

Vocab: permite declarar vocabularios a utilizar

Herramienta: <http://check.rdfainfo/>

----- Departamento de Informática, UTFSM ----- 2012/2013

Ejemplo RDFa

```
<p>En su último libro "La Primavera", Juan Torre habla sobre la primavera. El libro fue publicado en Junio de 2009</p>
```

En Turtle @prefix dc: <http://purl.org/dc/elements/1.1/>.

```
<http://www.libros.com/primavera> dc:title "La Primavera" ;  
dc:creator "Juan Torre";  
dc:date "2009-06-03" .
```

```
<p xmlns:dc="http://purl.org/dc/elements/1.1/"  
about="http://www.libros.com/primavera">  
En su último libro <span property="dc:title">La Primavera</span>,  
<span property="dc:creator">Juan Torre</span> habla sobre la primavera.  
El libro fue publicado el  
<span property="dc:date" content="2009-06-03">3 de Junio de 2009</span>  
</p>
```

***** Departamento de Informática, UTFSM ***** 2012/2013

Ejemplo RDFa (2)

```
<p>Me llamo Jose Luis Torre, nací el 1 de diciembre de 1974 y soy Profesor de la Universidad de Oviedo</p>
```

```
<div xmlns:foaf="http://xmlns.com/foaf/0.1/"  
xmlns:e="http://www.ejemplo.org#"  
xmlns:xsd="http://www.w3.org/2001/XMLSchema#"  
typeof="foaf:Person"  
about="[e:juan]">  
Me llamo <span property="foaf:name">Juan Luis Torre</span>,  
nací el  
<span property="foaf:birthDay"  
content="1974-12-01"  
datatype="xsd:dateTime">1 de diciembre de 1974</span>  
y soy profesor de la  
<span about="[e:uniovi]"  
typeof="foaf:Organization"  
property="foaf:name"  
rel="foaf:member"  
resource="[e:juan]">Universidad de Oviedo</span>  
</div>
```

***** Departamento de Informática, UTFSM ***** 2012/2013

Herramientas RDFa

RDF-Translator: <http://rdf-translator.appspot.com/>

Google Rich Snippets Tool:

<http://www.google.com/webmasters/tools/richsnippets>

Sindice Inspector: <http://sindice.com/developers/inspector>

Check RDFa: <http://check.rdfainfo.info/>

RDFa Developer (Extensión de Firefox)

StructuredData.org <http://linter.structured-data.org/>

RDFa Live loop <http://rdfa.digitalbazaar.com/live-loop/>

----- Departamento de Informática, UTFSM ----- 2012/2013

Microdatos

Propuestos para HTML5

Se identifica un nuevo ítem mediante `itemscope`

Propiedades mediante `itemprop`

Tipos mediante `itemtype`

Genera JSON

----- Departamento de Informática, UTFSM ----- 2012/2013

Ejemplo

Ejemplo

```
<p>En su último libro “La Primavera”, Juan Torre habla sobre la primavera. El libro fue publicado en Junio de 2009</p>
```

```
<p itemscope
    itemid="http://www.libros.com/primavera"
    itemtype="http://schema.org/Book">
En su último libro <span itemprop="name">La Primavera</span>,
<span property="author">Juan Torre</span>
habla sobre la primavera. El libro fue publicado el
<meta itemprop="datePublished"
      content="2009-06-03">3 de Junio de 2009</meta>.
</p>
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Ejemplo

```
<p>Me llamo Jose Luis Torre, nací el 1 de diciembre de 1974 y soy Profesor de la Universidad de Oviedo</p>
```

```
<div itemscope
    itemtype="http://schema.org/Person">
Me llamo <span itemprop="name">Jose Luis Torre</span>,
nací el
<time itemprop="date"
      datetime="1974-12-01">1 de diciembre de 1974</time>
y soy <span itemprop="jobTitle">Profesor</span>
de la
<span itemscope
    itemprop="affiliation"
    itemtype="http://schema.org/Organization">
<span itemprop="name">Universidad de Oviedo</span>
</span>
</div>
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Schema.org

Conjunto de vocabularios comunes adoptados por Google, Yahoo y Bing

Sigue la línea de sitemaps.org

Se basa en Microdatos

Modelo de datos jerárquico

Herencia universal de Thing

Cada clase contiene una serie de propiedades

Las subclases heredan las propiedades de las clases

----- Departamento de Informática, UTFSM ----- 2012/2013

Schema.org

Jerarquía (<http://schema.org/docs/full.html>)

Datatype

Boolean, Number (float,integer), Date, Text (url)

Thing (propiedades: name, description, image, url)

CreativeWork (Book, Movie, MusicRecording, Recipe, TVSeries, ...)

Event

Organization

Person

Place

Product

Review

----- Departamento de Informática, UTFSM ----- 2012/2013

Herramientas para microdatos

RDF-Translator:

<http://rdf-translator.appspot.com/>

Google Rich Snippets Tool:

<http://www.google.com/webmasters/tools/richsnippets>

LiveMicrodata:

<http://foolip.org/microdatajs/live/>

StructuredData.org

<http://linter.structured-data.org/>

Any23:

<http://any23.org/>

Sindice Inspector:

<http://sindice.com/developers/inspector>

----- Departamento de Informática, UTFSM ----- 2012/2013

Aplicaciones

Facebook Open Graph Protocol

Utiliza RDFa

Drupal 7: Soporte para RDFa

GoodRelations:

<http://www.heppnetz.de/projects/goodrelations/>

Posicionamiento semántico (Semantic SEO)

Originalmente RDFa, ahora admite Microdatos

Adoptado por Overstock, BestBuy, ...

LinkedOpenCommerce

<http://linkedopencommerce.com/>

----- Departamento de Informática, UTFSM ----- 2012/2013

GRDDL

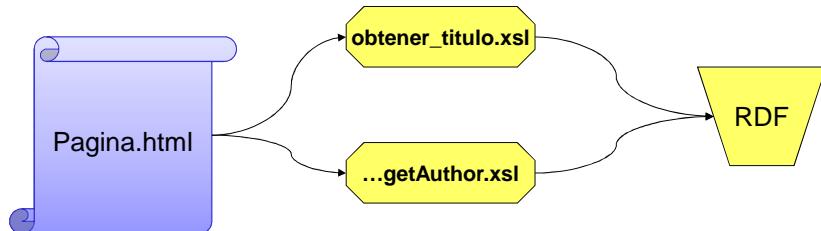
GRDDL = Gleaning Resource Descriptions over Dialects of Languages

Permite obtener información RDF a partir de ficheros XML/HTML
Se utiliza XSLT para definir una transformación de XML/HTML a RDF
Mecanismo para asociar transformaciones XSLT a tipos de documentos

----- Departamento de Informática, UTFSM ----- 2012/2013

Ejemplo en XHTML

```
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:grddl="http://www.w3.org/2003/g/data-view#"
      grddl:transformation="obtener_titulo.xsl"
      http://www.w3.org/2001/sw/grddl-wg/td/getAuthor.xsl" >
  <head>
    <title>Are You Experienced?</title>
    [...]
  </head>
```

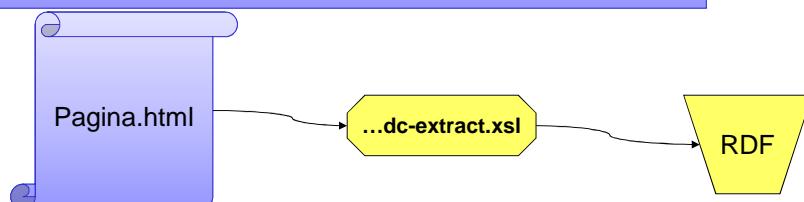


----- Departamento de Informática, UTFSM ----- 2012/2013

Ejemplo en HTML con DTDs

Se utiliza el atributo profile (ya existía en HTML 4.02)

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head profile="http://www.w3.org/2003/g/data-view">
  <title>Libros</title>
  <link rel="transformation"
        href="http://www.w3.org/2000/06/dc-extract/dc-extract.xsl" />
  <meta name="DC.Subject" content="Literatura" /> ...
</head>
...
</html>
```



----- Departamento de Informática, UTFSM ----- 2012/2013



Universidad Técnica Federico Santa María
Departamento de Informática



Ontologías y la Web Semántica

Jose Emilio Labra Gayo

Departamento de Informática
Universidad de Oviedo
<http://www.di.uniovi.es/~labra>

----- Departamento de Informática, UTFSM ----- 2012/2013

¿Qué es una Ontología?

Una ontología = Formalización de un dominio

Utiliza: lenguajes formales

Para: definir **vocabulario** de un dominio

Compartir el significado entre aplicaciones

Inferir nuevo conocimiento a partir de definiciones

Otros términos relacionados:

Taxonomía: Clasificación jerárquica

Tesoro: Definiciones de términos

----- Departamento de Informática, UTSFSM ----- 2012/2013

Ejemplos de dominios

Medicina

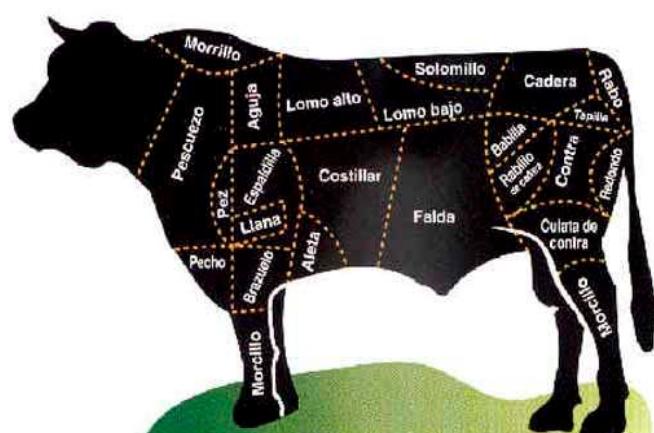
Biología

Aviación

Animales

Comida

...etc



----- Departamento de Informática, UTSFSM ----- 2012/2013

Partes de una ontología

Define conjunto de términos (vocabulario)

Corazón, Sangre, Sistema circulatorio

Propiedades entre dichos términos

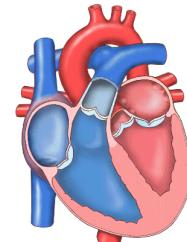
Ejemplo:

"el corazón **es un** órgano muscular que **es parte del** sistema circulatorio"

Describo en un lenguaje formal

Ejemplo (lógica):

$$\forall x(\text{Corazón}(x) \rightarrow \text{OrganoMuscular}(x) \wedge \exists y (\text{esParteDe}(x,y) \wedge \text{SistemaCirculatorio}(y)))$$



----- Departamento de Informática, UTFSM ----- 2012/2013

Ontología como rama del conocimiento

Ontología, rama de la metafísica

Desde Aristóteles (metafísica, IV)

Onto=ser, logos=estudio de (*estudio del ser*)

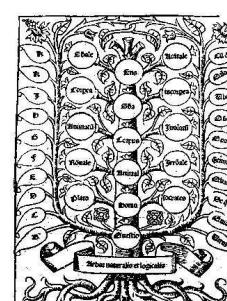
Estudia los entes, sus categorías y relaciones

Representación del conocimiento

Ontología = formalización de un dominio

Un **vocabulario compartido** que describe un determinado dominio

Un **conjunto de supuestos** sobre los términos de dicho vocabulario, generalmente se utiliza un **lenguaje formal** manipulable automáticamente.



Árbol de la naturaleza
y de la lógica
Ramón Llull (1235-1316)

----- Departamento de Informática, UTFSM ----- 2012/2013

Lógica

Lógica: Estudio de los razonamientos

Origen en Aristóteles (-342 a. de C.)

Desarrollo de la Lógica formal a finales s. XIX (De Morgan, Fregge)

Lógica computacional (Hilbert, Church, Turing, Herbrand, Tarski, ...)

Varios sistemas de Lógica

Lógica proposicional

Lógica de predicados

Otras lógicas: lógica modal, lógica descriptiva, lógica borrosa, etc.

----- Departamento de Informática, UTFSM ----- 2012/2013

Lógica Proposicional

Cada frase que puede ser verdadera o falsa es una proposición o enunciado (p)

Varias conectivas:

Negación:	$\neg p$
Conjunción:	$p \wedge q$
Disyunción:	$p \vee q$
Implicación:	$p \rightarrow q$
Equivalencia:	$p \leftrightarrow q$

“Si Juan juega al fútbol, se cansa y Juan juega al fútbol
Por tanto: Juan se cansa”

$$\frac{p \rightarrow q \\ p}{q}$$

----- Departamento de Informática, UTFSM ----- 2012/2013

Lógica proposicional

Existen sistemas de demostración que comprueban si un razonamiento es correcto

Ejemplo: Deducción natural

Propiedades:

Consistente: todos los razonamientos que se demuestran son correctos

Completo: todos los razonamientos correctos pueden demostrarse

Complejidad: NP (es uno de los problemas NP clásicos)

Expresividad: Muy poca.

Ejemplo: “*Todos los hombres son mortales, Sócrates es un hombre, luego Sócrates es mortal*”

----- Departamento de Informática, UTFSM ----- 2012/2013

Lógica de predicados

Extiende la lógica proposicional con predicados, funciones y cuantificadores

Ejemplo de predicado: $P(x,y) = x \text{ es padre de } y$

Ejemplo de función: $m(x) = \text{"madre de } x\text{"}$

Cuantificadores: Existencial: $\exists x P(x)$

Universal: $\forall x P(x)$

Ejemplo: “*Todos los hombres son mortales, Sócrates es un hombre, luego Sócrates es mortal*”

$$\forall x(H(x) \rightarrow M(x))$$

$$\frac{H(s)}{M(s)}$$

----- Departamento de Informática, UTFSM ----- 2012/2013

Lógica de Predicados

Existen varios sistemas de demostración en lógica de predicados.

Propiedades:

Consistente: Todo lo que demuestran es correcto

Completo: Todos lo que es correcto es demostrable

Semidecidible: Si una fórmula es correcta, lo detectan, si no lo es, pueden no detectarlo

Para resolver ese problema se han buscado **subconjuntos de lógica de predicados** de primer orden que sean decidibles:

Clausulas Horn

Lógica descriptiva

etc...

----- Departamento de Informática, UTFSM ----- 2012/2013

Complejidad

Los sistemas de demostración pueden ser muy complejos
Jerarquía de clases de complejidad

$$P \subseteq NP \subseteq PSPACE \subseteq EXPTIME \subseteq NEXPTIME \subseteq EXPSPACE$$

P = Problemas que resuelve una máquina de Turing determinista en tiempo polinómico

NP= Problemas que resuelve una máquina de Turing no determinista en tiempo polinómico

PSPACE= Problemas que resuelve una máquina de Turing determinista en espacio polinómico

EXPTIME=Problemas que resuelve una máquina de Turing determinista en tiempo $O(2^{p(n)})$

NEXPTIME=Problemas que resuelve una máquina de Turing no determinista en tiempo $O(2^{p(n)})$

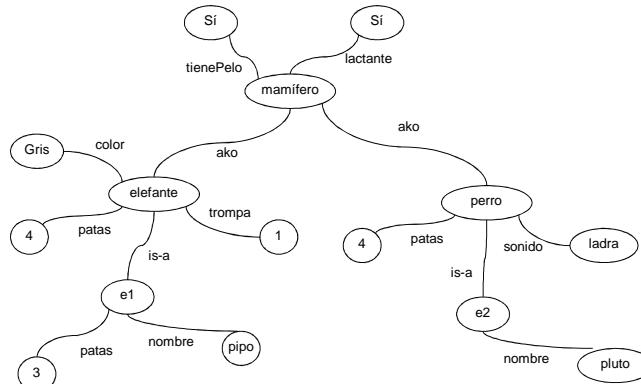
EXPSPACE=Problemas que resuelve una máquina de Turing determinista en espacio $O(2^{p(n)})$

----- Departamento de Informática, UTFSM ----- 2012/2013

Redes Semánticas

Redes Semánticas (Quillian, 68): Grafos dirigidos donde los vértices son conceptos y los enlaces son relaciones entre conceptos

2 tipos especiales de relaciones: is-a (pertenencia) y ako (inclusión)



----- Departamento de Informática, UTSFSM ----- 2012/2013

Frames

Desarrollados para estructurar el conocimiento de las redes semánticas

Un *frame* o marco = colección de atributos (slots) que describen una entidad

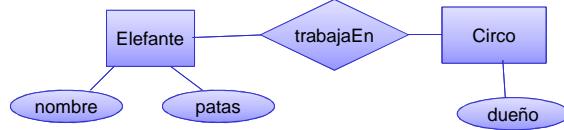
Puede representar un concepto (o clase) y un individuo (o instancia)

Clase: Mamífero tienePelo: Sí lactante: Sí	Individuo: e1 isa: Elefante patas: 3 nombre: Pipo
Clase: Elefante ako: Mamífero patas: 4 trompa: 1 color: gris	Individuo: e2 is-a: Perro nombre: Pluto
Clase: Perro: ako: Mamífero patas: 4 sonido: ladra	

----- Departamento de Informática, UTSFSM ----- 2012/2013

Diagramas Entidad-Relación

Diagramas Entidad-Relación (Chen, 1976):
Representaciones gráficas utilizadas para capturar
modelos de dominio.
Utilizados en el desarrollo de Bases de Datos



----- Departamento de Informática, UTSFSM ----- 2012/2013

Mapas de Tópicos (Topic Maps)

Mapas de tópicos (<http://www.topicmaps.org/>)
Estándar de definición de índices
XTM es un vocabulario para mapas de tópicos basado en XML

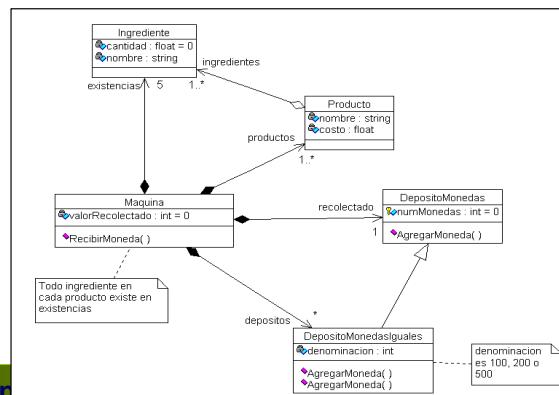
```
<topic id="pizzas"/> ...
<occurrence>
  <instanceOf>
    <topicRef xlink:href="#barbacoa"/>
  </instanceOf>
  <scope>
    <topicRef xlink:href="#pizza"/>
  </scope>
  <resourceRef xlink:href="barbacoa.jpg"/>
</occurrence>
...
</topic>
```

----- Departamento de Informática, UTSFSM ----- 2012/2013

Modelos Orientados a Objetos

Modelos Orientados a Objetos: Especificación de herencia y jerarquía de objetos

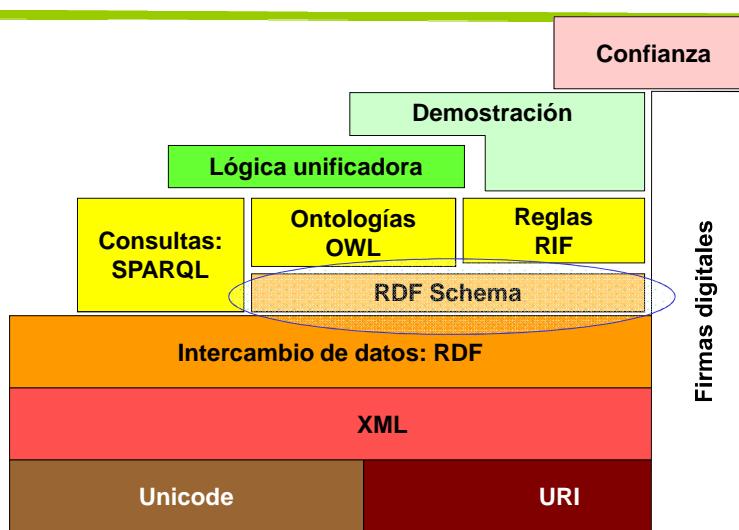
Lenguajes de modelado. UML incluye diagramas de clase que describen la estructura de objetos, atributos, operaciones, etc.



----- Departamento de Informática, UTSMS -----

2012/2013

RDF Schema



----- Departamento de Informática, UTSMS -----

2012/2013

RDF Schema Motivación

RDF permite establecer propiedades pero no dice nada acerca de las propiedades

RDF Schema tiene varias clases y propiedades predefinidas que permiten definir vocabularios.

Es permite definir:

- Clases y propiedades
- Jerarquías y herencia entre clases
- Jerarquías de propiedades

----- Departamento de Informática, UTFSM ----- 2012/2013

RDF Schema Clases e individuos

Hay que distinguir entre:

Cosas concretas (individuos) del dominio.

Ej. "Jose Labra", "Lógica"

Clases o conceptos = Conjuntos de individuos que comparten algunas propiedades
(rdfs:Class)

Ej. "Profesor", "Asignatura", "Estudiante", ...

rdf:type indica que un individuo pertenece a una clase

rdfs:subClassOf indica que una clase está incluida en otra

Nota

rdf:type = <<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>>
rdfs:Class = <<http://www.w3.org/2000/01/rdf-schema#Class>>

----- Departamento de Informática, UTFSM ----- 2012/2013

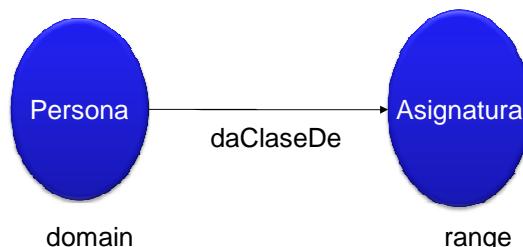
RDF Schema Rango y Dominio

Pueden declararse restricciones de Rango y Dominio

Ejemplo: `daClaseDe`

`rdfs:domain: Persona`

`rdfs:range: Asignatura`



RDF Schema Jerarquías

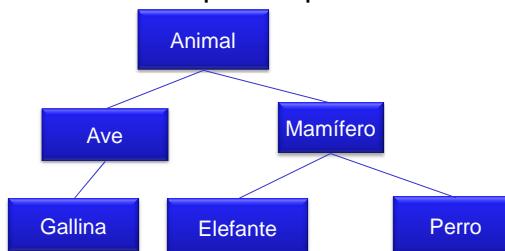
Las clases pueden organizarse en jerarquías

`rdfs:subClassOf` define que una clase es una subclase de otra

A es una subclase de B si todo individuo de A pertenece a B

Entonces, B es una superclase de A

Una clase puede tener múltiples superclases



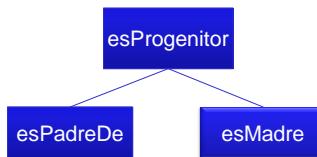
RDF Schema Jerarquía de Propiedades

Jerarquías entre propiedades **subPropertyOf**

Ej. Ser padre es una subpropiedad de ser progenitor

P es una subpropiedad de Q si y sólo si $P(x,y)$ entonces

$Q(x,y)$



----- Departamento de Informática, UTFSM ----- 2012/2013

RDF Schema: Inferencias

RDF Schema tiene una semántica predefinida que permite inferir nuevas declaraciones a partir de las existentes

En realidad, se genera un grafo nuevo a partir del grafo anterior

Ejemplos:

$X \text{ rdf:type } A \wedge A \text{ subClassOf } B \rightarrow X \text{ rdf:type } B$

$A \text{ rdfs:subClassOf } B \wedge B \text{ rdfs:subClassOf } C \rightarrow A \text{ rdfs:subClassOf } C$

$P \text{ rdfs:domain } A \wedge X \text{ P } Y \rightarrow X \text{ rdf:type } A$

$P \text{ rdfs:range } B \wedge X \text{ P } Y \rightarrow Y \text{ rdf:type } B$

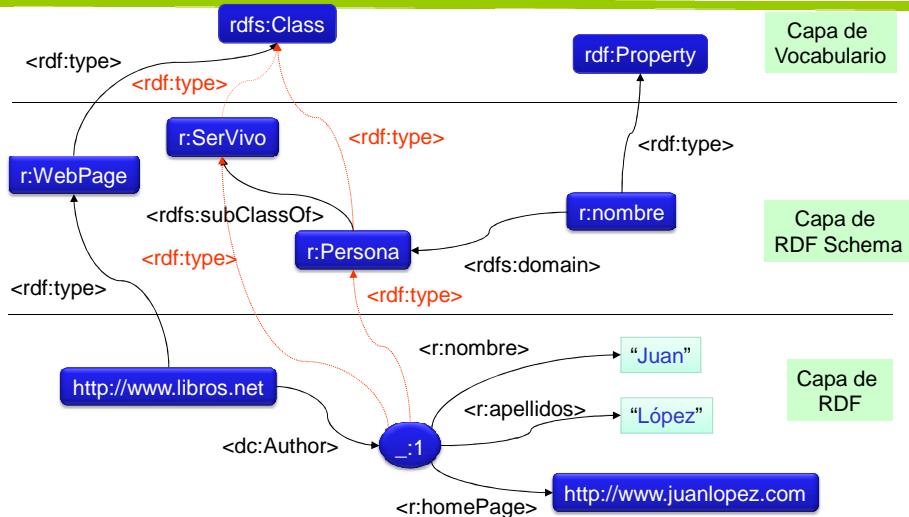
$P \text{ rdfs:subPropertyOf } Q \wedge Q \text{ rdfs:subPropertyOf } R \rightarrow P \text{ rdf:sSubpropertyOf } R$

$P \text{ rdfs:subPropertyOf } Q \wedge X \text{ P } Y \rightarrow X \text{ Q } Y$

etc.

----- Departamento de Informática, UTFSM ----- 2012/2013

RDF Schema: Inferencias



----- Departamento de Informática, UTFSM ----- 2012/2013

Ejercicio Barcos

Modelizar el siguiente conocimiento en RDF

Nombre	Primer Viaje	Proximo Viaje	Fecha Baja	Fecha Hundimiento	Comandante
Pisco	1913		1938		Olmos
Rambo	1969	2010			Gallardo
Titanic	1912			1912	Smith
Sauce	1980	2009			Torre

Expresar en RDF-S el siguiente conocimiento

Un barco está fuera de servicio si consta una fecha de baja o una fecha de hundimiento.

Un barco que tenga prevista un próximo viaje está en servicio.

El comandante de un barco es un marinero.

Un marinero es una persona

La fecha fin de servicio es la fecha de baja o la fecha de hundimiento.

----- Departamento de Informática, UTFSM ----- 2012/2013

Limitaciones de RDF Schema

RDF Schema sólo permite declarar información clases y propiedades

Es un primer paso hacia las ontologías pero se queda corto

Carece de expresividad para:

Información negativa (ej. Los hombres no son mujeres)

Cuantificadores (ej. Para que alguien sea considerado padre debe tener al menos un hijo)

Cardinalidad (ej. Un buen estudiante tiene que tener aprobadas más de 3 asignaturas)

No permite declarar atributos de propiedades (transitiva, simétrica, inversa, etc.)

----- Departamento de Informática, UTFSM ----- 2012/2013

Problema de RDF Schema

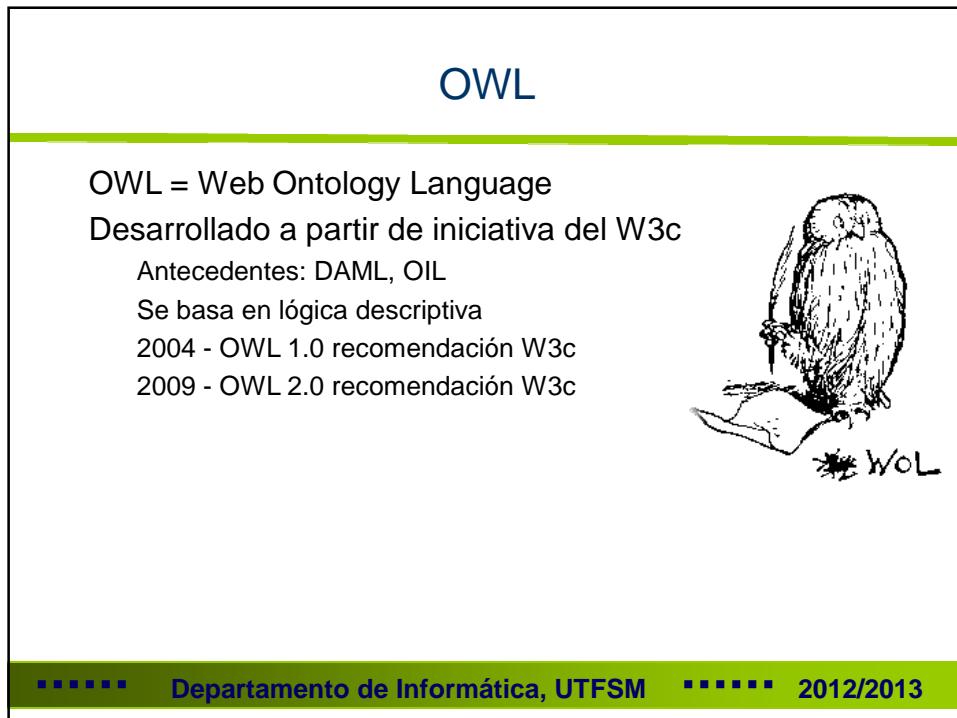
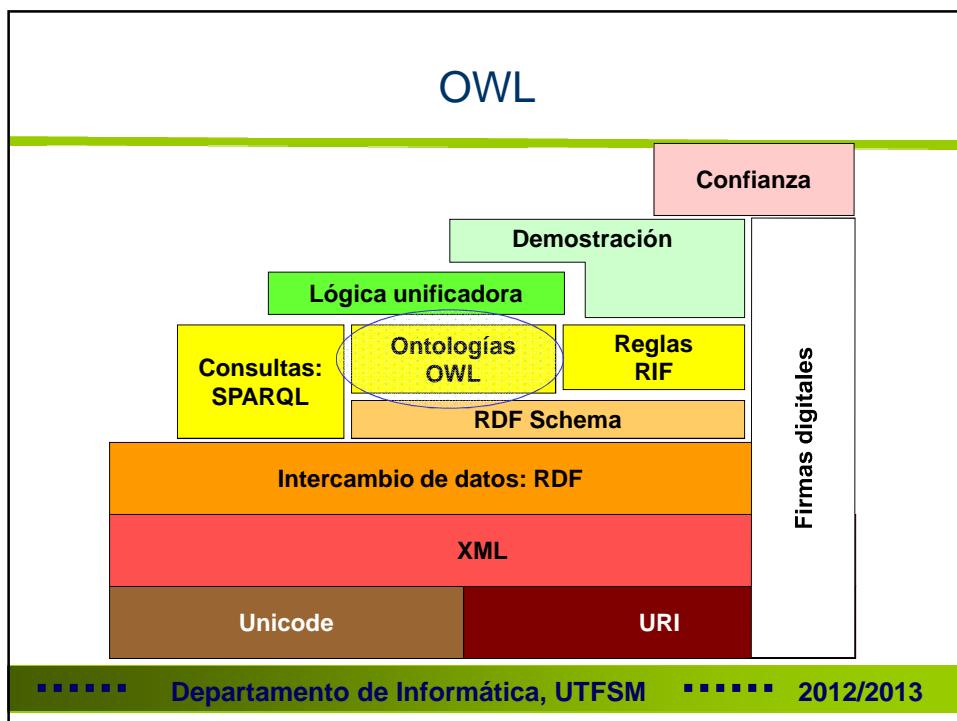
RDF Schema es demasiado liberal permitiendo mezclar clases con individuos y propiedades

Por ejemplo:

x rdf:type x

Pueden llegar a producirse paradojas en RDF Schema

----- Departamento de Informática, UTFSM ----- 2012/2013



Ejemplos de Ontologías

Cyc (<http://www.cyc.com>).

Conceptos de sentido común para Inteligencia Artificial
Utiliza lógica de predicados mediante lenguaje CycL

Frame Ontology y OKBC Ontology

Disponibles en Ontolingua (<http://www-ksl-svc.stanford.edu/>)
Utiliza KIF (Knowledge Interchange Format)

Ontologías en campos concretos:

Lingüística: WordNet (<http://www.globalwordnet.org/>)
Medicina: GALEN (<http://www.opengalen.org/>)
etc.

----- Departamento de Informática, UTFSM ----- 2012/2013

Ejemplos de Ontologías Dublin Core

Dublin Core Metadata Initiative (<http://www.dcmi.org>)

Utilizado para la catalogación de documentos

Espacio de nombres: <http://purl.org/dc/elements/1.1/>

Conjunto de elementos básicos cuyo significado es compartido

Contenido: Coverage, Description, Type, Relation, Source, Subject, Title

Propiedad Intelectual: Contributor, Creator, Publisher, Rights

Instanciación: Date, Format, Identifier, Language

Cada elemento básico admite una serie de cualificadores

Refinamiento de elementos

Ejemplo: Date.created, Description.tableOfContents

Esquema de codificación

Ejemplos: Identifier.URI, Date.DCMIPeriod

----- Departamento de Informática, UTFSM ----- 2012/2013

Evolución de las Ontologías para la Web

SHOE (Simple HTML Ontology Extensions) Univ. Maryland, 1996

Permite definir ontologías en documentos HTML

Objetivo = Facilitar búsquedas y anotaciones de documentos

OIL (Ontology Inference Layer)

Sintaxis RDF(S) y primitivas de representación del conocimiento en marcos

Se basa en el uso de *description logics*

DAML (DARPA Agent Markup Language)

Proyecto americano de creación de lenguaje para ontologías

DAML-OIL. Proyecto conjunto que será la base de OWL

OWL (Web Ontology Language) desarrollado en W3C (2004)

----- Departamento de Informática, UTFSM ----- 2012/2013

OWL

OWL (Web Ontology Language)

Desarrollado por el consorcio W3C (2004)

3 niveles:

OWL Full. Unión de sintaxis OWL y RDF (sin restricciones)

No se garantiza la eficiencia ni siquiera la decidibilidad

OWL DL (Description Logics). Limita la expresividad intentando conseguir decidibilidad

Profiles. Subconjuntos de OWL DL: EL, QL, RL, etc.

Más eficientes, menos expresivos

----- Departamento de Informática, UTFSM ----- 2012/2013

OWL

OWL DL se basa en Lógica Descriptiva (Description Logics)

En realidad equivale al formalismo $SHOIN(D_n)$

Características

Semántica bien definida

Propiedades formales (decidibilidad, complejidad)

Algoritmos de razonamiento conocidos

Varios razonadores (Pellet, HermiT, Quonto)

Incluye tipos de datos primitivos de XML Schema

----- Departamento de Informática, UTFSM ----- 2012/2013

Lógica Descriptiva

La lógica descriptiva consiste en:

Conceptos (o clases):

Ejemplo: Padre, Madre, Persona

Propiedades (o roles): Relaciones entre conceptos

Ejemplo: tieneHijo, esPadreDe

Individuos: Elementos del dominio

Ejemplo: Juan, Sergio, ...

----- Departamento de Informática, UTFSM ----- 2012/2013

Lógica Descriptiva

La lógica descriptiva es un subconjunto de la lógica de primer orden

Características:

Sólo se usan predicados de un argumento (clases) y dos argumentos (propiedades)

El uso de las variables está restringido

----- Departamento de Informática, UTFSM ----- 2012/2013

Lógica Descriptiva

La base de conocimiento contiene 2 niveles

Términos (TBox): Descripción de conceptos

Padre ≡ Persona \cap \exists tieneHijo Persona

Orgulloso ≡ Persona \cap \exists tieneHijo ReciénNacido

ReciénNacido \subseteq Persona

Aserciones (ABox): Descripción de individuos

ReciénNacido(Sergio)

tieneHijo(Jose,Sergio)

Persona(Jose)

----- Departamento de Informática, UTFSM ----- 2012/2013

Lógica descriptiva

Las expresiones en lógica descriptiva pueden representarse en lógica de primer orden

Padre ≡ Persona $\cap \exists$ tieneHijo Persona

$$\forall x(\text{Padre}(x) \leftrightarrow (\text{Persona}(x) \wedge \exists y(\text{tieneHijo}(x,y) \wedge \text{Persona}(y)))$$

Orgulloso ≡ Persona $\cap \exists$ tieneHijo ReciénNacido

$$\forall x(\text{Orgulloso}(x) \leftrightarrow (\text{Persona}(x) \wedge \exists y(\text{tieneHijo}(x,y) \wedge \text{RecienNacido}(y)))$$

ReciénNacido \subseteq Persona

$$\forall x(\text{RecienNacido}(x) \rightarrow \text{Persona}(x))$$

----- Departamento de Informática, UTFSM ----- 2012/2013

Lógica Descriptiva Definición de Conceptos

Definición de conceptos

Equivalencia: $C \equiv D$

Ejemplo: Asturiano ≡ NacidoEnAsturias

Subclase: $C \subseteq D$ (C está incluido en D ó D subsume a C)

Ejemplo: Asturiano \subseteq Español

Intersección: $C \cap D$

Ejemplo: Mujer ≡ Persona \cap Femenino

Unión: $C \cup D$

Ejemplo: Persona ≡ Hombre \cup Mujer

Complemento: $\neg C$

Ejemplo: Masculino ≡ \neg Femenino

Concepto vacío: \perp

Clases Disjuntas: $C \cap D \equiv \perp$

----- Departamento de Informática, UTFSM ----- 2012/2013

Lógica Descriptiva Cuantificadores

Descripción de Propiedades

Existencial ($\exists R C$)

x pertenece a $\exists R C$ si existe algún valor $y \in C$ tal que $R(x,y)$

Ejemplo: $\text{Madre} \equiv \text{Mujer} \cap \exists \text{ tieneHijo Persona}$

Universal ($\forall R C$)

x pertenece a $\forall R C$ si para todo y , si $R(x,y)$ entonces $y \in C$

Ejemplo: $\text{MadreFeliz} \equiv \text{Madre} \cap \forall \text{ tieneHijo Sano}$

Una Madre es feliz si todos sus hijos están sanos

NOTA: Si no tuviese hijos, también se cumpliría...

----- Departamento de Informática, UTFSM ----- 2012/2013

Lógica Descriptiva Cardinalidades

Cardinalidad ($P = n$)

x pertenece a ($P = n$) si existen n $y \in C$ tales que $R(x,y)$

Ejemplo: $\text{Elefante} \subseteq \text{Animal} \cap \text{tienePatas} = 4$

Cardinalidad máxima ($P \leq n$)

x pertenece a ($P \leq n$) si existen n ó menos $y \in C$ tales que $R(x,y)$

Ejemplo: $\text{MalEstudiante} \equiv \text{Estudiante} \cap \text{tieneAprobada} \leq 3$

Cardinalidad mínima ($P \geq n$)

x pertenece a ($P \geq n$) si existen n ó más $y \in C$ tales que $R(x,y)$

Ejemplo: $\text{BuenEstudiante} \equiv \text{Estudiante} \cap \text{tieneAprobada} \geq 3$

$$\begin{aligned} & \forall x (\text{BuenEstudiante}(x) \leftrightarrow \text{Estudiante}(x) \wedge \\ & \exists y_1 \exists y_2 \exists y_3 (\text{tieneAprobada}(x, y_1) \wedge \\ & \quad \text{tieneAprobada}(x, y_2) \wedge \\ & \quad \text{tieneAprobada}(x, y_3) \wedge \\ & \quad y_1 \neq y_2 \wedge y_2 \neq y_3 \wedge y_1 \neq y_3)) \end{aligned}$$

----- Departamento de Informática, UTFSM ----- 2012/2013

Lógica Descriptiva

Atributos de propiedades

Reflexiva: P es reflexiva $\Rightarrow \forall x P(x,x)$

Ejemplo: `viveCon` es reflexiva

Irreflexiva: P es irreflexiva $\Rightarrow \forall x \neg P(x,x)$

Ejemplo: `esPadreDe` es irreflexiva

Simetría. Si $P(x,y)$ entonces $P(y,x)$

Ejemplo: `viveCon`

Asimétrica. Si $P(x,y)$ entonces $\neg P(y,x)$

Ejemplo: `esPadreDe`

Transitividad. Si $P(x,y)$ y $P(y,z)$ entonces $P(x,z)$

Ejemplo: `viveCon`

----- Departamento de Informática, UTFSM ----- 2012/2013

Lógica descriptiva

Relaciones entre propiedades

Inversa: P es inversa de $Q \Rightarrow P(x,y) \Leftrightarrow Q(y,x)$

Ejemplo: `daClaseDe` es inversa de `tieneProfesor`

SubPropiedad: P subpropiedad de Q si $P(x,y) \Rightarrow Q(x,y)$

Ejemplo: `esHijoDe` es subpropiedad de `esDescendienteDe`

----- Departamento de Informática, UTFSM ----- 2012/2013

Lógica Descriptiva Funcionalidad

Propiedad Funcional.

$P(x,y) \wedge P(x,z) \rightarrow y = z$

Ejemplo: [edad](#)

Propiedad Funcional inversa.

$P(x,y) \wedge P(z,y) \rightarrow x = z$

Ejemplo: [dni](#)

Claves. similares a las propiedades func. inversas

$P(x,y) \wedge P(z,y) \rightarrow x = z$.

Se definen en OWL 2 (específicas para una clase)

Ejemplo: [dni](#)

----- Departamento de Informática, UTFSM ----- 2012/2013

Lógica Descriptiva Razonamiento

A partir de una base de conocimiento Σ se ofrecen varios mecanismos de inferencia:

1.- Satisfacibilidad de conceptos: De Σ no se deduce que $C \equiv \perp$

Ejemplo: [Orgulloso](#) \cap [ReciénNacido](#)

2.- Subsunción: Deducir si un concepto está incluido en otro

$\Sigma \Rightarrow C \subseteq D$

Ejemplo: [Orgulloso](#) \subseteq [Padre](#)

Padre \equiv Persona \cap \exists tieneHijo Persona
Orgulloso \equiv Persona \cap \exists tieneHijo ReciénNacido
ReciénNacido \subseteq Persona
Padre \subseteq \neg ReciénNacido

ReciénNacido(Sergio)
tieneHijo(Jose,Sergio)
Persona(Jose)

----- Departamento de Informática, UTFSM ----- 2012/2013

Lógica Descriptiva Razonamiento

3.- Instanciación: $\Sigma \rightarrow a \in C$

Ejemplo: Orgulloso(Jose)

4.- Recuperación de Información

Dado un concepto C, obtener a tales que $a \in C$

Ejemplo: ? Orgulloso

Jose

5.- Realización/Comprensión (realizability).

Dado un elemento a, obtener concepto más específico C tal que $a \in C$

Ejemplo: ? jose

Orgulloso

Padre ≡ Persona $\cap \exists$ tieneHijo Persona
Orgulloso ≡ Persona $\cap \exists$ tieneHijo ReciénNacido
ReciénNacido \subseteq Persona
Padre $\subseteq \neg$ ReciénNacido

ReciénNacido(Sergio)
tieneHijo(Jose, Sergio)
Persona(Jose)

----- Departamento de Informática, UTFSM ----- 2012/2013



Universidad Técnica Federico Santa María
Departamento de Informática



Lenguaje OWL

----- Departamento de Informática, UTFSM ----- 2012/2013

OWL

Definición de ontologías

Aunque OWL es un lenguaje basado en lógica descriptiva, existen varias sintaxis:

RDF (N3 ó RDF/XML)

Sintaxis abstracta

Sintaxis Manchester

Espacio de nombres:

<http://www.w3.org/2002/07/owl#>

Cabecera

Declarar una URI de la clase owl:Ontology

```
@prefix : <http://ejemplo.org#>.  
@prefix owl: <http://www.w3.org/2002/07/owl#>.  
..... <> a owl:Ontology .
```

13

OWL

Cabecera

En la cabecera pueden incluirse diversas anotaciones

Anotaciones posibles:

rdfs:label	Etiqueta del recurso
rdfs:comment	Comentario del recurso
owl:versionInfo	Información sobre la versión del recurso
rdfs:seeAlso	Indica otro recurso con más información
rdfs:isDefinedBy	Indica que otro recurso contiene la definición
owl:imports	Indica la URI de una ontología que se va a incorporar a la ontología actual

```
@prefix : <http://ejemplo.org#>.  
@prefix owl: <http://www.w3.org/2002/07/owl#>.  
..... : a owl:Ontology ;  
       rdfs:comment "Ejemplo de Ontología" ;  
       owl:imports <http://paises.org> .
```

/2013

OWL Definiciones básicas

```
:juan a :Alumno ;
      :nombre "Juan Manuel" ;
      :apellidos "Gallardo" ;
      :esAmigoDe :pepe .

:pepe a :Alumno ;
      :nombre "Jose Luis" ;
      :apellidos "Torres" ;
      :tieneProfesor :jj .

:jj a :Profesor ;
     :nombre "Juan Jose" ;
     :apellidos "Bravo" .
```

----- Departamento de Informática, UTFSM ----- 2012/2013

OWL Subclases

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.

:Profesor rdfs:subClassOf :Persona .
:Alumno rdfs:subClassOf :Persona .
```

Existen 2 clases predefinidas:
owl:Thing - Contiene a todos los individuos
owl:Nothing - representa el conjunto vacío

----- Departamento de Informática, UTFSM ----- 2012/2013

OWL Propiedades

2 tipos de propiedades:

ObjectProperty - relaciona individuos

Ejemplo: [:daClaseDe](#)

DatatypeProperty - relaciona un individuo con un valor

Ejemplo: [nombre](#), [edad](#)

----- Departamento de Informática, UTFSM ----- 2012/2013

OWL Dominio y rango

Las definiciones de dominio y rango se toman de RDF Schema

Dominio: Conjunto inicial

Rango: Conjunto final

```
:daClaseDe rdfs:domain :Profesor .  
:daClaseDe rdfs:range   :Asignatura .
```

Nota: Las definiciones de dominio y rango pueden introducir poca flexibilidad en el modelo

----- Departamento de Informática, UTFSM ----- 2012/2013

Relaciones entre propiedades

rdfs:subPropertyOf: P es subpropiedad de Q, si $P(x,y) \Rightarrow Q(x,y)$

```
:esPadreDe rdfs:subPropertyOf :esProgenitor .
```

owl:inverseOf: P es inversa de Q si $P(x,y) \Rightarrow Q(y,x)$

```
:esPadreDe owl:inverseOf :esHijoDe .
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Propiedades disjuntas

P es disjunta de Q si $P(x,y) \Rightarrow \neg Q(x,y)$

Puede definirse propiedad a propiedad:

```
:tienePadre owl:propertyDisjointWith :tieneMadre .
```

...o para varias propiedades a la vez:

```
[] a owl:AllDisjointProperties ;  
owl:members ( :tienePadre ,  
:tieneMadre ) .
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Tipos de propiedades

owl:SymmetricProperty: $P(x,y) \Rightarrow P(y,x)$

Ejemplo: viveCon

owl:ASymmetricProperty: $P(x,y) \Rightarrow \neg P(y,x)$

Ejemplo: esPadreDe

owl:ReflexiveProperty: Para todo x , $P(x,x)$

Ejemplo: viveCon

owl:IrreflexiveProperty: Para todo x , $\neg P(x,x)$

Ejemplo: esPadreDe

owl:TransitiveProperty: $P(x,y) \wedge P(y,z) \Rightarrow P(x,z)$

Ejemplo: viveCon

:viveCon a owl:SymmetricProperty,
owl:ReflexiveProperty,
owl:TransitiveProperty.

12/2013

Propiedades funcionales

owl:FunctionalProperty: $P(x,y) \wedge P(x,z) \Rightarrow y = z$

Ejemplo: tieneMadre

owl:InverseFunctionalProperty: $P(x,y) \wedge P(z,y) \Rightarrow x = z$

Ejemplos: esMadreDe, dni

:tieneMadre a owl:FunctionalProperty.

owl:hasKey: Define una clave para una clase.

Una clave es un conjunto de valores de propiedades que identifican únicamente a un elemento.

:Persona owl:hasKey (:dni :nombre).

----- Departamento de Informática, UTFSM ----- 2012/2013

Restricciones de propiedades Existencial

owl:someValuesFrom: Al menos un valor debe pertenecer a una clase

```
:Padre owl:equivalentClass [  
    a owl:Restriction ;  
    owl:onProperty :tieneHijo ;  
    owl:someValuesFrom :Persona  
].
```

Notación en lógica descriptiva: $\text{Padre} \equiv \text{Persona} \cap \exists \text{ tieneHijo Persona}$

Notación en lógica de predicados: $\forall x(\text{Padre}(x) \leftrightarrow (\text{Persona}(x) \wedge \exists y(\text{tieneHijo}(x,y) \wedge \text{Persona}(y)))$

----- Departamento de Informática, UTFSM ----- 2012/2013

Restricciones de propiedades Universal

owl:allValuesFrom: Todos los valores deben pertenecer a una clase

```
[] a owl:Class ;  
    owl:intersectionOf ( :Person :Feliz) ;  
    owl:equivalentClass [ a owl:Restriction ;  
        owl:onProperty :hasChild ;  
        owl:allValuesFrom :Feliz ].
```

Notación en lógica descriptiva:

$\text{Persona} \cap \text{Feliz} \equiv \forall \text{ tieneHijo Feliz}$

Notación en lógica de predicados: $\forall x(\text{Persona}(x) \wedge \text{Feliz}(x) \leftrightarrow \forall y(\text{tieneHijo}(x,y) \rightarrow \text{Feliz}(y)))$

----- Departamento de Informática, UTFSM ----- 2012/2013

Restricciones de propiedades

Valores

owl:hasValue: Debe contener un valor determinado

```
:HijosDeJuan owl:equivalentClass
[ a owl:Restriction ;
  owl:onProperty :esHijoDe ;
  owl:hasValue :Juan
].
```

Notación en
lógica de predicados:

$$\forall x(\text{HijosDeJuan}(x) \leftrightarrow \text{esHijoDe}(x, \text{Juan}))$$

----- Departamento de Informática, UTFSM ----- 2012/2013

Cardinalidades

owl:cardinality: N valores exactos

owl:minCardinality: Al menos N valores

owl:maxCardinality: N valores como mucho

```
:Persona owl:equivalentClass
[ a owl:Restriction ;
  owl:onProperty :esHijoDe ;
  owl:cardinality "2"^^xsd:nonNegativeInteger ;
].
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Cardinalidades cualificadas

owl:qualifiedCardinality: N valores exactos de una clase

owl:minQualifiedCardinality: Al menos N valores de una clase

owl:maxQualifiedCardinality: N valores como mucho de una clase

```
:DosHijosDoctores owl:equivalentClass
[ a owl:Restriction ;
  owl:onProperty :esPadreDe ;
  owl:qualifiedCardinality "2"^^xsd:nonNegativeInteger ;
  owl:onClass :Doctor
].
```

Conjuntos por enumeración

owl:oneOf define un conjunto de valores

```
:Estacion a owl:Class ;
owl:OneOf (
:Primavera
:Otonio
:Verano
:Invierno
) .
```

Operaciones de conjuntos

`owl:unionOf` define la unión de varias clases

```
:Progenitor owl:equivalentClass  
[ a owl:Class ;  
  owl:unionOf ( :Padre :Madre )  
 ].
```

`owl:intersectionOf` define la intersección de unas clases

```
:Madre owl:equivalentClass [ a owl:Class ;  
  owl:intersectionOf ( :Progenitor :Mujer )  
 ].
```

`owl:complementOf`

```
:PersonaSinHijos owl:equivalentClass  
[ a owl:Class ;  
  owl:intersectionOf ( :Persona [ owl:complementOf :Progenitor ] )  
 ].
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Clases disjuntas

`owl:disjointWith`

```
:Persona owl:disjointWith :Asignatura .  
:Asignatura owl:disjointWith :Ciudad .  
:Ciudad owl:disjointWith :Asignatura .
```

`owl:allDisjointClasses`

```
[] a owl:AllDisjointClasses ;  
  owl:members (:Persona :Asignatura :Ciudad)
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Equivalencia

`owl:sameAs` establece que 2 individuos son el mismo
`owl:equivalentClass` establece que 2 clases son la misma

`owl:equivalentProperty` establece que 2 propiedades son la misma

----- Departamento de Informática, UTFSM ----- 2012/2013

Individuos diferentes

`owl:differentFrom`

```
:Juan owl:differentFrom :Pepe .  
:Juan owl:differentFrom :Luis .  
:Luis owl:differentFrom :Pepe .
```

`owl:AllDifferent`

```
[] a owl:AllDifferent ;  
  owl:distinctMembers (:Juan :Pepe :Luis) .
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Declaraciones negativas

owl:negativePropertyAssertion permite declarar que no se cumple una propiedad
Hay que indicar la propiedad, el sujeto y el objeto.

Ejemplo: Declara que Juan no es padre de Ana

```
[] a owl:NegativePropertyAssertion ;  
    owl:sourceIndividual :Juan ;  
    owl:assertionProperty :esPadreDe ;  
    owl:targetIndividual :Ana .
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Cadenas de propiedades

owl:propertyChainAxiom define una cadena de propiedades

```
:esAbueloDe owl:propertyChainAxiom  
  ( :esPadreDe  
    :esPadreDe  
  ) .  
  
:esTioDe owl:propertyChainAxiom  
  ( :esHermanoDe  
    :esPadreDe  
  ) .
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Restricciones self

Define la clase de todos los individuos que se relacionan consigo mismos mediante una propiedad

```
:Autonomo a owl:Class;
owl:equivalentClass [
  a owl:SelfRestriction ;
  owl:onProperty :alimentaA
].
```

----- Departamento de Informática, UTFSM ----- 2012/2013



Universidad Técnica Federico Santa María
Departamento de Informática



OWL Sintaxis XML

----- Departamento de Informática, UTFSM ----- 2012/2013

OWL Sintaxis XML

OWL se basa en RDF (utiliza sintaxis XML de RDF)
También existen otras formas sintácticas más sencillas
Las ontologías comienzan por owl:Ontology

```
<owl:Ontology rdf:about="http://www.uniovi.es/ontologia_1.1">
  <rdfs:comment>Ejemplo de Ontología</rdfs:comment>
  <owl:priorVersion
    rdf:resource="http://www.uniovi.es/ontologia_1.0"/>
  <owl:imports
    rdf:resource="http://www.uniovi.es/personas"/>
  <rdfs:label>Ontología de la Universidad</rdfs:label>
</owl:Ontology>
```

owl:imports es una propiedad transitiva

----- Departamento de Informática, UTFSM ----- 2012/2013

Clases en OWL

Las clases se definen mediante owl:Class

owl:Class es una subclase de rdfs:Class

Clases equivalentes mediante equivalentClass

```
<owl:Class rdf:id="Profesor">
  <owl:equivalentClass rdf:resource="#PersonalDocente"/>
</owl:Class>
```

owl:Thing es la clase más general

owl:Nothing es la clase vacía

Las clases disjuntas se definen mediante owl:disjointWith

```
<owl:Class rdf:about="#ProfesorAsociado">
  <owl:disjointWith rdf:resource="#catedrático"/>
  <owl:disjointWith rdf:resource="#titular"/>
</owl:Class>
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Propiedades en OWL

2 tipos de propiedades

Propiedades de Objetos relacionan un objeto con otro objeto. ej.
"esHijoDe"

```
<owl:ObjectProperty rdf:id="esHijoDe">
  <owl:domain rdf:resource="#Persona"/>
  <owl:range rdf:resource="#Persona"/>
  <rdfs:subPropertyOf rdf:resource="#esDescendienteDe"/>
</owl:ObjectProperty>
```

Propiedades de tipos de datos relacionan un objeto con valores de tipos de datos
(enteros, literales, etc.), ej. "edad"

Habitualmente, se utilizan los tipos de datos de XML Schema

```
<owl:DatatypeProperty rdf:id="edad">
  <rdfs:range
    rdf:resource="http://www.w3.org/2001/XMLSchema#nonNegativeInteger"/>
</owl:DatatypeProperty>
```

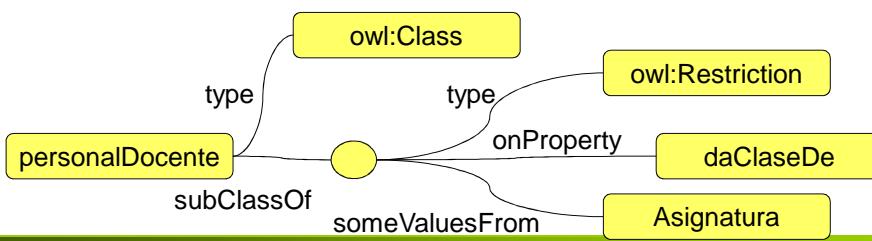
----- Departamento de Informática, UTFSM ----- 2012/2013

Definición de Clases

Clases como restricciones de propiedades

```
<owl:Class rdf:about="#personalDocente">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#daClaseDe"/>
      <owl:someValuesFrom rdf:resource="#Asignatura"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

personalDocente ⊆ ∃ daClaseDe Asignatura



----- Departamento de Informática, UTFSM ----- 2012/2013

Propiedades en OWL Restricciones

`allValuesFrom` (\forall) indica que todos los valores deben ser de un tipo
NOTA: Los que no tiene ningún valor, también cumplen la condición
`someValuesFrom` (\exists) Al menos un valor de la propiedad debe tener
un tipo

Ejemplo: Un estudiante es una persona que cursa al menos una
asignatura

`hasValue` Al menos uno de los valores tiene un valor

`minCardinality`, `maxCardinality` restringen el número máximo/mínimo
de valores

----- Departamento de Informática, UTFSM ----- 2012/2013

Propiedades en OWL Combinaciones booleanas

Combinaciones booleanas

`complementOf`, `unionOf`, `intersectionOf`

```
<owl:Class rdf:id="personasUniversidad">
    <owl:unionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#personalDocente"/>
        <owl:Class rdf:about="#estudiantes"/>
        <owl:Class rdf:about="#PAS"/>
    </owl:unionOf>
</owl:Class>
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Propiedades en OWL

Enumeraciones

oneOf permite realizar enumeraciones

```
<owl:oneOf rdf:parseType="Collection">
    <owl:Thing rdf:about="#Lunes"/>
    <owl:Thing rdf:about="#Martes"/>
    <owl:Thing rdf:about="#Miércoles"/>
    <owl:Thing rdf:about="#Jueves"/>
    <owl:Thing rdf:about="#Viernes"/>
    <owl:Thing rdf:about="#Sábado"/>
    <owl:Thing rdf:about="#Domingo"/>
</owl:oneOf>
```

----- Departamento de Informática, UTFSM ----- 2012/2013

Individuos en OWL

Se declaran igual que en RDF

```
<rdf:Description rdf:ID="jose">
    <rdf:type rdf:resource= "#profesor"/>
</rdf:Description>
```

```
<personalDocente rdf:ID="jose">
    <uni:edad rdf:datatype="&xsd;integer">35<uni:edad>
</personalDocente>
```

----- Departamento de Informática, UTFSM ----- 2012/2013

OWL – Web semántica

No asume nombres únicos

Web = modelo abierto

Información incompleta
2 URLs diferentes podrían identificar el mismo objeto
No soporta UNA (Unique name assumption)
Permite **inferir** que 2 elementos son iguales
No está pensado para validar modelos

Ejemplo

```
Persona ⊑ tienePadre = 1
tienePadre(luis,jose)
tienePadre(luis,pepe)
Persona(luis)
```

No indica error en el modelo

Infiere que "pepe" y "jose" son iguales

----- Departamento de Informática, UTFSM ----- 2012/2013

OWL – Web Semántica

Asumción de mundo abierto

Web = Sistema abierto

Sistemas tradicionales usaban *closed world assumption*
En OWL se usa *open world assumption*

Ejemplo

```
Soltero ⊑ ¬ ∃ estaCasadoCon Persona
Casado ⊑ ∃ estaCasadoCon Persona
Person(a(pepe))
Person(a(Maria))
Person(a(luis))
estaCasadoCon(maria,pepe)
Casado(luis )
```

El sistema infiere que
María está casada

El sistema no infiere que
pepe esté casado ni soltero

El sistema infiere que luis
Está casado con alguien...
pero no sabe con quién.

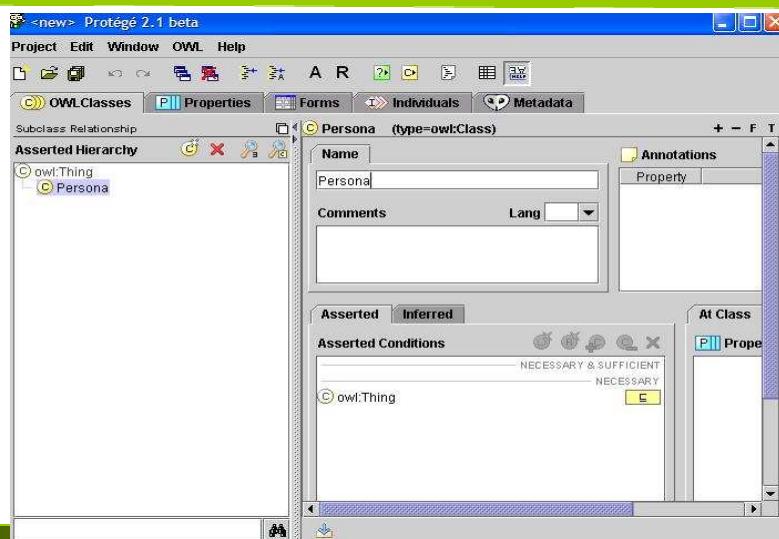
----- Departamento de Informática, UTFSM ----- 2012/2013

OWL Herramientas

Herramientas para manipulación de documentos OWL
Protègè (<http://protege.stanford.edu>) es una herramienta para creación de ontologías desarrollada en Stanford (se basa en Frames)
Arquitectura que facilita el desarrollo de plugins
Plugin para edición de documentos OWL
Swoop: Herramienta inspirada en un visualizador web con la posibilidad de editar ontologías
TopBraid (Comercial)

----- Departamento de Informática, UTFSM ----- 2012/2013

OWL Herramientas



Ejercicio. Resolver el Problema de Einstein en OWL

Problema propuesto por Einstein y traducido a varios idiomas conservando su lógica. Einstein aseguraba que el 98% de la población mundial sería incapaz de resolverlo.

Condiciones iniciales:

Tenemos cinco casas, cada una de un color.

Cada casa tiene un dueño de nacionalidad diferente.

Los 5 dueños beben una bebida diferente, fuman marca diferente y tienen mascota diferente.

Ningún dueño tiene la misma mascota, fuma la misma marca o bebe el mismo tipo de bebida que otro.

Datos:

1. El noruego vive en la primera casa, junto a la casa azul.
2. El que vive en la casa del centro toma leche.
3. El inglés vive en la casa roja.
4. La mascota del Sueco es un perro.
5. El Danés bebe té.
6. La casa verde es la inmediata de la izquierda de la casa blanca.
7. El de la casa verde toma café.
8. El que fuma PallMall cría pájaros.
9. El de la casa amarilla fuma Dunhill.
10. El que fuma Blend vive junto al que tiene gatos.
11. El que tiene caballos vive junto al que fuma Dunhill.
12. El que fuma BlueMaster bebe cerveza.
13. El alemán fuma Prince.
14. El que fuma Blend tiene un vecino que bebe agua.

¿Quién tiene peces por mascota?

***** Departamento de Informática, UTFSM ***** 2012/2013

OWL Sistemas de Inferencia

HermiT (Java) razonador OWL 2

Pellet (Java) incluye razonador para OWL 2

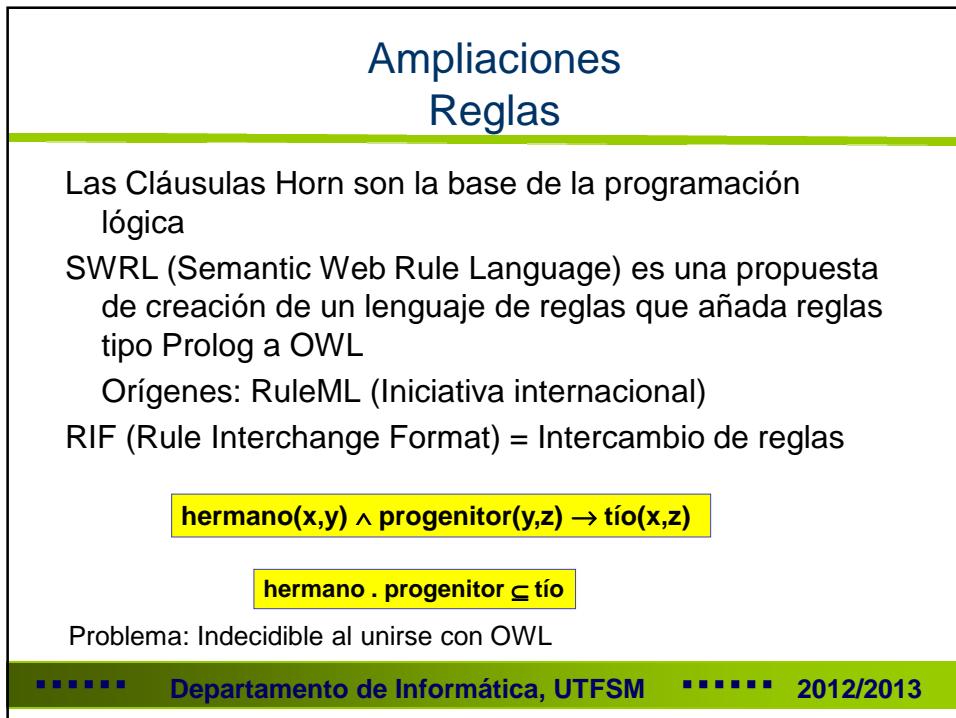
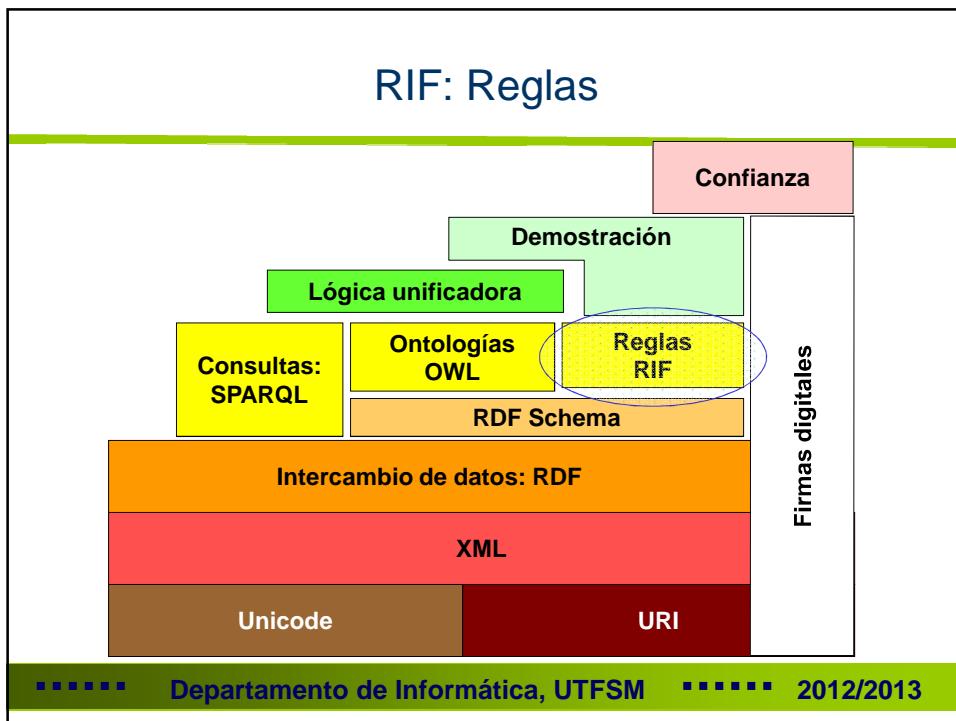
Soporte en línea de comandos o mediante interfaz DIG

Fact++ (C++) razonador

RACER. Sistema de inferencia implementado en Lisp

JENA. API Java para RDF. Incluye sistema de inferencia

***** Departamento de Informática, UTFSM ***** 2012/2013



Monotonía y Reglas

Problema: negación por fallo

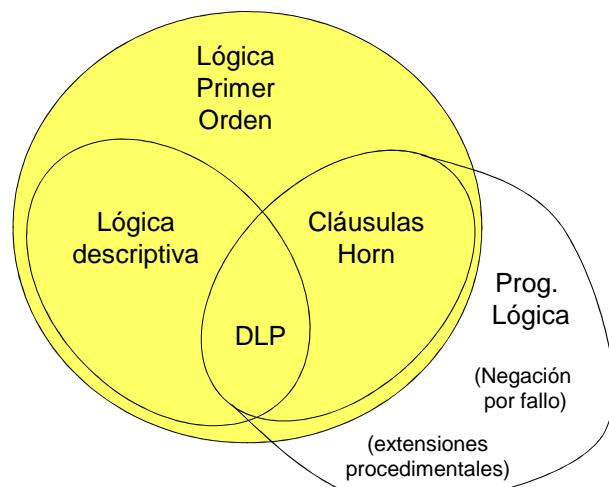
Lógica de primer orden es monótona

Negación por fallo no es monótona

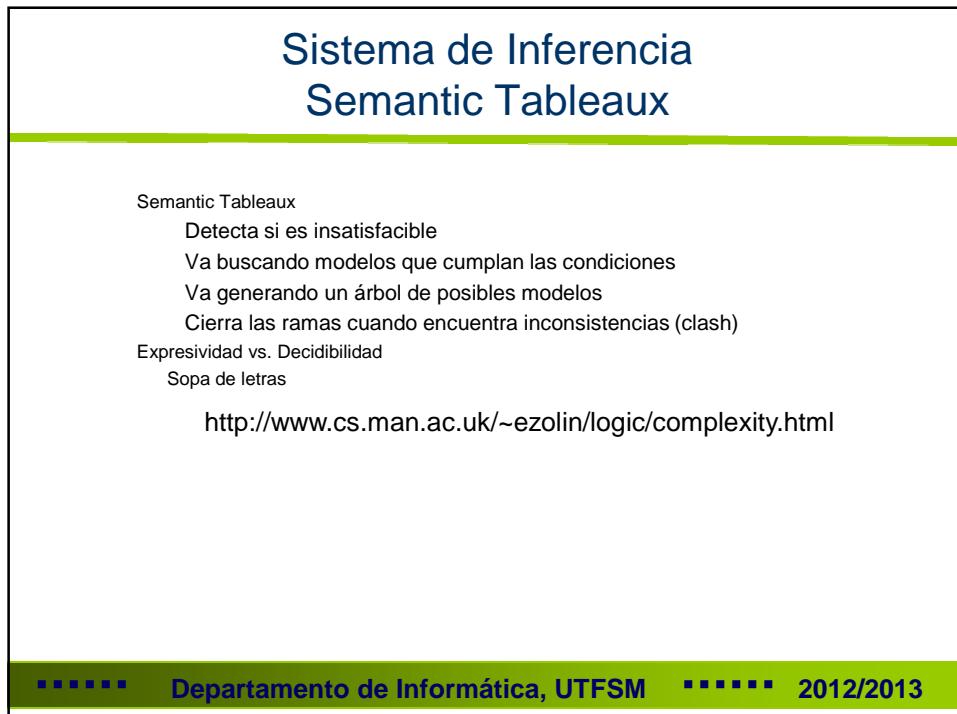
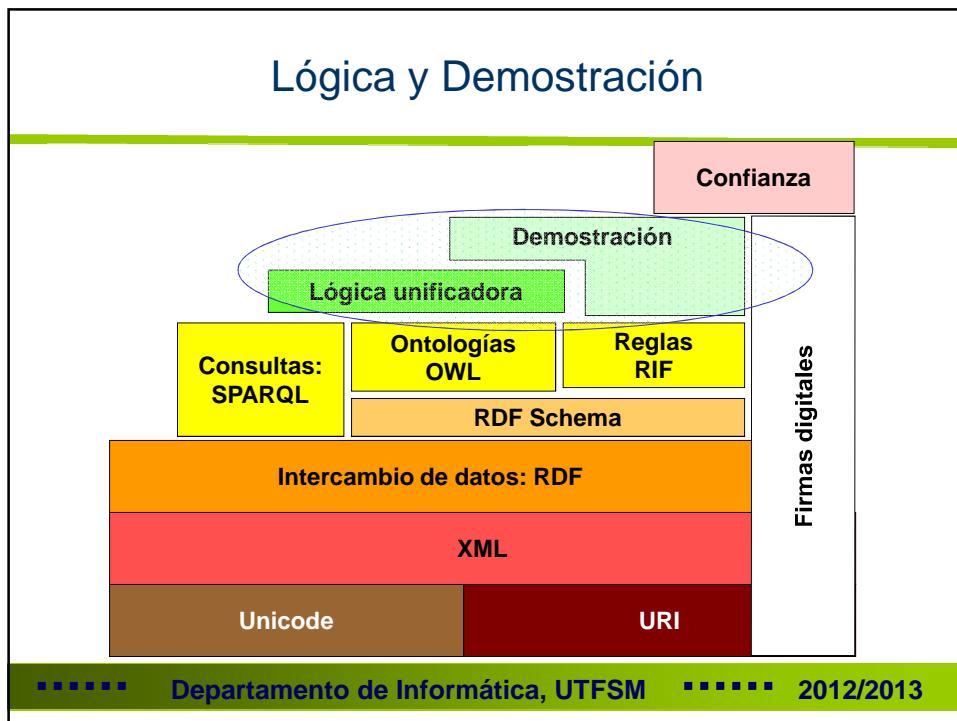
Programación lógica con negación por fallo no es un subconjunto de lógica de primer orden

----- Departamento de Informática, UTFSM ----- 2012/2013

Interacción entre programación lógica y lógica descriptiva



----- Departamento de Informática, UTFSM ----- 2012/2013



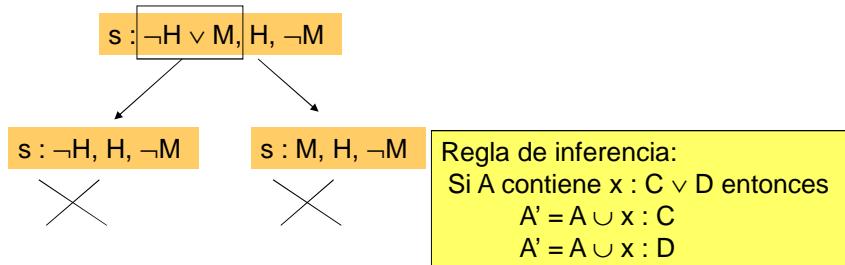
Semantic Tableaux Ejemplo

Hombre \subseteq Mortal

Hombre(Sócrates)
Mortal(Sócrates)

Razonamiento: $\{ H \subseteq M, H(s) \} \Rightarrow M(s)$

Forma normal: $\{ \neg H \vee M, H(s), \neg M(s) \}$



----- Departamento de Informática, UTFSM ----- 2012/2013

Semantic Tableaux Algunas Reglas de inferencia

Si A contiene $x : C \vee D$ entonces
 $A' = A \cup x : C$
 $A' = A \cup x : D$

Si A contiene $x : C \wedge D$ entonces
 $A' = A \cup x : C, D$

Si A contiene $x : \exists R C y \nexists z$ tal que $R(x,z) y z : C$ entonces
 $A' = A \cup \{ y : C, R(x,y) \}$ para un $y \notin A$

Si A contiene $x : \forall R C y R(x,y)$ pero no contiene $y : C$
 $A' = A \cup y : C$

----- Departamento de Informática, UTFSM ----- 2012/2013

Semantic Tableaux

Otro ejemplo

$\{ \exists \text{ hijo Persona} \subseteq \text{Padre}, \text{Padre} \subseteq \text{Persona} \} \Rightarrow \exists \text{ hijo Persona} \subseteq \text{Persona}$

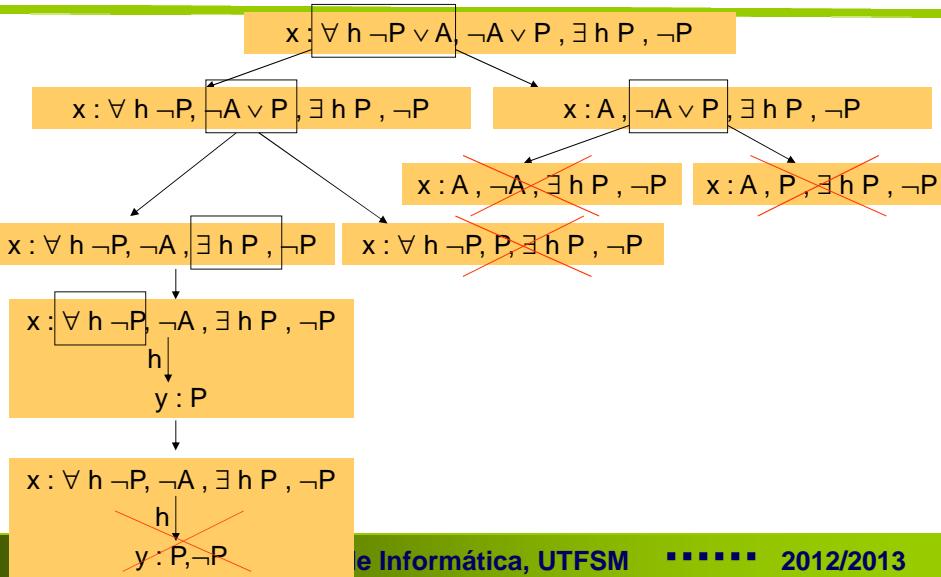
Cambiando nombres: $\{ \exists h P \subseteq A, A \subseteq P \} \Rightarrow \exists h P \subseteq P$

Forma normal: $\{ \forall h \neg P \vee A, \neg A \vee P, \exists h P, \neg P \}$

----- Departamento de Informática, UTSFSM ----- 2012/2013

Semantic Tableaux

Otro ejemplo



Semantic Tableaux Otro ejemplo

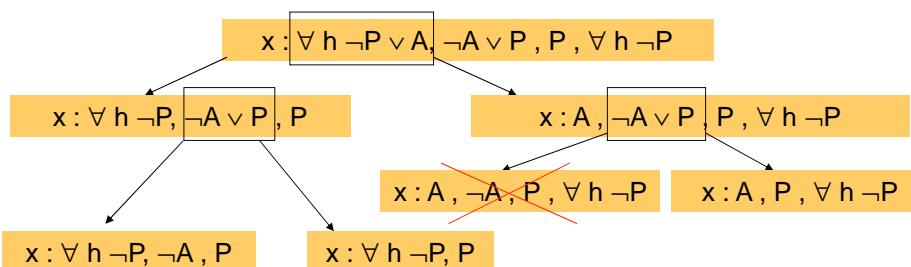
$\{ \exists \text{ hijo Persona} \subseteq \text{Padre}, \text{Padre} \subseteq \text{Persona} \} \Rightarrow \text{Persona} \subseteq \exists \text{ hijo Persona}$

Cambiando nombres: $\{ \exists h P \subseteq A, A \subseteq P \} \Rightarrow P \subseteq \exists h P$

Forma normal: $\{ \forall h \neg P \vee A, \neg A \vee P, P, \forall h \neg P \}$

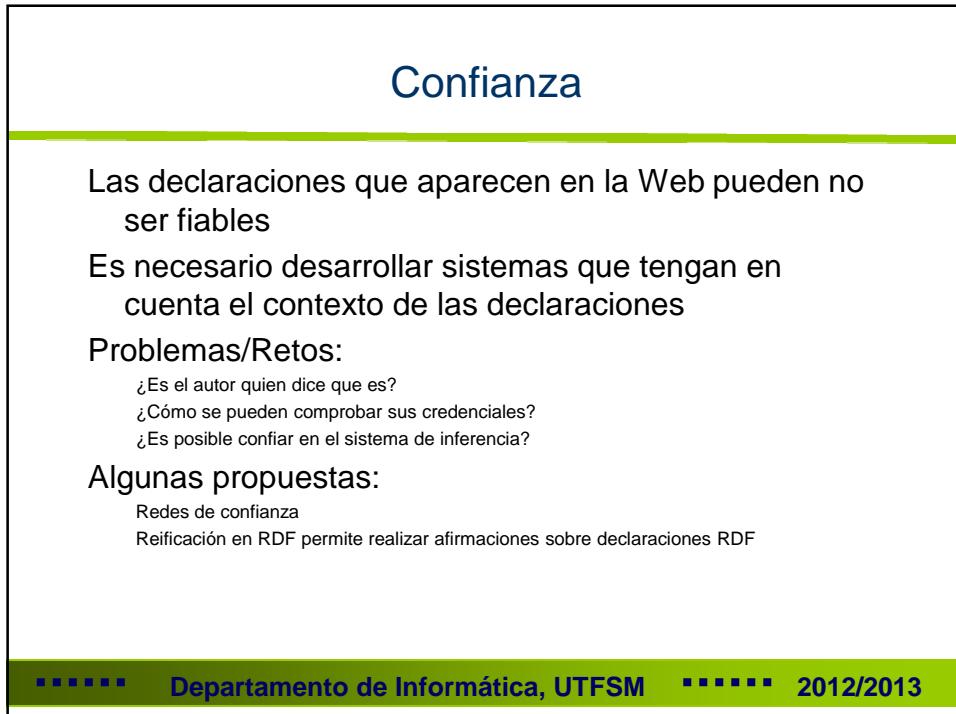
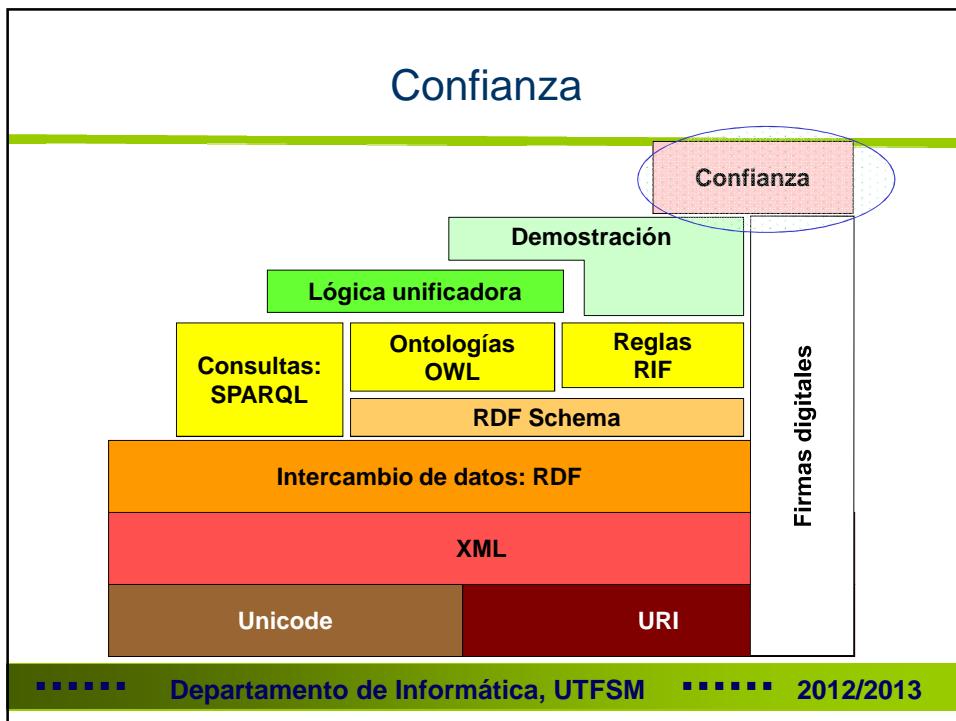
----- Departamento de Informática, UTFSM ----- 2012/2013

Semantic Tableaux Otro ejemplo



Se encuentra un modelo \Rightarrow No se cumple

----- Departamento de Informática, UTFSM ----- 2012/2013



El futuro de la Web Semántica

La Web Semántica está de modapuede ser un problema...

Compromiso Expresividad vs Eficiencia

Razonamiento con individuos limitado

Complejidades exponenciales

Aplicaciones de muestra rudimentarias

Necesidad de una *Killer Application*

Generación de meta-information

Representación de meta-information

Depuración de ontologías

¿Y la confianza?

Inclusión de Técnicas de certificación

Explicación de Respuestas (D. McGuiness)

----- Departamento de Informática, UTFSM ----- 2012/2013

Fin de la Presentación



----- Departamento de Informática, UTFSM ----- 2012/2013