Web Semántica y datos enlazados (y abiertos)

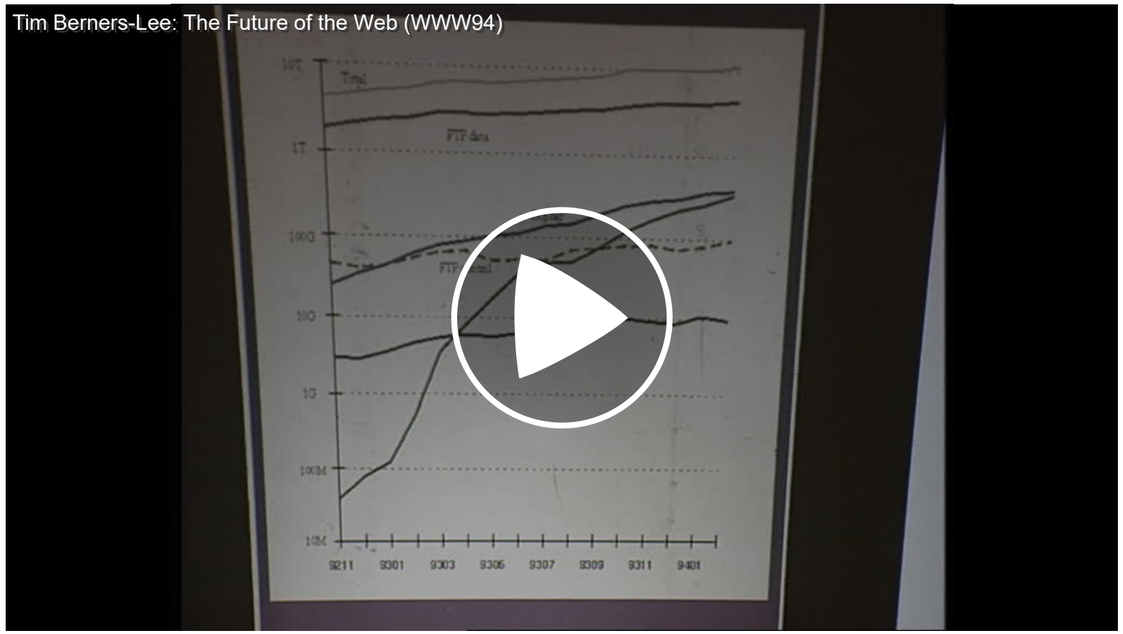
Daniel Gayo Avello

Última modificación: Mon, 15 May 2023 09:33:06 GMT

Tabla de contenidos

Introducción

- Tim Berners-Lee hizo la primera mención a la semántica en la Web en su ponencia plenaria en el primer Congreso de la Web (WWW94). (A partir del minuto 3:30 en el vídeo)



- Esa idea original sería refinada en 1998 en "Semantic Web Road map" y recibiría el impulso definitivo con el artículo publicado en *Scientific American* *"The Semantic Web"*.

Introducción

- Un aspecto crucial de la Web de documentos es que sólo puede ser interpretada por los humanos, no por las máquinas.
- Veamos el siguiente documento:

> J'nEbñjfm Hbzp-Bwfmmp, bñ bttpdjbuf qspgfttps jñ uif Efqbsunfñu pg Dpnqvufs Tdjfñdf bu uif Vñjwfstjuz pg Pwjfep. Nz nbjñ bsfb pg jñufsftu jt Xfc JS cvu J'n dvssfñumz gpdvtfe pñ tpdjbm nfejb sftfbsdi. J ibwf qvcmjtife jñ jñufsñbujpñbm dpñgfsfñdft, kpvsñbmt bñe nbhbajñft, tvdi bt Dpnnvñjdbujpñt pg uif BDN, JFFF Jñufsñfu Dpnqvujñh, ps JFFF Nvmujnfejb. Jñ 2013 J bdufe bt hvftu dp-fejups gps b tqfdjbm ittvf pg Jñufsñfu Sftfbsdi pñ uif qsfeiduiwf qpxfs pg tpdjbm nfeib. J ibwf

- Esa es la "visión" que tiene una máquina de un texto plano; es posible que haya un sentido subyacente pero no puede extraerse de manera inmediata. Con acceso a más texto plano podrían extraerse cadenas significativas pero seguiría sin haber una semántica accesible para la máquina.

Introducción

- Veamos el siguiente hipertexto:

> J'nEbñjfm Hbzp-Bwfmmp, bñ bttpdjbuf qspgfttps jñ uif Efqbsunfñu pg Dpnqvufs Tdjfñdf bu uif Vñjwfstjuz pg Pwjfep. Nz nbjñ bsfb pg jñufsftu jt Xfc JS cvu J'n dvssfñumz gpdvtfe pñ tpdjbm nfejb sftfbsdi. J ibwf qvcmjtife jñ jñufsñbujpñbm dpñgfsfñdft, kpvsñbmt bñe nbhbajñft, tvdi bt *Dpnnvñjdbujpñt pg uif BDN*, *JFFF Jñufsñfu Dpnqvujñh*, ps *JFFF Nvmujnfejb*. Jñ 2013 J bdufe bt hvftu dp-fejups gps b tqfdjbm jttvf pg *Jñufsñfu Sftfbsdi* pñ uif qsfejdujwf qpxfs pg tpdjbm nfejb. J ibwf

- Esa sería la "visión" de una máquina de un **hipertexto básico**. Sigue sin disponer del significado de ese texto pero sí "sabe" que las siguientes cadenas son importantes y van, además, asociadas a una URL:
  - bsfb pg jñufsftu (https://danigayo.prof/research)
  - tpdjbm nfejb sftfbsdi (https://danigayo.prof/research/#socialmediaresearch)
  - J ibwf qvcmjtife (https://danigayo.prof/publications)
  - JFFF Jñufsñfu Dpnqvujñh (https://danigayo.prof/publications/detail.php?id=...)
  - JFFF Nvmujnfejb (https://danigayo.prof/publications/detail.php?id=...)
  - tqfdjbm jttvf pg Jñufsñfu Sftfbsdi pñ uif qsfejdujwf qpxfs pg tpdjbm nfejb (http://www.emeraldinsight.com/toc/intr/23/5)
  - dibqufs pñ Qpmjujdbm Pqjñjpñ (https://danigayo.prof/publications/detail.php?id=...)

Introducción

- Con independencia de lo que signifique ese texto o las URLs un buscador "sabría" que debería asociarlo como metadatos a las URLs indicadas ya que tal vez pueda ser útil para resolver consultas; después de todo, ya hay al menos un sitio donde se hace referencia a esas URLs con esos términos.
- Veamos ahora el siguiente documento:

> J'nEbñjfm Hbzp-Bwfmmp, bñ bttpdjbuf qspgfttps jñ uif Efqbsunfñu pg Dpnqvufs Tdjfñdf bu uif Vñjwfstjuz pg Pwjfep. Nz nbjñ bsfb pg jñufsftu jt Xfc JS cvu J'n dvssfñumz gpdvtfe pñ tpdjbm nfejb sftfbsdi. J ibwf qvcmjtife jñ jñufsñbujpñbm dpñgfsfñdft, kpvsñbmt bñe nbhbajñft, tvdi bt *Dpnnvñjdbujpñt pg uif BDN*, *JFFF Jñufsñfu Dpnqvujñh*, ps *JFFF Nvmujnfejb*. Jñ 2013 J bdufe bt hvftu dp-fejups gps b tqfdjbm jttvf pg *Jñufsñfu Sftfbsdi* pñ uif qsfeiduiwf qpxfs pg tpdjbm nfejb. J ibwf

- Se trata en apariencia del mismo hipertexto anterior pero hay diferencias muy importantes. Si se analiza dicho documento con la herramienta *Schema Markup Validator* encontramos lo siguiente.
- Así, gracias a los **microdatos incrustados en el HTML**, una máquina descubriría:
  - 4 publicaciones periódicas,
  - un capítulo de libro,
  - una persona que trabaja en un departamento de una universidad.

Introducción

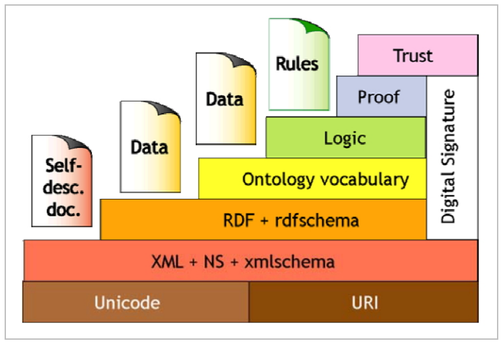
- Según la propia definición del W3C los microdatos son una extensión del lenguaje HTML para incrustar metadatos. Son similares aunque menos potentes que RDFa y JSON-LD.
- Los microdatos se basan en el uso de vocabularios, prefiriéndose la reutilización de vocabularios existentes antes de la creación de vocabularios nuevos.
- La colección de vocabularios más usada es la de Schema.org que ya ha cumplido 10 años y cuenta con el apoyo de las principales compañías de buscadores Web (Bing, Google, Yahoo! y Yandex).
- En las prácticas de laboratorio trabajaréis con microdatos y JSON-LD, en las próximas sesiones de teoría veremos los fundamentos conceptuales sobre los que se sustenta la Web Semántica (y los datos enlazados) vinculándolos con distintos estándares.

Introducción

- ¿RDFa, JSON-LD, vocabularios?
- Desde fases muy tempranas se planteó la Web Semántica como una (cada vez más) compleja pila de tecnologías que, en última instancia, permitirían explotar los datos de la Web como si se tratase de una base de datos.

< Prev

La pila de la Web Semántica en 2000



Next >

- Las tecnologías cruciales en esa visión de la Web Semántica son RDF y SPARQL, así como OWL y SKOS (ambos basados en RDF, más: ¿Qué relación y diferencias hay entre OWL y SKOS?)
- Eso no significa que sean tecnologías con un altísimo grado de implantación en la Web actual pero está cambiando... A modo de ilustración comparemos uso de RDFa y JSON-LD a lo largo del tiempo y veamos qué uso se hace de esos y otros formatos estructurados de datos (algunos no son estándares de la Web Semántica).

Introducción

- Además de los distintos mecanismos para ofrecer semántica sin recurrir a datos estructurados (p.ej., los microdatos HTML5) hubo intentos aún más laxos como las **folksonomías**.
- Ese término hace referencia a estructuras organizativas que surgen de manera espontánea al facilitar el etiquetado colaborativo de recursos *online*.
- El ejemplo arquetípico es el sitio web de marcadores Delicious pero, en principio, cualquier plataforma que permita etiquetar libremente recursos puede dar lugar a una folksonomía.
- Un sitio que aún usa activamente folksonomías es Bibsonomy que permite a investigadores y científicos gestionar sus referencias bibliográficas.
  - Veamos un par de ejemplos: [1] y [2].
  - Como se puede ver, las etiquetas son muy diversas, en ocasiones describen el *paper* (p.ej., `information-retrieval` o `link-mining`) pero en otras se usan para indicar el uso que el usuario pretende darle (p.ej., `diploma_thesis` o `eventually_useful`), **no hay jerarquía** alguna, **no hay control de sinónimos** (p.ej., `web` y `www` o `semanticweb`, `semantic_web`, `semantic-web` o `semweb`) y tampoco hay un mecanismo que permita solventar las ambigüedades.
- Esos y otros problemas hacen que el uso de las folksonomías sea limitado pero, debido a su sencillez, aún se siguen ofreciendo posibilidades de etiquetado similares en muchas plataformas, p.ej., ***hashtags*** en diversos medios sociales, etiquetas en *Flickr*, *keywords* en *IMDb*, o estanterías en *Goodreads*.

La Web Semántica ha muerto, ¡viva la Web Semántica!

- No lo digo yo... Lo dijo Hal Abelson en 2005 y aquí:

> *A lot of the enthusiasm around the SemWeb reminds me of the AI hullabaloo of the 1980s. Moreover, much of the technology seems little different from the knowledge representation work invented in that era.*
>
> *Over the past 20 years, AI researchers have come to appreciate the limitations of traditional knowledge representation techniques. It seems that statistical methods and machine learning have proven more productive than reasoning based on ontologies. Thereís even some evidence that people perform common recognition based on simple feature matching, rather than using classification-based reasoning.*
>
> *In light of this, it's interesting that major web applications like Google and Flickr are increasingly relying on statistical and clustering methods, while "big S" SemWeb technologies are encountering resistance from the folksonomy communities with claims that RDF and OWL are too complex and "unnatural".*

La Web Semántica ha muerto, ¡viva la Web Semántica!

- No lo digo yo... Lo dijeron Nigel Shadbolt, Wendy Hall y Tim Berners-Lee en 2006:

> *The Semantic Web is a Web of actionable information—information derived from data through a semantic theory for interpreting the symbols. The semantic theory provides an account of "meaning" in which the logical connection of terms establishes interoperability between systems. [...] This simple idea, however, remains largely unrealized.*

> *The Scientific American article assumed that this would be straightforward, but it's still difficult to achieve in today's Web.*

> *Because we haven't yet delivered large-scale, agent-based mediation, some commentators argue that the Semantic Web has failed to deliver. We argue that agents can only flourish when standards are well established [...]*

La Web Semántica ha muerto, ¡viva la Web Semántica!

- No lo digo yo... Lo dijo Jim Hendler en 2006:

> *To me one of the most exciting things going on in the Semantic Web is that we're starting to see the "A little Semantics goes a long way" starting to come true — this thing that is being called "Web 3.0" is, if you look at it and squint a bit, the side of the Semantic Web that is looking at how do you use a small amount of Sem Web (think Foaf or Skos) to add a bit of organizational knowledge (and to webize with URIs) to tagging sites, microformats, and etc.*
>
> *It is the realization that the REST approach to the world is a wonderful way to use RDF and it is enpowered by the emerging standards of SPARQL, GRDDL, RDF/A and the like.*
>
> ***In short, it is the Semantic Web vision of Tim's, before Ora and I polluted it with all this ontology stuff, coming real!***

La Web Semántica ha muerto,
¡viva la Web Semántica!

- No lo digo yo... Lo dijo Mor Naaman en 2007, literalmente, *"the Semantic Web is dead"*.
- Lo dijo Grigoris Antoniou también en 2007:

> *The semantic web may fail but semantic web technologies will stay.*

La Web Semántica ha muerto, ¡viva la Web Semántica!

- No lo digo yo... Lo dijo Grigoris Antoniou en 2007:

> *The semantic web may fail but semantic web technologies will stay.*

- Ejemplos:
  - Google está impulsando el uso de JSON-LD para disponer de datos estructurados y el *Knowledge Graph* demuestra que los grafos de conocimiento son cruciales para explotar la Web como un sistema de información (véase).
  - Facebook también tiene un API basada en grafos para representar toda la información almacenada en la plataforma. Además, gracias al *Open Graph Protocol* cualquier página web puede etiquetarse con metadatos (relativos a un vocabulario limitado) que facilitan su integración en dicho grafo (y esos metadatos pueden ser explotados por cualquier otro servicio). Por otro lado, lanzó en 2015 GraphQL, un lenguaje de consultas basado en grafos para consumir APIs. GraphQL no persigue los mismos objetivos que SPARQL pero se están buscando formas de integrarlo con RDF (véase también HyperGraphQL o *An Introduction To GraphQL*).
  - SPARQL es uno de los lenguajes de consulta en grafos que se va a analizar para desarrollar el futuro estándar GQL
  - Schema.org es una iniciativa de las compañías de buscadores para disponer de vocabularios estandarizados para usar con datos estructurados y que, entre otros formatos, se puede descargar en RDFS. La relación entre Schema.org y Open Graph se discute brevemente aquí.
- En resumen, conocer algo de RDF y SPARQL puede resultar muy interesante.
- Para tener una visión global del campo os recomiendo estos dos artículos: *A Survey of the First 20 Years of Research on Semantic Web and Linked Data* y *A Review of the Semantic Web Field*.

RDF

- RDF significa **R**esource **D**escription **F**ramework.
- Los **recursos** pueden ser cualquier cosa que tenga una IRI (pincha aquí para saber más sobre IRIs): por supuesto páginas HTML, pero también personas, organizaciones, países, conceptos, etc.
- Por **descripción** entendemos atributos, características y relaciones de y entre los recursos.
- ***Framework*** hace referencia a los modelos, lenguajes y sintaxis empleados para realizar dichas descripciones.
- RDF es un modelo basado en **tripletas** puesto que todos los elementos del conocimiento pueden expresarse como (**sujeto**, **predicado**, **objeto**), p.ej. (**France**, **capital**, **Paris**) o (**EEUU**, **presidente**, **Joe Biden**).
- RDF es, además, un modelo en grafo que **enlaza** las descripciones de los recursos. Así, las tripletas serían aristas en un grafo de conocimiento: (**vértice**, **arista**, **vértice**).

RDF

- En RDF todos los elementos de la tripleta (subjeto, predicado y objeto) suelen ser IRIs:
  - (**France**, **capital**, **Paris**) podría expresarse como (**https://www.wikidata.org/wiki/Q142**, **https://www.wikidata.org/wiki/Property:P36**, **https://www.wikidata.org/wiki/Q90**).
  - (**EEUU**, **presidente**, **Joe Biden**) podría expresarse como (**https://www.wikidata.org/wiki/Q30**, **https://www.wikidata.org/wiki/Property:P6**, **https://www.wikidata.org/wiki/Q6279**).
- Los valores de las propiedades (i.e., el objeto), pueden ser literales (i.e., cadenas de caracteres). P.ej. la tripleta (**Daniel Gayo Avello**, **nombre**, **"Daniel Gayo Avello"**) podría representarse como (**https://orcid.org/0000-0002-4705-6891**, **http://xmlns.com/foaf/0.1/name**, **"Daniel Gayo Avello"**)

RDF

- RDF tiene múltiples formatos de serialización:
  - Turtle, un formato compacto y bastante amigable. El que usaremos, a partir de este punto, para los ejemplos.
  - N-Triples, un formato muy simple, fácil de parsear y basado en líneas pero que no es tan compacto como Turtle.
  - N-Quads, un superconjunto de N-Triples, para serializar múltiples grafos RDF.
  - JSON-LD, un formato de serialización basado en JSON. Trabajaréis con él durante las prácticas de laboratorio.
  - N3 (Notation3), un formato de serialización no estandarizado.
  - RDF/XML, sintaxis en XML que fue el primer estándar de serialización para RDF (en desuso).
- Algunas recomendaciones sobre qué formato usar en función de las circunstancias: aquí.

RDF

- Veamos los ejemplos anteriores serializados en Turtle.
  - (France, capital, Paris)

```
@prefix wd: <http://www.wikidata.org/entity/> .
@prefix p: <http://www.wikidata.org/prop/> .
wd:Q142 p:P36 wd:Q90 .
```

  - (EEUU, presidente, Joe Biden)

```
@prefix wd: <http://www.wikidata.org/entity/> .
@prefix p: <http://www.wikidata.org/prop/> .
wd:Q30 p:P6 wd:Q6279 .
```

  - (Daniel Gayo Avello, nombre, "Daniel Gayo Avello")

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix orcid: <https://orcid.org/> .
orcid:0000-0002-4705-6891 foaf:name "Daniel Gayo Avello" .
```

RDF

- Reglas fundamentales para escribir tripletas RDF:
  - El sujeto siempre es un recurso, nunca un literal.
  - Las propiedades son relaciones binarias y sus tipos siempre están identificados por IRIs.
  - Los valores pueden ser recursos o literales.
- Reglas básicas para escribir Turtle:
  - La directiva **@prefix** permite hacer muy compactos los archivos Turtle al ahorrarnos tener que escribir IRIs completas.
  - Las URLs e IRIs van siempre encerradas entre las cuñas **<** y **>**
  - Las sentencias (uso de directivas o tripletas) terminan con un punto **.**
  - Los literales de cadena van encerrados entre comillas dobles **"**
  - Pueden asociarse múltiples predicados a un mismo sujeto separándolos con puntos y comas **;**
  - Pueden asignarse varios valores a una misma propiedad separándolos con comas **,**
  - Por defecto los literales son cadenas y se les puede asignar un idioma mediante un sufijo como **@en**, **@es** o **@fr**. Por ejemplo, **"United States"@en**, **"Estados Unidos"@es** o **"États-Unis"@fr**. Los literales con idioma y sin idioma son disjuntos, p.ej. **"United States"** y **"United States"@en** no son la misma cadena.
  - A los literales se les puede asignar un tipo de datos XML Schema. Por ejemplo, **"13"^^xsd:int**, **"3.141592"^^xsd:float** serían literales tipados (suponiendo que se haya definido correctamente el prefijo **xsd**).
  - No se pueden combinar tipos XSD y etiquetas de idioma.
  - Se pueden dar tipos a los recursos, p.ej.

```
@prefix schema: <https://schema.org/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix orcid: <https://orcid.org/> .
orcid:0000-0002-4705-6891 rdf:type schema:Person .
```

RDF

- Para trabajar con RDF y Turtle conviene contar con validadores y conversores:
  - IDLab Turtle Validator
  - RDF Validator and Converter
  - RDFShape: Data info
- También os pueden interesar las siguientes herramientas: sketch.zazuko.com y prefix.zazuko.com.
- Para saber más: *"Validating RDF Data"*.

Datos enlazados (*Linked Data*)

- Tim Berners-Lee propuso el concepto de datos enlazados en 2006.
- Técnicamente los datos enlazados se refieren a la publicación de datos en la Web de tal manera que sean legibles por una máquina, sean enlazables y enlacen a otros datos.
- Conceptualmente, los datos enlazados se refieren a un conjunto de buenas prácticas para la publicación y enlace de datos estructurados en la Web.
- Así, la publicación de datos enlazados se rige por los siguientes principios:
    1. Utilizar RDF como formato de datos.
    2. Utilizar URIs para nombrar las "cosas".
    3. Proporcionar información útil siempre que alguien se conecte a una URI. Puede ser RDF, HTML u otro formato según la negociación del contenido con el cliente.
    4. Incluir enlaces a otras URIs para poder descubrir "cosas" relacionadas.
- Michael Smethurst (de la BBC) sugirió un quinto principio muy razonable: no reinventar los identificadores que otra gente está usando en la web; dicho principio está muy relacionado con este patrón. Aunque esto no siempre se puede aplicar debería tratar de evitarse este tipo de situaciones: `72c536dc-7137-4477-a521-567eeb840fa8`, `74ASZWbe4lXaubB36ztrGX`, `Q392`, `Bob_Dylan`.
- En base a esos principios los datos enlazados son navegables, para que sean consultables es preciso utilizar SPARQL (que veremos más adelante).

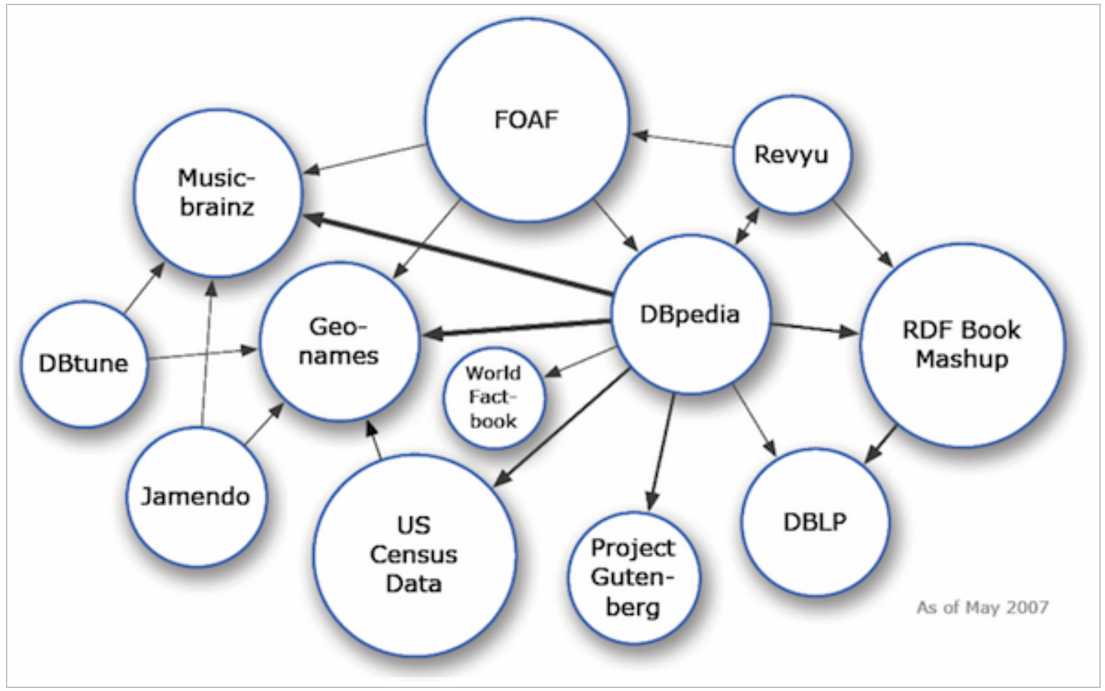
Datos enlazados abiertos (*Linked Open Data*)

Datos enlazados abiertos (*Linked Open Data*)

< Prev

La nube LOD en 2007

Next >



>> Play

SPARQL

- Por segunda en la historia de la Web vez nos encontramos con un grafo que se vuelve inmanejable al aumentar de tamaño.
- El hecho de que los datos esten enlazados entre sí no significa que resulte práctico navegarlos: se necesita un sistema de búsqueda.
- Entra en escena SPARQL...
  - Siendo honestos, ya en 2001, se planteó desde el W3C la necesidad de estandarizar lenguajes de consultas para grafos RDF.
  - En 2004 existían más de 20 implementaciones de lenguajes de consulta para grafos RDF.
  - En 2004 se publicó el primer borrador de SPARQL, en 2008 se convirtió en recomendación y su última versión es la 1.1 de 2013.
  - SPARQL es un acrónimo recursivo: *SPARQL Protocol and RDF Query Language*.

SPARQL

Sintaxis

- Parecida a la de Turtle.
- URIs entre cuñas **<** y **>**
- Se usan prefijos del mismo modo:

```
prefix dc: <http://purl.org/dc/terms/>
dc:creator
```

- Los literales van entre comillas dobles **"** y pueden ir tipados: **"Asturias"**, **"123"^^xsd:integer**
- Los comentarios van precedidos de almohadilla **#**
- Las variables comienzan por interrogación **?**, p.ej. **?nombre**

SPARQL

DBpedia y Wikidata

- Para poder ofrecer ejemplos de SPARQL necesitamos al menos un repositorio RDF sobre el cual consultar.
- Usaremos tanto DBpedia como Wikidata. ¿Qué son?
  - **DBpedia** es un proyecto para la **generación** de LOD **a partir de** información disponible en la Wikipedia.
  - **Wikidata** es otro proyecto LOD para la **creación** de un grafo de conocimiento abierto, lo dirige la Fundación Wikimedia pero **no importa datos de Wikipedia**, aunque Wikipedia si utiliza datos de Wikidata. Al igual que la Wikipedia los datos de Wikidata son editables por cualquiera.
- En ambos casos se dispone de un punto de acceso SPARQL al que enviar consultas en ese lenguaje (DBpedia, Wikidata).

SPARQL

Ejemplos

- Una consulta SPARQL tiene el siguiente aspecto:

```
SELECT ?a ?b ?c
WHERE
{
  x y ?a .
  m n ?b .
  ?b f ?c .
}
```

- La cláusula **SELECT** lista las variables que deseamos obtener y la cláusula **WHERE** contiene las restricciones que deben cumplirse en forma de tripletas.
- El resultado serán todas aquellas tripletas del repositorio que verifiquen las restricciones.

SPARQL

Ejemplos

- Supongamos que queremos encontrar todos los hijos de Julio Iglesias. El pseudocódigo para dicha consulta sería el siguiente:

```
SELECT ?hijo
WHERE
{
  #  hijo "tiene como padre a" Julio Iglesias
  ?hijo padre Julio Iglesias .
}
```

- Para poder formular la consulta necesitamos saber cómo se representan en Wikidata la propiedad "tiene como padre a" y la entidad Julio Iglesias.
- Julio Iglesias es el concepto Q122003 y padre sería la propiedad Property:P22.
- La consulta quedaría tal que así:

```
SELECT ?hijo
WHERE
{
# ?hijo padre Julio Iglesias
  ?hijo wdt:P22 wd:Q122003 .
}
```

SPARQL

Ejemplos

- En la consulta anterior se usaron los prefijos **wdt:** y **wd:**, el primero corresponde a propiedades y el segundo a entidades.
- Por otro lado, podemos mejorar la consulta para que nos proporcione la etiqueta correspondiente a las tripletas obtenidas (en este caso hijos):

```
SELECT ?hijo ?hijoLabel
WHERE
{
# ?hijo padre Julio Iglesias
  ?hijo wdt:P22 wd:Q122003 .
  SERVICE wikibase:label { bd:serviceParam wikibase:language "[AUTO_LANGUAGE]". }
}
```

SPARQL

Ejemplos

- Una consulta similar en DBpedia sería así:

```
SELECT ?person WHERE {
  ?person dbo:parent :Julio_Iglesias .
}
```

- El prefijo **dbo:** hace referencia a la ontología de DBpedia y el identificador para la entidad Julio Iglesias es más razonable que en Wikidata: **Julio_Iglesias**.
- **¡Atención!**
  - Algo notorio en los ejemplos anteriores es que los resultados difieren entre ambos repositorios y, además, en ningún caso tienen la información completa (Julio Iglesias tiene 9 hijos).
  - Una cuestión no trivial a la hora de utilizar y derivar conocimiento de la Web (usando o no tecnologías de Web Semántica) es la **procedencia de los datos** (aka ***provenance***) (véase [1], [2] y [3]).
  - A día de hoy se sigue una política 🤞 y se confía en que salga lo mejor 😌.

SPARQL

Ejemplos

- Cambiemos de figura paterna a Donald Trump

```
SELECT ?hijo ?hijoLabel
WHERE
{
# ?hijo padre Donald Trump
  ?hijo wdt:P22 wd:Q22686 .
  SERVICE wikibase:label { bd:serviceParam wikibase:language "[AUTO_LANGUAGE]". }
}
```

- Nos interesa este ejemplo porque Trump ha tenido hijos con varias esposas así que podemos "complicar" la consulta tal que así para buscar sus hijos con Ivana Trump:

```
SELECT ?hijo ?hijoLabel
WHERE
{
# ?hijo padre Donald Trump
  ?hijo wdt:P22 wd:Q22686 .

# ?hijo madre Ivana Trump
  ?hijo wdt:P25 wd:Q242351 .
  SERVICE wikibase:label { bd:serviceParam wikibase:language "[AUTO_LANGUAGE]". }
}
```

SPARQL

Ejemplos

- Supongamos ahora que queremos encontrar las hijas que Donald Trump haya tenido con su esposa Ivana:

```
SELECT ?hijo ?hijoLabel
WHERE
{
# ?hijo padre Donald Trump

  ?hijo wdt:P22 wd:Q22686 ;
  # madre Ivana Trump
        wdt:P25 wd:Q242351 ;
  # genero femenino
        wdt:P21 wd:Q6581072 .
  SERVICE wikibase:label { bd:serviceParam wikibase:language "[AUTO_LANGUAGE]". }
}
```

- Obsérvese cómo se ha usado el punto y coma **;** para tener varios predicados referidos a la misma variable.

SPARQL

Ejemplos

- La consulta equivalente para DBpedia sería la siguiente:

```
SELECT ?person WHERE {
  ?person dbp:father :Donald_Trump ;
          dbp:mother :Ivana_Trump ;
          foaf:gender "female"@en .
}
```

- Cuestiones interesantes:
  - No hemos usado **dbo:parent** como con Julio Iglesias sino **dbp:father** y **dbp:mother**, ha cambiado tanto el prefijo como la propiedad. Dicho de otro modo, DBpedia no tiene gran consistencia interna.
  - Para la propiedad género se usa la ontología de FOAF y se especifica el valor como una cadena de caracteres en lengua inglesa **"female"@en**.

SPARQL

Ejemplos

- ¿Cómo supe cómo había que especificar el género en DBpedia?
- No sabía, tuve que hacer "trampas":

```
SELECT ?person ?genero WHERE {
  ?person dbp:father :Donald_Trump ;
          dbp:mother :Ivana_Trump ;
          foaf:gender ?genero .
}
```

- Obsérvese cómo podemos hacer consultas con múltiples variables, en este caso **?person** y **?genero**.

SPARQL

Ejemplos

- Hasta el momento hemos buscado objetos que tienen propiedades. ¿Cómo podemos buscar instancias de una clase determinada?
- Antes veamos cómo podemos saber a qué clases pertenece un objeto dado.
- Por ejemplo, ¿qué son Julio Iglesias y Donald Trump? (aviso: consulta algo "chapucera")

```
SELECT ?claseJulio ?claseJulioLabel ?claseDonald ?claseDonaldLabel
WHERE
{
# Julio Iglesias instancia de ?claseJulio
  wd:Q122003 wdt:P31 ?claseJulio .
# Donald Trump instancia de ?claseDonald
  wd:Q22686 wdt:P31 ?claseDonald .
  SERVICE wikibase:label { bd:serviceParam wikibase:language "[AUTO_LANGUAGE]". }
}
```

- Julio Iglesias **es** un ser humano y Donald Trump, también. ~~Donald Trump **es dos cosas**: ser humano y milmillonario.~~

SPARQL

Ejemplos

- Una consulta semejante en DBpedia sería la siguiente (y aquí la chapuza ya puede ser peligrosa):

```
SELECT ?claseJulio ?claseDonald WHERE {
  :Julio_Iglesias rdf:type ?claseJulio .
  :Donald_Trump rdf:type ?claseDonald .
}
```

- En DBpedia el número de clases a las que puede pertenecer una entidad llega al absurdo...
  - Por un lado tenemos las que podríamos considerar "normales" como **foaf:Person** o **http://schema.org/Person**...
  - ...pero luego hay clases como **http://dbpedia.org/class/yago/WikicatRealMadridCastillaFootballers** o **http://dbpedia.org/class/yago/WikicatConspiracyTheorists**
- Los problemas de generar tripletas automáticamente a partir de la Wikipedia comienzan a manifestarse...

SPARQL

Ejemplos

- Centrémonos pues en Wikidata a partir de ahora y veamos cómo podemos encontrar todos los seres humanos con las ocupaciones de Julio Iglesias o Donald Trump.
- Así pues, ¿a qué se dedican cada uno de ellos?

```
SELECT ?ocupacion ?ocupacionLabel
WHERE
{
# Julio Iglesias tiene como ocupación ?ocupacion
  wd:Q122003 wdt:P106 ?ocupacion .
  SERVICE wikibase:label { bd:serviceParam wikibase:language "[AUTO_LANGUAGE]". }
}

SELECT ?ocupacion ?ocupacionLabel
WHERE
{
# Donald Trump tiene como ocupación ?ocupacion
  wd:Q22686 wdt:P106 ?ocupacion .
  SERVICE wikibase:label { bd:serviceParam wikibase:language "[AUTO_LANGUAGE]". }
}
```

- Para el caso de Julio Iglesias elegiremos **wd:Q177220** (cantante) y para Donald Trump **wd:Q911554** (magnate).

SPARQL

Ejemplos

- Busquemos todos los cantantes:

```
SELECT ?person ?personLabel
WHERE
{
# ?person tiene como ocupacion cantante
  ?person wdt:P106 wd:Q177220 .
  SERVICE wikibase:label { bd:serviceParam wikibase:language "[AUTO_LANGUAGE]". }
}
```

- Busquemos a todos los magnates:

```
SELECT ?person ?personLabel
WHERE
{
# ?person tiene como ocupacion magnate
  ?person wdt:P106 wd:Q911554 .
  SERVICE wikibase:label { bd:serviceParam wikibase:language "[AUTO_LANGUAGE]". }
}
```

- Dos cuestiones importantes a tener en cuenta con estos ejemplos:
  - Potencialmente, los resultados pueden ser muy extensos así que pueden limitarse con la clásula **`LIMIT`**, p.ej. los 10 primeros cantantes y los 10 primeros magnates.
  - Los resultados no están ordenados por relevancia, una consulta SPARQL es una consulta boolean con esteroides así que no hay resultados "mejores" o "peores". Esa es la teoría, por supuesto, porque los usuarios sí esperan algún tipo de orden. ¡Artículo muy interesante!

SPARQL

Para saber más...

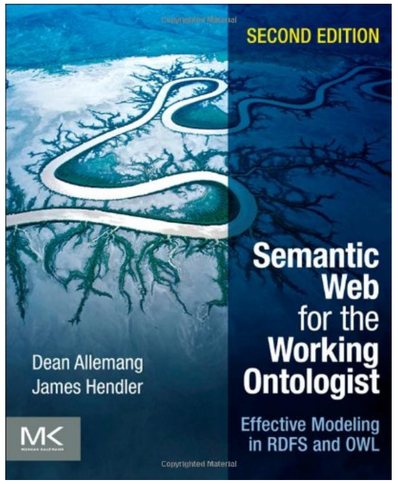
- La referencia de SPARQL 1.1 sería el punto de partida obligado.
- Labra tiene unas transparencias excelentes sobre SPARL que cubren, además de las consultas **SELECT**, las consultas **ASK**, **DESCRIBE** y **CONSTRUCT**.
- Wikidata tiene un tutorial sobre SPARQL que va bastante más allá de estos ejemplos.
- El libro *"Learning SPARQL"* de Bod DuCharme también es recomendable.
- Va a haber una sesión de prácticas dedicada a SPARQL.

Implementaciones

- No existe el MySQL de RDF y SPARQL. Sin embargo...
- Existen múltiples implementaciones de almacenes de tripletas.
- Existen múltiples implementaciones de motores de consulta SPARQL.
- El proyecto HDT puede ser muy interesante.

Otros estándares semánticos

- Al ver la pila de tecnologías semánticas aparecieron RDF Schema y OWL.
- No vamos a utilizarlos en la asignatura pero puede ser interesante echarles un vistazo para entender mejor iniciativas como Schema.org.
- Además de las referencias del W3C puede resultar de interés el libro *"Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL"*.
- Schema.org ofrece las definiciones de sus vocabularios en formatos RDF pero también en RDFS y OWL (aunque estos dos últimos pueden desaparecer).



Grafos de conocimiento

- En la asignatura "Repositorios de Información" visteis, entre otras cosas, bases de datos en grafo.
- En ese tipo de bases de datos la información se representa mediante nodos relacionados entre sí mediante aristas y tanto los nodos como las aristas pueden tener propiedades con valores asignados.
- Un grafo de conocimiento es extremadamente similar y la razón fundamental por la que se distinguen ambos términos es para separar campos de trabajo similares en cuanto a filosofía pero diferentes en cuanto a tecnologías y alcance:
  - El concepto de bases de datos en grafo está fuertemente asociado al mundo NoSQL e implementaciones concretas como Neo4j, Titan, GraphDB u OrientDB.
  - Los grafos de conocimiento tienden a asociarse con las tecnologías de la Web Semántica, los datos enlazados o a grafos concretos como el de Google, Wikidata, DBpedia o YAGO.
- Ni que decir tiene ambos campos se solapan en ocasiones (p.ej., GraphDB utiliza RDF y SPARQL) y en otras un grafo de conocimiento no tiene por qué usar tecnologías de Web Semántica (p.ej., el grafo de Google).
- En estos momentos los grafos de conocimiento están atrayendo muchísima atención no solo entre la comunidad investigadora sino en la industria y van a jugar un papel muy importante en los próximos años.
- Para saber más os recomiendo este artículo que ofrece una perspectiva histórica breve pero muy interesante y este informe que, entre otras cosas, describe los retos que deberían afrontarse a corto plazo (p.ej., la integración entre distintos grafos de conocimiento o el modo de construir un grafo público de **todo el conocimiento**).

Conclusiones

- Posiblemente la visión original de la Web Semántica fue demasiado ambiciosa y, tal vez, la pila de tecnologías sea excesivamente compleja.
- No obstante, la misión de conseguir explotar el conocimiento subyacente en la Web como una gigantesca base de datos/grafo de conocimiento es muy potente y valiosa.
- De hecho, el concepto del grafo de conocimiento es explotado por los buscadores Web con asiduidad.
- Por otro lado, versiones mas "ligeras" de algunas tecnologías semánticas tienen cabida en aplicaciones prácticas en la Web (p.ej. microdatos o JSON-LD).
- Necesidades de información elaboradas podrían simplificarse enormemente si fuera posible añadir información semántica.
- El libro *"A Developer's Guide to the Semantic Web"* puede ofrecer una visión pragmática para desarrolladores. También os pueden interesar los libros *"Linked Data Patterns"* y *"Social Semantics: The Search for Meaning on the Web"*.
- Un aspecto crucial como la procedencia de los datos aún no está bien resuelto y puede resultar muy problemático pues, recordemos, la Web es un entorno adversarial.