Linking Enterprise Data

François-Paul Servant
WWW 2008 Workshop: Linked Data on the Web - April 22, 2008 - Beijing
“Business model for the Semantic Web” [Berners-Lee]

- Integrating data sources of a corporation is expensive.
  - \( \sim N^2 \) point to point connections between \( N \) systems

- Using SW technologies can cut down these costs.
  - \( N \) RDF views of the \( N \) systems
However, adoption of SW technologies in companies remains slow.

- SW technologies are often simply overlooked,
  - Not advertised by solution providers
- or considered as promising
  - but not ready to be used right now
- Even a negative prejudice
  - yet another technological hype, whose promises have been heard many times before.

- “We already have XML and Web Services. Why do we need SW?”
Linking Open Data / Linking Enterprise Data

- The “Linking Open Data” community initiative:
  - a major contribution to the concretization of the Web of Data
    - Proof of feasibility
    - Best practices

- Why not try the same strategy in an enterprise?
  - to publish enterprise data as linked data
  - to produce guidelines, sample code,
  - to give examples of applications using that data.
Envisioning one company’s Information Systems as a space of linked data

- Linked Data principles provide effective solutions for 2 questions regarded by Renault as priorities for its IS architecture:
  - data repositories
  - services

- Our work was intended to demonstrate that
Summary of what we did

- Publishing of a data repository as Linked Data
  - REST service returning RDF to requests made by programs
- Implementation of a (very) simple RDF browser
  - Provides the HTML web application
- Access to the data from an outer application
Previous experiences with Linked Data: www.semanlink.net
Previous experiences with Linked Data: Prototype repository of repair and diagnostics operations
The field of the use case

- **Technical after-sales documentation**
  - A typical case for using SW technologies:
    - many objects need to be shared between many different systems,
      - ranging from engineering to repair shops,
      - in a rather complex process.
The repository

- **A dictionary of the terms that documentation writers may use**
  - Used to enforce an homogenous naming scheme throughout the whole documentation
  - Used to index repair and diagnostics methods, and to define links between documents

- **Why this repository ?**
  - Not available as a service
  - XML dump available
  - Well managed
  - Interesting uses

- **Content:**
  - for each term:
    - An identifying code
    - Labels (in 20 languages)
    - List of the corresponding “Generic Parts”
      - = link between repair documentation and spare parts catalogs
      - “Generic Parts”: codification of the components of a car maintained by the Parts Department.
  - A SKOS-like hierarchy of the terms
    - Routine maintenance : Filters : Oil filter
Development environment

- **Java**
  - Servlet
  - Jena

- **Javascript**
Implementing Linked Data principles

- Minting URIs for the items of the repository
  - namespace / item_class / item_code
    - "Slash URI"
      - can be dereferenced one at a time
      - namespace / item_class / item_code ? param=val &...

- "Non-Information Resources"
  - "HTTP-range 14" (303 redirect, with content negotiation)
    - namespace / item_class / item_code
    - namespace / item_class / item_code.rdf
    - namespace / item_class / item_code.html
Answering requests for the RDF about an NIR

- return all statements involving the NIR, either as subject or as object
  - NIR_URI, ?, ?
  - ?, ?, NIR_URI
  - and a few more, such as
    - the links between nir, nir.rdf and nir.html
    - the labels of the returned resources that belong to the dataset
Answering requests for the HTML about an NIR

- **Several options are possible for HTML generation**
  - server side (JSP)
  - client side (Javascript)

- **Client side generation of HTML**
  - Tabulator’s Javascript RDF parser
    - adapted to work with Internet Explorer
  - GET nirURI.html returns a quasi empty HTML page, containing a script that downloads nirURI.rdf and displays it

- **Advantages**
  - clean separation of “view” and “model”
  - decreases load on the server
  - possibility to change the display without sending a new request to the server
  - incremental load of RDF data
Generating HTML on the client with Javascript

Jena Model

Servlet

GET nirURI

303 redirect nirURI.html

GET nirURI.html

<html>
  <body onload="display('nirURI.rdf')">
  </body>
</html>

GET nirURI.rdf

RDF

nirURI,?,?,?,?nirURI

RDF

Parsing in JS, display

Time
Broader term
(Parent in hierarchy)

“Generic Parts”
Broader term(s): engine and peripherals

Narrower term(s):
- combustion chamber
- complete engine
- cooling
- cylinder head
- engine block
- engine oil
- lubrication
- rotating parts
- supports
- TDC pressure
- valve timing
- wiring

Properties

  - combustion chamber
  - complete engine
  - cooling
  - cylinder head
  - engine block
  - engine oil
  - lubrication
  - rotating parts
  - supports
  - TDC pressure
  - valve timing
  - wiring
- http://www.w3.org/2000/01/rdf-schema#label
  - cs : motor
Turning the solution into a simple, yet generic RDF browser

- With what we described until now, we can only dereference NIR from our own dataset
  - because it is Javascript that gets the RDF
  - Standard Javascript security restriction
  - Changing browser’s default settings (as Tabulator does) was not an option
- We implemented the usual trick of the “HTTP proxy”
  - Requests to dereference externals URIs are sent to the servlet,
  - the servlet forwards them to the actual server,
  - and then returns the result.
François-Paul Servant
April 22, 2008 - Beijing
WWW2008 Workshop: Linked Data on the Web

Semanlink page about a tag

Same resource displayed by this project’s RDF browser
Access to repository data from outer application

- The repository, published as Linked Data, is a service
  - easy to get connected to (HTTP GET)
  - no dedicated code is needed to understand its RDF data
    - a RDF parser is enough
- Let's use it!
Car configurator
Here a partially defined car is shown

<table>
<thead>
<tr>
<th>Véhicule</th>
<th>Pièces</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objet : 007</td>
<td><strong>Pièce générique</strong></td>
</tr>
<tr>
<td><strong>X90</strong></td>
<td></td>
</tr>
<tr>
<td>Objet : 008</td>
<td></td>
</tr>
<tr>
<td><strong>L90 B90 F90 K90 U90</strong></td>
<td></td>
</tr>
<tr>
<td>Objet : 011</td>
<td></td>
</tr>
<tr>
<td><strong>CHANOR CHAAUG CHADIM</strong></td>
<td></td>
</tr>
<tr>
<td>Objet : 013</td>
<td></td>
</tr>
<tr>
<td><strong>E0 E1 E2 E3 E3S E2N</strong></td>
<td></td>
</tr>
<tr>
<td>Objet : 018</td>
<td></td>
</tr>
<tr>
<td><strong>M0 M6 MJ MK MW MY AA AG M8 M9 MA MB MC MD ME MF MG MH ML MM MP MR MS MT MU MV ND1B NM1C NM1D NM2D</strong></td>
<td></td>
</tr>
<tr>
<td>Objet : 019</td>
<td></td>
</tr>
<tr>
<td><strong>DIESEL ESS FLEXFL</strong></td>
<td></td>
</tr>
<tr>
<td>Objet : 020</td>
<td></td>
</tr>
<tr>
<td><strong>5PL 2PL 7PL</strong></td>
<td></td>
</tr>
<tr>
<td>Objet : 026</td>
<td></td>
</tr>
<tr>
<td><strong>ALLE ARGE AUTR BALK BELG BENG BOSN CHIL CROA CSIA CSIB CSIC CSID CSIF CSIG CSII CSIN CSIU CZSK DOME ESPA FRAN GREC HOLL HONG INDE ISRA ITAL MAGH MARO PARA PERO POLO PORT ROUM SLVN SLVQ SUIS TCHE TOME TURQ AFS4 AFSU BRES COLO GRBR IRAN IRIK IRLA IRSN MEX4 MEXI NORD PHTN RUSS VENE</strong></td>
<td></td>
</tr>
<tr>
<td>Objet : 027</td>
<td></td>
</tr>
<tr>
<td><strong>DD DG</strong></td>
<td></td>
</tr>
<tr>
<td>Objet : 029</td>
<td></td>
</tr>
</tbody>
</table>

One property of the car
Possible values

Part’s reference for a generic part depend on car’s properties (SAT problem)

Other property of the car (fuel type)
Diesel is selected

One reference

“Generic Part” code

Boolean formula of the properties of the car that tells whether the reference is used on that car
Choose in the hierarchy defined in the repository of repair terms

or enter term's code

Véhicule

Objet: 007

X90

Objet: 008

L90 B90 F90 K90 U90

Objet: 011

CHANOR CHA AUG CHADU

Objet: 013

E0 E1 E2 E3 E4 E5 E2N

Objet: 018

M0 M6 MJ MK MW MY AA A B

Objet: 019

DIESEL ESS FLEXFL

Objet: 020

5PL 2PL 7PL

Objet: 026

ALLE ARGE AUTR BALK BI

Objet: 027

DD DG

Objet: 029

WWW2008 Workshop: Linked Data on the Web

François-Paul Servant

April 22, 2008 - Beijing
Use of the repository for technical documentation

Repair methods about gearbox

- Boîte de vitesses mécanique : Dépose - Repose
  - X90/D4D/JH3

- Boîte de vitesses mécanique : Dépose - Repose
  - X90/K4M/JR5/JH3

- Boîte de vitesses mécanique : Dépose - Repose
  - X90/K9K_792/JR5/JH3

- Boîte de vitesses mécanique : Dépose - Repose
  - X90/K9K_790/JR5/JH3

- Boîte de vitesses mécanique : Dépose - Repose
  - X90/K9K_792/JR5/JH3

- Boîte de vitesses mécanique : Dépose - Repose
  - X90/K9K_790/JR5/JH3

Boolean formula of the characteristics of a car. Document applies for a given car iff this formula is true.
A (current) limitation of linked data principles

- **Attached to each document, there is an “Applicability”**
  - a Boolean formula of the characteristics of a car that defines the set of cars this document is relevant to.

- **When searching information for a repair, you’re only concerned with the documents relevant to the car being repaired.**

- **It doesn’t make sense to return the documents about all the cars:**
  - the client wouldn’t be able to understand the Boolean formulae
  - Only the server can deal with them.

- **A form is needed**
  - Let’s suppose that those formulae depend of 3 variables: v:model, v:engine and v:gearbox
  - In a web 1.0 application, a form would be displayed to the user, asking to enter values for those variables.

- **How can we handle that in the context of the web of data?**
  - With Linked Data, there is no standardized equivalent : no way for the server to request some input from the client
RDF Form

- `<rdf:Description rdf:about="rdc:gearbox">
  <form:hasForm><form:Form>
    <form:param rdf:resource="v:model"/>
    <form:param rdf:resource="v:engine"/>
    <form:param rdf:resource="v:gearbox"/>
  </form:Form></form:hasForm>
</rdf:Description>`
Forms in the web of data

- "The Semantic Web [...] is about making links, so that a person or machine can explore the web of data." [Berners-Lee]
- Beside hypertext links, forms are an important feature of the web.
- They are needed also in the web of data.
Conclusion

- Several other such points need to be discussed among the LOD community
  - e.g., the question of the amount of data to return when fetching the URI of a NIR

- Anyway, Linked Data principles proved to be effective in a corporate context:
  - we published a repository as Linked Data
    - and that turned it into a service
      - easy to get connected to and easy to use
      - that fully respects the web architecture
        » with immediate and free benefits (caching)
      - (this has to be compared with WS-* services)
    - the repository was simple, but the solution is largely reusable and expandable

- We will now
  - improve the prototype of technical documentation based on RDF metadata
  - and take all opportunities to continue working on Linked Data!
- Thank you!
Dealing with applicability

- **Use the VIN (“Vehicle Identification Number” - kind of URI for cars)**
  - Constructors manage a database of cars, indexed by VIN, and that stores their technical characteristics
  - We could define the URI of the “gearbox of a given car”, adding the VIN as parameter:
    - http://.../element/1018?vin=VF123...
    - http://.../element/1018 gives the list of all documents about gearbox
    - The VIN database gives the characteristics of car VF123..
    - The server evaluates the applicability of each document
    - returns the filtered list
  - But
    - this URI doesn’t fit well within Linked Data
    - we need a way for the server to answer to a GET http://.../element/1018
      “information for a given car available by passing the VIN as parameter”.
  - Also, cache is of little use for URIs with a parameter such as the VIN
    - the server can know that the applicabilities of the documents about gearbox involve only (for instance), the model of the car, its engine type and its gearbox type
    - So, it would be nice if the server were able to ask the client who dereferenced the URI of the gearbox:
      “please enter model, engine type and gearbox to get the document you’re concerned with.”