Linking Data from RESTful services

Rosa Alarcon
Computer Science Department
Pontificia Universidad Catolica de Chile

Erik Wilde
School of Information
UC Berkeley

LDOW 2010
Abstract

- Semantic Web goal: extend current human-readable Web resources with semantic information encoded in machine-processable form
- Web of Data: Linked Data principles, several data sources compliant with the Semantic Web technologies, such as, RDF triple stores, and SPARQL endpoints
- REST: a set of the architectural principles that underlie the human-readable Web
- To provide a mechanism for describing REST (i.e. human-readable Web) resources and transform them into semantic resources
- Lower entry barrier

LDOW - 2010
REST (Representational State Transfer)

- **Resources have unique, opaque identifiers**
  - Avoid coupling between clients and servers

- **Uniform interface: methods with known semantics that change the state of resources**
  - HTTP: GET, PUT, DELETE, POST, OPTIONS

- **Resources (conceptual) have multiple representations**
  - JSON, XML, XHTML, etc.

- **Hyperlinks**
  - Related resources + State change
ReLL: Resource Linking Language
ReLL Description Schema
ReLL Snippet: Describing the ISchool

```xml
<service ... targetNamespace = "http://rell.org/school/" ... >
  <resource xml:id = "person">  
    <name> ... 
    <desc> ... 
    <uri match="http://.*?/people/(faculty|students|staff|visitors)/[a-zA-Z]+" type="regex"/>
  <representation xml:id="person-html" type="iana:text/html">  
    <name> ... 
    <link xml:id="person-website" type="website">
      <selector select="/div[@class = 'field-field-person-website']/a/@href" type="xpath"/>
    </link>
  </representation>
  </resource>
  <link xml:id="person-course" type="personcourse" target="course">
    <selector select="/span[@class = 'views-field-title']/a/@href" type="xpath"/>
    <protocol type="http">
      <request method="get"/>
      <response media="iana:html"/>
    </protocol>
  </link>
</service>
```
From ReLL to RDF
Implementation: RESTler
Crawled REST Services
Crawled REST Resources
Generated RDF
Getting RDF from Resources

```xml
<http://www.ischool.berkeley.edu/people/faculty/erikwilde> a school:person;
    vCard:FN "Erik Wilde";
    vCard:ADR _:node14m5kienpx1603;
    vCard:TITLE "Adjunct Professor";
    vCard:ORG _:node14m5kienpx1604;
    vCard:EMAIL _:node14m5kienpx1606;
    vCard:TEL _:node14m5kienpx1607;
    vCard:URL <http://dret.net/netdret/>;
    vCard:PHOTO <http://www.ischool.berkeley.edu/files/imagecache/profile-pic/DSC_0176.JPG>;
    school:person-website <http://dret.net/netdret/>;
    school:person-course <http://www.ischool.berkeley.edu/programs/courses/242>,
       <http://www.ischool.berkeley.edu/programs/courses/152>,
       <http://www.ischool.berkeley.edu/programs/courses/190-waim>,
       <http://www.ischool.berkeley.edu/programs/courses/290-wa>.
```
ISchool Camera SPARQL

(a)

PRE: PREFIX dcterms:<http://purl.org/dc/terms/>
PR: PREFIX flickr:<http://rell.org/flickr/>
S: SELECT DISTINCT ?person ?camera 
T: ?picture flickr:photo-taken ?camera}
Conclusions

- Do RESTful services even should be described?
  - Descriptions introduce coupling between service provider and consumer.
  - Shared set of assumptions & preconditions that facilitates documentation, understanding, & change identification (e.g. new ids, access schemes or representation format), so that clients and serves can react appropriately (e.g. alerting the client manager, attempting a fallback, or abort).
  - Mostly useful for automatic agents that translate the contract into RDF triples.

- Hence, ReLL can achieve loose coupling and still allow clients to behave when the unexpected occurs.

LDOW - 2010
Conclusions

- **Limitations**
  - Static description of RESTful services (new resources, and changes ignored)
  - Design a ReLL document for each REST service (e.g. Virtuoso’s Sponger)
  - Design a specific XSLT for each resource type to harvest information

- **On the bright side**
  - Web technologies (e.g. XPath, XSLT and XML) familiar to Web developers

- **Future work**
  - Dynamic and automatic generation of ReLL descriptions; may require information retrieval, text mining and probably machine learning techniques
  - Complex REST models (e.g. methods such as PUT, DELETE and POST)
  - Include Linked Data vocabularies