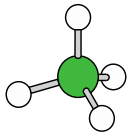
Center for Open Middleware

Building interoperable read-write Linked Data applications with the W3C Linked Data Platform and the LDP4j framework

ESWC 2015 - May 31st , 2015 - Portoroz (Slovenia)



Miguel Esteban Gutiérrez,
Nandana Mihindukulasooriya,
Raúl García Castro,
Center for Open Middleware / Ontology Engineering Group
Universidad Politécnica de Madrid,
Spain.

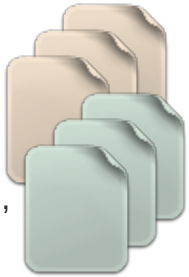
Building interoperable read-write Linked Data applications with the W3C Linked Data Platform and the LDP4j framework

DESIGN CONSIDERATIONS FOR INTEROPERABLE READ-WRITE LINKED DATA APPLICATIONS

What we have to consider

R+W LD
Business Logic

- LDP (and paging) 1.0
- RDF 1.1
- LD+JSON
- RFC 3987, 5785, 5789, 5988, 6585, 7240



LDP

RDF Sources
Containers (Basic, direct, and indirect)
Paging
RDF transformation

- Fielding, 2000;
- Fielding & Taylor, 2002

REST best practices

Client-server
Stateless
Cacheable
Layered system
Uniform interface

- RFC 2616, 3986, 7230-7235

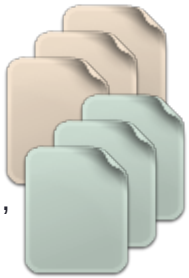
HTTP

Syntax
Routing
Operation semantics
Content negotiation
Conditional requests

What we can use up-to-now

R+W LD Business Logic

- LDP (and paging) 1.0
- RDF 1.1
- LD+JSON
- RFC 3987, 5785, 5789, 5988, 6585, 7240



LDP

RDF Pa… Third pa… utilities that s…pport LDP

Only for RDF-based data access…d

- Fielding, 2000;
- Fielding & Taylor, 2002

REST best practices

No support at all

- RFC 2616, 3986, 7230-7235

HTTP

Servlet 3.0 JAX-RS

Syntax and routing

What do we want to develop?

Linked Data Application "Maturity Model"

- **Linked Data Enabled Application**
  - Expose all or part of its data following the Linked Data principles
  - The data exposed is "sound and complete" from the application perspective
- **Linked Data Capable Application**
  - Consumes data published following the Linked Data principles
- **Linked Data Aware Application**
  - *Linked Data Enabled* and *Linked Data Capable* application
  - Capable of integrating its own data with other Linked Data

Interoperability!!!

Interoperability requirements

- We need to provide the means for ensuring that the data exchanged between interacting applications is **consistent** and **valid** whilst keeping the **integrity** of the data in each of these applications

- **Dimensions:**
  - *Vocabulary management*
  - *Consistency model*
  - *(RDF) Validation mechanism*

Vocabulary management

- *How do we get to know which vocabularies are used an application?*
  - Not just the namespaces, but the particular implementation used (*i.e.*, in case of using versioned ontologies)
- *How do we get to know how these vocabularies are to be used?*
  - Not only the terms (*i.e.*, concepts and properties) that can be used, but:
    - how does the application expect to consume data using the vocabulary (*i.e.*, POST and PUT operations)
    - how does the application provide data using the vocabulary (*i.e.*, GET operations)
- **LDP 1.0 does not consider neither provide mechanisms for dealing with vocabularies**

Consistency model

- *How do we ensure that the data managed by the application is consistent?*
  - Not just enforcing that no invalid data is published, but advertising those aspects that have an impact on the validity of the data beyond those expressed by the vocabulary and the constraints on top of it (*e.g.*, data provenance, property modifiability)
- *How do we ensure that a collection of ineracting applications keep their state consistent?*
  - Not just enforcing that their data is valid, but that aggregated data is consistent at a given point in time,
    - Proactively advertising resource state changes so that consumers can act upon data changes
    - Providing transaction support.

- **LDP 1.0 does not consider consistency requirements neither provides mechanisms for the notification of resource state changes**

RDF validation mechanism

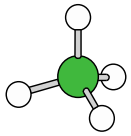
- Distributed scenario
- Based upon vocabulary and consistency constraints
- Constrained by several factors:
  - **Data source factors**
    - Behavioral aspects (static, periodically-updated, randomly-updated)
    - Structural aspects (*Materialization, inlining)*
  - **Procedure factors**
    - Data aspects
    - Temporal aspects (on-the-fly, upfront, long-lived, short-lived)
  - **Context factors**
    - Operational aspects (data provenance, application-managed vs user provided, write-once/read-many)

Conclusions

- Interoperability of read-write Linked Data applications requires
  - Addecuate usage of the vocabularies used by the applications
  - Provision of vocabulary management and consistency enforcement capabilities
- RDF validation in a Linked Data scenario has other concerns **beyond** traditional *structural* and *data range* validation issues
- Procedures for validating Linked Data need to be **customized** to accommodate the particularities of the scenario in terms of the
  - the **data sources** to be consumed,
  - the **processes** to be carried out, and
  - the **context** in which they are to be applied
- The factors that impact the design and development of the RDF validation process provides a roadmap for the design and implementation of interoperable read-write Linked Data applications!!!

Center for Open Middleware

Questions?



Miguel Esteban Gutiérrez,
Nandana Mihindukulasooriya,
Raúl García Castro,
Asunción Gómez Pérez
Center for Open Middleware / Ontology Engineering Group
Universidad Politécnica de Madrid,
Spain.