RDF Validation in a Linked Data world
A vision beyond structural and value range validation

Miguel Esteban-Gutiérrez, Raúl García-Castro, Nandana Mihindukulasooriya

Ontology Engineering Group, Center for Open Middleware
Universidad Politécnica de Madrid
{mesteban, rgarcia, nmihindu}@fi.upm.es

Abstract

Data validation is a vital step for ensuring the quality of data and the expressive languages for doing so and their related tools are essential for a data model to be adopted by the industry. Many data representation and storage technologies, like relational databases or XML, use expressive schema languages for defining the structure and the constraints on data and allow ensuring that the quality and the consistency of data is kept intact. In the context of semantic and Linked Data technologies, which are built upon the *Open World Assumption* and *Non-unique Name Assumption*, data validation becomes a challenge as the languages currently used to describe these constraints (i.e., RDF Schema and OWL) are more suited for inferring data than for data validation. There is a clear need for more expressive languages to define rules for validating RDF data.

However, having a wider view on the different use cases where RDF data is being used and considering the applications that consume RDF data as Linked Data, we can discover that there are requirements and concerns that go beyond the structural validation and data value range validation. In this paper, we identify the different requirements and factors that need to be taken into account and discussed in the context of data validation in applications that publish and consume Linked Data. These factors are grouped into three main categories: data source factors, procedure factors, and context factors. We believe that having this broader view will help to identify the concrete requirements for data validation especially in the context of Linked Data.

1. Introduction

In order to foster the adoption of Linked Data technologies as the means for facilitating application integration in enterprise-grade environments it is necessary to provide the means for ensuring that the data that will be exchanged between the applications in the enterprise portfolio is consistent and valid whilst keeping the integrity of the data in each of these applications.

Depending on the exposure to Linked Data technologies, the applications to be integrated can be classified as *linked data enabled*, *linked data capable*, and *linked data aware* applications. On the one hand, *linked data enabled* applications are those that expose their data following the Linked Data principles[1]. These applications may expose all or part of their data using a specific vocabulary[2], but the data they expose is sound and complete from the application perspective[3]. On the other hand, *linked data capable* applications are those that can consume data published following the Linked Data principles. Thus, these applications are capable of crawling the web looking for additional resources according to their business needs. Finally, *linked data aware* applications are those *linked data enabled* and *linked data capable* applications capable of integrating their own data with other Linked Data.

[1] http://www.w3.org/DesignIssues/LinkedData.html
[2] In this context, the vocabulary is meant to be the model used for representing the data, and can range from a single ontology to a combination of partial definitions of several ontologies, i.e., an ad-hoc domain specific ontology that reuses only part of Dublin Core and FOAF ontologies.
[3] That is, the exposed data satisfies the restrictions that apply to the data model according to the constrains defined by the business logic of the application.

The main difficulty in the development of *linked data aware* applications is that of validating the data exchanged with other parties during the enactment of its business processes. The implementation of a validation process requires considering different aspects related to the data sources implied, to the procedure that is to be implemented, and to the context in which the validation process will be carried out.

The purpose of this paper is to provide an insight on the different factors that have to be taken into account with regard to the abovementioned aspects when developing a validation process for a linked data aware application, that is, data source, procedure and application factors. Thus, Sections 2, 3, and 4 analyze the factors specific to each of these aspects, and Section 5 includes some discussion on the paper.

2. Data source factors

When implementing a validation process it is first and foremost necessary to have a clear understanding of the characteristics of the data sources that will be used, as these characteristics will determine the expectations the application will have with regards to the content provided by the data source as well as how the application provides data.

The first factor is the **dynamics** of the data source. For instance, the data source may be serving *static data*, that is, data that once it is published is never changed. However, the data source may also expose *dynamic data*, that is, data that is updated after being published. It is then relevant to know the periodicity with which the published data is updated, as it is different to deal with data sources which are updated periodically than with data sources that are randomly updated or that provide streams of data.

Another factor is related to the **publication strategy** used by the data source for exposing its data. Thus, the data source may *disallow inlining resource definitions*, that is, the contents available via a given URL only refer to the resource identified by that URL and no others (in other words, all the triples have as subject that URL or are blank nodes related to the subject resource). However, the data source may also *allow inlining resource definitions*, supporting in this way *resource aggregation* and *composition* patterns. On the one hand, in the case of the **resource aggregation pattern**, the contents available via a given URL include the definition of the resource identified by that URL as well as full or partial definitions of other individuals identified with their own URLs. On the other hand, in the case of the **resource composition pattern**, the contents available via a given URL include the definition of the resource identified by that URL as well as the full definition of the composite resources that are part of the main resource, where each of them is identified with a hash URL relative to the main URL.

In the same line, other important factor is the **provision strategy** used by the data source. Thus, the data source may expose only *raw data*, that is, only explicit triples. In this situation, if any reasoning about the triples is required during the validation process the requestor will be forced to do it by using the vocabularies used by the data source. On the contrary, the data source may also expose *materialized data*, serving partially-materialized definitions or fully-materialized definitions. On the former case, client-side reasoning might be required if the needs of the requestor go beyond the intended usage of the data source provider (the data source only materializes that implicit data that is relevant for their own purposes). On the latter case, no client-side reasoning is required as all the implicit data is included in the contents delivered.

Beyond the abovementioned factors that ultimately define which type of data a client may expect when dereferencing a URL, it is also necessary to understand the behavior of the data source with regards to other two factors: *access control* and *resource state management*.

With regards to **access control**, it is necessary to know the **granularity** provided by the implemented policies: do these policies provide *coarse-grained access* to the contents (retrieve all or nothing) or do they provide a *fine-grained access* to the contents (allow retrieving part of the contents depending on the identity of the user).

Finally, it is necessary to have an understanding about how **resource state** is managed by the data source. Thus, the data source may have a *unique state* for the published resources or may have *differentiated states* for the published resources depending on the identity of the requestor.

3. Procedure factors

Apart from having a clear understanding of the characteristics of the data sources, when designing the validation process it is also necessary to think in advance about the procedure itself.

Thus, the first thing to clarify is the **number of data sources** involved, as it is not the same implementing a validation procedure for a single data source than for multiple and possibly heterogeneous data sources.

Then, it is also necessary to identify the **validation scope**, differentiating between *validation in the small* and *validation in the large*. In the former case, the validation will be limited to the contents retrieved when dereferencing a resource together with locally available data, whereas in the latter the validation of the contents of a resource will also require dereferencing other linked data sources.

Another factor to determine is that of the **estimated duration** of the validation procedure. Thus, simple fast validation procedures that can be carried out in a short period of time will not pose the same problems as complex validation procedures that require lengthy operations which span a wide period of time.

Following with the temporal aspects, it is also necessary to identify the **immediateness** with which the validation process will be carried out, since it can be carried out *on-the-fly* as part of the retrieval operation, or it can be *deferred* and carried out *just-in-time* when the data is to be consumed.

Finally, it is also necessary to determine the **completeness of the data that is to be validated**. Thus, in the case of dealing with complete information, it will be possible to use *closed world reasoning* or *local-closed world reasoning* for validating the data. On the contrary, when the information to validate might be incomplete, *open world reasoning* at most will be possible for the validation process.

4. Context factors

Beyond the above mentioned factors, when implementing the validation process it is also necessary to understand the context in which the process is to be applied. When talking about applications, the context is the operation within which validation is required.

Thus, **write** operations may pose additional requirements that are not present on read operations. For example, when serving a *creation* request, a *linked data aware application* may impose additional restrictions on the data: apart from being syntactically and logically valid, there might be business-related restrictions that the data has to satisfy, for instance, with regards to the provenance of the data (i.e., when creating a bug in a bug tracker, the creator and assignees must be users of the bug-tracker).

In the same manner, **update** operations may pose different restrictions than creation ones. For instance, a *linked data aware application* may have certain parts of the model under its control, and disallow the modification of those tidbits.

5. Discussion

In this paper, we have focused on identifying requirements for RDF data validation. However, rather than focusing on conventional structural and value-range validation, we wanted to bring another perspective into the discussion by intentionally broadening our viewpoint and exploring the different factors that are related to validation in the context of Linked Data applications.

These factors are derived from looking at RDF data from the point of view of the requirements that have arisen in the ALM iStack project[4], which is centered on the development of a loosely-coupled Application Lifecycle Management platform based on Linked Data. In this scenario, the validation process requires considering different aspects related to the data sources implied, to the procedure that is to be implemented, and to the context in which the validation process will be carried out.

We believe these factors reveal several concerns that must be taken into to account when defining a language or standard way for doing RDF validation, whose value surpasses that of traditional structural and range-value validation techniques as they have a strong impact with regards the usage of the validated data.

Acknowledgments

The authors are partially supported by the ALM iStack project of the Center for Open Middleware[5].

[4] https://sites.google.com/a/centeropenmiddleware.com/alm-istack/
[5] http://www.centeropenmiddleware.com/