Building Linked Data Applications with Fusion: A Visual Interface for Exploration and Mapping

Samur Araujo[1], Geert-Jan Houben[1], Daniel Schwabe[2], Jan Hidders[1]

[1]Delft University of Technology, PO Box 5031, 2600 GA Delft, the Netherlands
[2]PUC-Rio, Rua Marques de Sao Vicente, 225, Rio de Janeiro, Brazil

{s.f.cardosodearaujo, g.j.p.m.houben, a.j.h.hidders}@tudelft.nl
dschwabe@inf.puc-rio.br

Abstract. Building applications over Linked Data often requires a mapping between the application model and the ontology underlying the source dataset in the Linked Data cloud. Explicitly formulating these mappings demands a comprehensive understanding of the underlying schemas (RDF ontologies) of the source and target datasets. This task can be supported by integrating the process of schema exploration into the mapping process and help the application designer with finding the implicit relationships that she wants to map. This demo describes Fusion - a framework for closing the gap between the application model and the underlying ontologies in the Linked Data cloud. Fusion simplifies the definition of mappings by providing a visual user interface that integrates the exploratory process and the mapping process. Its architecture allows the creation of new applications through the extension of existing Linked Data sources with additional data.

Keywords: semantic web, data interaction, data management, RDF mapping, Linked Data

1 Introduction

Nowadays, the Linked Data[1] cloud provides a new environment for building applications where many datasets are available for consumption. Although data in this cloud is ready to use, applications over the Linked Data cloud have currently an intrinsic characteristic: they consume RDF[2] data "as is", since designers do not have write permission over the data in the cloud which would enable them to change the data in any way. This fact raises an important issue concerning the development of applications over Linked Data: how to fill the gap between the ontology associated with the application model and the ontology used to represent the underlying data from the Linked Data cloud? The main benefit of mapping these two models is that

[1] Linked Data - http://linkeddata.org/
[2] http://www.w3.org/TR/2004/REC-rdf-primer-20040210/

Linked Data can then be accessed through properties defined in the application model, which is more convenient for the designer, consequently simplifying considerably the development and maintenance of the application.

This demo presents Fusion [1], a lightweight framework to support application designers in building applications over Linked Data. It supports designers in mapping the ontology of the used Linked Data sources to their application model by integrating the process of exploration of the target schema with the task of expressing a mapping rule itself. Fusion features a visual user interface that guides the designer in the process of specifying a mapping rule. It uses a standard RDF query language and allows Linked Data to be accessed using properties defined in the application model, consequently simplifying the use of Linked Data in a specific context.

2 Architecture Overview

The main aim of Fusion is to help the designer in discovering relationships in RDF graphs that exist in the Linked Data cloud and specifying rules for the derivation of new properties for these relationships. Fusion's architecture provides a complete environment to specify and execute a derivation rule. An overview of Fusion's architecture is shown in Fig. 1. The Fusion server engine is responsible for executing the derivation rule itself. During the process of executing a rule, it queries a source endpoint in the Linked Data, processes the results, and produces a set of new triples that will be added to the Fusion repository. Any RDF data store can be used as a Fusion repository. Currently, Fusion implements adapters for Sesame[6] and Virtuoso[7] data stores, although other adapters can be easily added to its architecture. All derived triples in Fusion contain as subject a resource whose URI belongs to the queried dataset, so the derived data is intrinsically interlinked with the Linked Data cloud. For this reason, a query over a federation of endpoints that includes the Fusion repository endpoint will allow the designer to have a view over the Linked Data that also includes the properties defined in her application model.

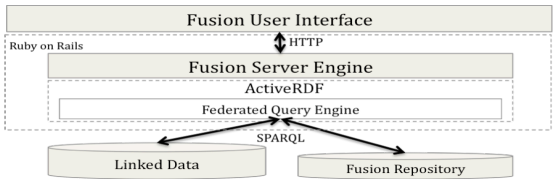

Fig. 1 – Fusion's architecture overview.

Fusion is implemented in Ruby on Rails[8] as a web application. It uses the ActiveRDF[9] API that allows an RDF graph to be accessed in the *object-oriented*

[6] http://www.openrdf.org/
[7] http://virtuoso.openlinksw.com/dataspace/dav/wiki/Main/
[8] http://rubyonrails.org/
[9] http://www.activerdf.org/

paradigm. By using this API the properties of an RDF *resource* can be accessed as an attribute of its corresponding Ruby[10] object. This architecture allows the designer to write complex functions for computing a new *datatype* property value using the full power of the Ruby language, which cannot be achieved simply by using the SPARQL language.

Derivation rules could be executed on demand, by any rule engine associated to the databases in the federation. However, even if theoretically possible, executing inference rules, or even instantiating a *virtual view*, over the Linked Data is still an open problem, since it raises many performance issues. Indeed, querying data that is already materialized is always faster than querying data that needs to be processed at runtime. Fusion avoids this problem by materializing the result of the rules as new triples in the Fusion repository when the derivation rules are defined.

3 Example of Use

This section presents a scenario that illustrates the use of Fusion to create an application by extending Linked Data sources with additional properties. Suppose that the designer wants to establish the relationship between US senators and the US state that they represent. Therefore she needs to construct a derivation rule that will find and define such a correspondence between politicians and states in the GovTrack.Us's Linked Data. In the first step in the process, the designer provides an example of two resources in GovTrack.Us that she knows in advance that are actually related, for instance, the politician Christopher Bond and the state of Missouri. Also, she needs to declare the GovTrack.Us endpoint to be queried and the maximum depth of the path. As the result of this first step, Fusion shows all the paths that connect these two example resources satisfying the maximum path length. This result is shown in Fig. 2. In this example, the paths found have a maximum length of 3.

Fig. 2 – Fusion's interface showing the discovered paths.

[10] http://www.ruby-lang.org/en/

In this view, the designer can now look for the path that has the intended semantics. Note that with this view the tool assists the designer in this discovery process, since she does not need to query the schema manually in order to find these paths. The first path shown in Fig. 2 indicates that the politician Christopher Bond has a role as senator representing the state Missouri, and in our example case the designer can now infer that this is an instance of the path that she is looking for. After this conclusion, the designer chooses that instance to be the template for the rule.

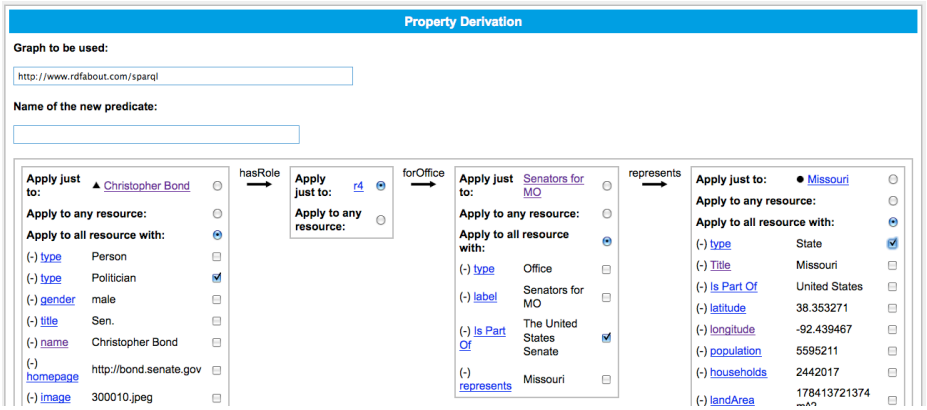

Fig. 3 – Generalizing the path for the property isSenatorOf in GovTrack.Us.

In the next step, shown in Fig. 3, the designer will define the derivation rule itself, which means that she visually formulates a query, which generalizes the selected path from the first step into a query that selects the elements to be connected through the property *isSenatorOf*. To complete this operation she also needs to define the graph where the derived triples will be stored and a specific URI to be used as the predicate of the new triples, which in this example will be the URI *http://example.org/isSenatorOf*. Note that in this example 3 nodes were generalized such that only paths between resources of the RDF type *Politician* and RDF type *State* that contain an intermediate node that *is part of* the United States Senate will be considered during the derivation process. Consequently, Fusion will derive the new property *isSenatorOf* for all instances of the class Politician that are connected to an instance of the class State through the designated path. The whole process ends with Fusion adding new triples to Fusion repository.

References

1. Araujo S., Houben G., Schwabe D., Hidders J. Fusion – Visually Exploring and Eliciting Relationships in Linked Data. In Proceedings of the 9th International Semantic Web Conference (ISWC2010). Shanghai, China. Nov 07-11, 2010.