Web Semantics: Science, Services and Agents on the World Wide Web 7 (2009) 204–219

Contents lists available at ScienceDirect

Web Semantics: Science, Services and Agents on the World Wide Web

journal homepage: www.elsevier.com/locate/websem

# Querying for provenance, trust, uncertainty and other meta knowledge in RDF☆

Renata Dividino*, Sergej Sizov, Steffen Staab, Bernhard Schueler

*ISWeb - Information Systems and Semantic Web, University of Koblenz-Landau, Universitätsstraße 1, 56072 Koblenz, Germany*

ARTICLE INFO

*Article history:*
Received 28 January 2009
Received in revised form 1 June 2009
Accepted 13 July 2009
Available online 24 July 2009

*Keywords:*
Semantic Web
Meta knowlegde
SPARQL
RDF

ABSTRACT

The Semantic Web is based on accessing and reusing RDF data from many different-sources, which one may assign different levels of authority and credibility. Existing Semantic Web query languages, like SPARQL, have targeted the retrieval, combination and re-use of facts, but have so far ignored all aspects of meta knowledge, such as origins, authorship, recency or certainty of data.

In this paper, we present an original, generic, formalized and implemented approach for managing many dimensions of meta knowledge, like source, authorship, certainty and others. The approach re-uses existing RDF modeling possibilities in order to represent meta knowledge. Then, it extends SPARQL query processing in such a way that given a SPARQL query for data, one may request meta knowledge without modifying the query proper. Thus, our approach achieves highly flexible and automatically coordinated querying for data and meta knowledge, while completely separating the two areas of concern.



## 1. Introduction

Integrating and re-using Semantic Web data becomes more and more fruitful and worthwhile in order to answer questions and deliver results. Typically, engines like Swoogle provide points of access for RDF data, crawlers may fetch relevant RDF data, and query languages like SPARQL with their corresponding query engines allow for selecting and re-using data in the appropriate format. With the arrival of more and more data in the Semantic Web and more sophisticated processing through query and reasoning engines, one now, however, encounters challenging questions linked to meta knowledge about the data like:

- Where is this data from?
- Who provided the data?
- When was this data provided?
- Was the provider certain about the truth of this data?
- Was the data believed by others, too?

For instance, when querying the Semantic Web with the help of SPARQL for the affiliation of a person named “James Hendler”, one finds (at least) two answers, i.e. “University of Maryland” and “Rensselaer Polytechnic Institute”. Without further indication as to where, by whom, when, etc. such information was given, it is impossible to decide which of the two affiliations is still valid.

The problem might be remedied in several ways. First, an ideosyncratic solution by the search engine, such as returning the corresponding RDF files or links to sources of knowledge extraction (say < http://www.cs.umd.edu/survey.pdf> and < http://www.rpi.edu/report.doc>), might help in this special case. However, an ideosyncratic solution may not be appropriate in a second case in which the ‘when’ was more relevant than the ‘where’ or in a third case where such a piece of information had to be aggregated from several resources. Second, the person or system requesting the meta knowledge might manually extend the SPARQL query formalizing the request for the affiliation in order to return the where, the who and the when. Such a modification will, however, be very tedious, as it will include a number of additional optional statements, and expressing it manually will be error prone. Also, it will not help in delivering meta knowledge that arises from joining several statements, e.g. meta knowledge about uncertainty that was based on several meta knowledge statements with different values of uncertainty. Therefore, querying Semantic Web data requires a principled, generic approach to the treatment of meta knowledge that is able to adapt to many dimensions of meta knowledge and that is open to accommodate to new dimensions when the need arises. Such a principled, original framework is given in this paper. We start to explain our approach with a discussion of important design choices in Section 2. We model meta knowledge in existing RDF structures by embedding a slightly more expressive

☆ This proposal is a completely revised and extended version of Schueler et al. [B. Schueler, S. Sizov, S. Staab, D.T. Tran, Querying for meta knowledge, WWW'08: Proceeding of the 17th International Conference on World Wide Web, ACM, New York, NY, USA, 2008]. Major changes include proofs showing that the meta knowledge evaluation of SPARQL queries is equivalent to the standard SPARQL evaluation, a discussion of soundness and completeness, as well as an extended empirical evaluation section.

* Corresponding author.
*E-mail addresses:* dividino@uni-koblenz.de (R. Dividino), sizov@uni-koblenz.de (S. Sizov), staab@uni-koblenz.de (S. Staab).

1570-8268/$ – see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.websem.2009.07.004

language, which we call RDF+, into RDF. We define the abstract syntax of RDF+, its semantics and its embedding in RDF in Section 4. In Section 5, we extend the SPARQL syntax and semantics to work on data and meta knowledge of RDF+. The extension allows the user to extend a given conventional SPARQL query by a keyword for meta knowledge triggering the construction of meta knowledge by the query processor. Section 6 summarizes the overall use and processing of SPARQL queries with meta knowledge. Sections 7 and 8 report on initial graceful results for meta knowledge processing from a theoretic point of view. Finally, Section 9 provides pointers to the prototype implementation of the system and initial experimental testing.

## 2. Scenario

In our scenario, the sample user aims to explore the knowledge and meta knowledge by querying knowledge extracted from Web pages of Computer Science departments. Here, we assume that he aims to find experts in the domain of Semantic Web and their affiliations. Example 2.1 shows the relevant facts that may have been obtained from websites of different universities.

**Example 2.1.** Relevant facts obtained from different websites

| | |
|---|---|
| Site A | JamesHendler's research topic is SemanticWeb.<br>JamesHendler is affiliated with RensselaerPI |
| Site B | JamesHendler's research topic Robotics.<br>JamesHendler affiliated with UnivMaryland.<br>RudiStuder's research topic SemanticWeb.<br>RudiStuder affiliated with Univ. Karlsruhe |

In addition, the user wants to exploit meta knowledge for obtaining results with best certainty and for analyzing contradicting answers (e.g. different affiliations for the same person "James Hendler" in Example 2.1). An example of meta knowledge associated to the extracted facts is presented in Example 2.2 and shows that the facts have been extracted from different-sources, at different timepoints, and with different degrees of extraction confidence.

**Example 2.2.** Associated meta knowledge of facts presented in Example 2.1

| | |
|---|---|
| Site A | source = www.rpi.edu/report.doc<br>certainty degree = 0.9<br>timestamp = 5/5/2007 |
| Site B | source = www.cs.umd.edu/survey.pdf<br>certainty degree = 0.6<br>timestamp = 6/6/2001 |

In the next section, we discuss design choices for representing meta knowledge taking into consideration the existing Semantic Web representation structures.

## 3. Design choices

This section summarizes and shortly motivates the design choices for our meta knowledge framework.

### 3.1. Reification

Establishing relationships between knowledge and meta knowledge requires appropriate reification mechanisms for supporting statements about statements. Our general objective is to execute queries on original data (i.e. without meta knowledge) directly without complex transformations. For compliance with existing applications that access the repository in a common way (e.g. using SPARQL queries), we do not modify existing user data. This requirement does not allow us to use mechanisms like RDF reification, which decompose existing triples and fully change the representational model. In our framework described in section 4, we adopt the notion of Named RDF Graphs for meta knowledge representation [5,6].

### 3.2. Storage mechanisms

Following the overall philosophy of RDF, we do not separate meta knowledge from "normal" user knowledge in the repository. Following this paradigm, a user or developer has unlimited access to all contents of the triple store and can manipulate meta knowledge directly. In other words, the user can directly access meta knowledge (e.g. using suitable SPARQL queries). Beyond explicitly designed queries for meta knowledge access, in Section 5 we describe the extension of SPARQL that allows us to access meta knowledge about the result set automatically without user intervention.

### 3.3. Dimensions of meta knowledge

An important point for the application design is the definition of relevant meta knowledge properties and their suitable interpretation for arbitrarily complex query patterns. In general, these properties are application dependent and must be carefully chosen by the system administrator. In our scenario (Sections 2 and 6) we discuss common and widely used properties, such as timestamp, source, and (un)certainty, and show ways of defining and utilizing them in our framework.

### 3.4. Syntax extensions

Seamlessly integrated access to meta knowledge requires corresponding extensions of existing querying mechanisms. These can be realized at different levels, for instance at the level of query languages (e.g. SPARQL) or at the level of application-specific interfaces (e.g. Sesame API). In Section 5 we describe our SPARQL extension for constructing query results with associated meta knowledge. It is system-independent and not related to some particular implementation of the RDF repository. Furthermore, it fully supports the existing SPARQL syntax and semantics. Compliance with existing established standards makes the integration with existing applications and interfaces substantially easier.

## 4. Syntax and semantics for RDF with meta knowledge

In order to adapt our sample application scenario to our framework, we formalize it using the notion of Named RDF Graphs [5,6]. We assume that the user utilizes knowledge which has been initially extracted from Web pages of Computer Science departments and stored in form of RDF triples in his personal 'active space' [21], backed by a local RDF repository. Example 4.1 shows the relevant facts already presented in Example 2.1 in RDF triple language TriG [1] with Named Graphs in a simplified form that abstracts from (default) namespaces.

**Example 4.1.** Facts of Example 2.1 represented in TriG with Named Graphs

| | |
|---|---|
| *G* 1 | { JamesHendler researchTopic SemanticWeb.<br>JamesHendler affiliatedWith RensselaerPI } |
| *G* 2 | { JamesHendler researchTopic Robotics.<br>JamesHendler affiliatedWith UnivMaryland.<br>RudiStuder researchTopic SemanticWeb.<br>RudiStuder affiliatedWith UnivKarlsruhe } |

Likewise, the meta knowledge associated to the extracted knowledge presented in Example 2.2 is stored into the same RDF repository using the notion of Named RDF Graphs.