RDF & RDF QUERY LANGUAGES – BUILDING BLOCKS FOR THE SEMANTIC WEB

Adam ALTĂR – SAMUEL* [1]
Alexandru COSTIN [2]
Dragoș ENACHE [3]

ABSTRACT
Although the World Wide Web is a relatively new concept, its expansion and the impact it had on the population were great. The Web is characterized by a network of interlinked documents, most of them designed to be read by humans, not interpreted by machines. The Semantic Web comes as a complement to the current web in which the available information can be interpreted by machines also, not only by people. Web services provide methods of publishing services in an UDDI registry, description of interfaces using WSDL and messaging through the network using various protocols, the most common being SOAP. This paper examines the close link between The Semantic Web and Web services, as well as other technologies necessary for this interaction.

Keywords: semantic web, web services, RDF (Resource Description Framework), SPARQL, WSDL (Web Service Description Language), OWL (Web Ontology Language)

1. Introduction

The Semantic Web is an extension of the current web, which allows searching, using and combining data in an easier manner. It is based on information and machine interpretable metadata language expressed in RDF (Resource Description Framework).
This implies that computers should be capable of processing the growing amount of data found on the Internet. Nowadays, Web content is designed to be understood by people, but problems arise when a machine needs to distinguish relevant content from all the information received, in particular because of the vast variety of digital types of content and viewers.
The Semantic Web will complement the current web, creating an environment where software agents will be able to handle various sophisticated tasks in which information will have a well-defined meaning. Thus, there's hope that in the near future, computers could not only display, but also "understand" data.
The concept was revealed by Tim Berners-Lee, who also invented the WWW (World Wide Web), URI (Uniform Resource Identifier), HTTP (HyperText Transfer Protocol) and HTML (HyperText Markup Language).
The Semantic Web can become an excellent platform for e-Learning. The characteristics of the semantic web, i.e. well defined meaning of concepts and automatically processable

[1] *Corresonding author. Teaching Assistant, Department of Computer Science, Mathematics and Statistics, Romanian – American University, 1B, Expozitiei Blvd., district 1, code 012101, Bucharest, Romania, Email: adamalt@gmail.com
[2] Faculty of Computer Science for Business Management, Romanian – American University, 1B, Expozitiei Blvd., district 1, code 012101, Bucharest, Romania, Email: alexandru.costin@itc.ro
[3] Faculty of Computer Science for Business Management, Romanian – American University, 1B, Expozitiei Blvd., district 1, code 012101, Bucharest, Romania, Email: Dragos_rash@yahoo.com

metadata, used by appropriate software agents, establish an efficient approach for satisfying the requirements of eLearning. Learning materials can be interpreted semantically and, at the request of users, reorganized in order to create new didactic modules. Based on the user's requests and preferences, learning materials and other information considered relevant can be combined in a simple and intuitive manner. This process is based on semantic queries and navigation through the learning materials and is possible through the use of ontologies, which provide exact definitions of concepts and notions.

The technologies underlying the Semantic Web are still at an early stage and although the future looks very promising, there are still debates regarding the direction and the features of The Semantic Web.
Information that can be found on the web, in HTML files, can be useful in some cases, but not always. A Google search returns about 25% of the total relevant results, and, sometimes, searches do not return any results, even if there are relevant sites for that search.
For example, you can easily find sites that offer information on weather, local events, TV programs, etc., but all are presented in HTML. The problem is that in certain contexts, gathering only the necessary data and using it in a different environment can be very difficult.

2. Web Semantics

The Semantic Web can be seen as an "engineering" solution. It will be built using URI syntax. An URI is simply a Web identifier (character strings starting with "http:" or "ftp:", etc.). Anyone can create an URI and any resource that has an URI can be found on the web.
For the development of the Semantic Web, a new language was created, which uses a triplet of URIs (analogue to the subject, predicate and complement of a sentence). This language is called RDF (Resource Description Framework) and aims at standardizing the processes of sending and receiving "machine-understandable" data. RDF is based on XML (eXtensible Markup Language). Once data is expressed in RDF, its processing becomes easy, because there are already many XML parsers widely available.

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:dc="http://85.204.140.247/dc/#"
   xmlns:eu=" http://85.204.140.247/adamalt/#" >
   <rdf:Description rdf:about="">
      <dc:creator rdf:parseType="Resource">
         <eu:nume>Adam Altar </eu:nume>
      </dc:creator>

      <dc:title> RDF and Web Services: Buiding – blocks for the Semantic Web:
</dc:title>
   </rdf:Description>
</rdf:RDF>
```

This piece of RDF code "says" that this paper has the title "RDF and Web Services: Building – blocks for the Semantic Web" and was written by Adam Altar.
In order to lay the backbone of The Semantic Web, data should be published in RDF. Using an RDF document, users can define objects with various properties and the relationships between them. RDF documents use URIs to encode data, ensuring that concepts are not just words, but have a unique definition that everyone can find on the Internet (at the address specified by the URI).
For example, suppose we have access to a database with information on certain people (including the city where they live). To find all the people living in a certain city, a software agent should “know” that the field City is the field that contains the names of the cities associated with each person. This can be specified using RDF files.
RDF is a standard model for data interchange on the Web. RDF has features that facilitate data merging even if the underlying schemas differ, and it specifically supports the evolution of schemas over time without requiring all the data consumers to be changed.
RDF extends the linking structure of the Web to use URIs to name the relationship between things as well as the two ends of the link (this is usually referred to as a “triple”). Using this simple model, it allows structured and semi-structured data to be mixed, exposed, and shared across different applications.
This linking structure forms a directed, labeled graph, where the edges represent the named link between two resources, represented by the graph nodes. This graph view is the easiest possible mental model for RDF and is often used in easy-to-understand visual explanations. For example, Figure 1 presents an RDF graph that describes the author of this paper.

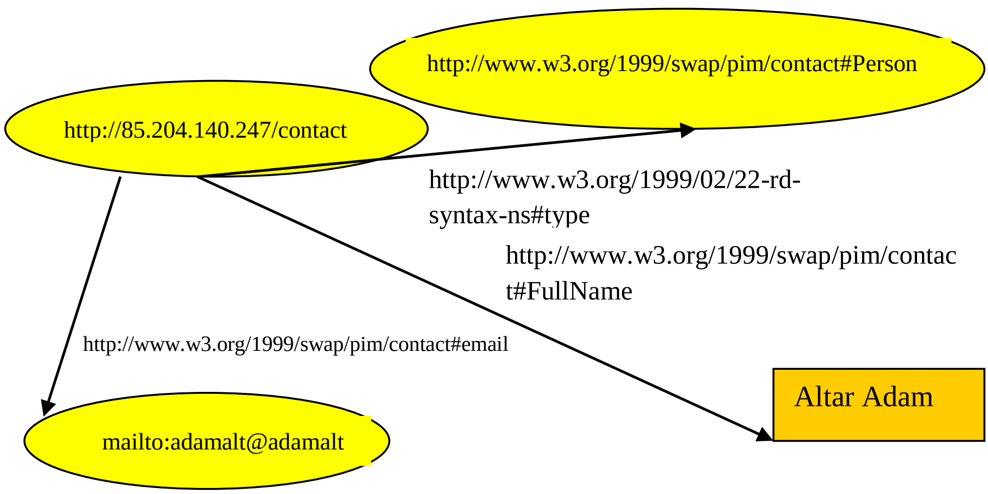

Figure 1

3. Direct Mapping

Direct mapping defines an RDF graph representation of the data in relational databases – in other words, it is a conversion process for database relations that transform them into RDF-compliant graphs. The relational database is taken as input and a direct graph is generated, taking into account the foreign and primary keys, plus the values in each row. There are a few tools that users can use to map their data.

Virtuoso is one of those tools. An OpenLink Software product, it focuses on data management and administration, including direct mapping, which has become one of its focus points. When it comes to RDF, this software prides itself with such features such as SPARQL Protocol and Query Language support, RDF Data Management and SPARQL integration within SQL. Virtuoso is not free, but there is also an open-source version which doesn't include the virtual database engine and data replication functionality.

„The era of the Semantic / Linked-Data Web - a time when we will construct mesh-ups where today we assemble mash-ups - is nigh. A confluence of industry standards and product developments is setting the stage for the next major evolutionary stage of the World Wide Web."

(Virtuoso RDF - About)

Spyder is another useful tool, and unlike Virtuoso, it's free and is less business-oriented, focusing only on direct mapping. It exposes relational data as if it were stored as RDF, making the data available to any SPARQL-compliant query on a website. Spyder makes use of W3C standards in its operations: SPARQL for queries, SQL for communicating with data sources etc. Compatible with Oracle, DB2, MySQL and SQLServer, this handy tool is one of the favorites when it comes to direct mapping for personal use.

The D2RQ Platform is a open source system for accessing relational databases as virtual, read-only RDF graphs. It offers RDF-based access to the content of relational databases without having to replicate it into an RDF store. D2RQ is able to query a non-RDF databse using SPARQL, access the content of the database as Linked Data over the web and access information in a non-RDF database.
The D2RQ Platform has three main features:D2RQ Mapping Language, D2RQ Engine, D2R Server.
D2RQ Mapping language is used for describing the relation between an ontology and an relational data model.
D2RQ Engine calls SQL queries against the database and passes query results up to the higher layers of the frameworks.
D2R Server HTTP server that provides a Linked Data view,HTML view for debugging and a SPARQL protocol enpoint over the database.
It supports databases such as Oracle, MySQL, SQL Server and others.

4. Ontologies

Another very important part of the Semantic Web will be represented by ontologies.

Suppose we have two databases with information about specific individuals, one with the fields Name, Surname, Street, City, and the other with the fields: full name and address. Clearly, the two databases express the same thing and an application that interacts with the two must "know" this. So, the application must have a way of discovering when concepts defined in different ways actually express the same thing. The solution can be found in ontologies.

The Philosophical definition of an ontology is: the metaphysical study of the nature of being and existence. In the case of The Semantic Web, an ontology represents a file that defines relationships between terms. In general, an ontology has a taxonomy and a set of rules of inference.

Taxonomies define classes of objects, and the relationships between them. For example, an address is a type of location and a zip code only applies to locations. Thus, a large number of connections can be expressed between objects, setting properties for certain classes (subclasses receiving class properties).

Using rules of inference, an application can "deduce" certain relationships between objects. For example, if the person A lives on S Street and S Street is in town T, than A lives in T.

Ontologies could boost the functioning of the web in many ways. They can be used to improve search results – a search program will return only pages that refer to a well-defined concept, rather than returning pages based on keywords (obviously much less effective). More advanced applications will use ontologies to "understand" the definitions of certain concepts. Currently, there are different programs that can create ontologies, but the one that stands out by being the most flexible and by incorporating most Semantic Web requirements. The application is named Protégé (more about this project can be found at http://protege.stanford.edu).

The Semantic Web will really come into being when many applications (agents) that collect data from the web, from various sources, process it and then exchange results with other such applications will exist. The effectiveness of these agents will grow exponentially with the increase of the amount of data that can be processed by computers.

An important thing that agents should have is the ability to exchange "proofs" between them in the unifying language of the Semantic Web (the language which expresses logical inferences drawn using rules and information found in ontologies). For example, suppose information is found about a person X, who lives in Timișoara. Obviously, we want to know if this information is correct and if that person X is the person we seek. In order to accomplish this, the agent will invoke the service that returned data about person X, which, in turn, will return to the agent the checks required and, if needed, will return a list of relevant sites where information about person X is found. Although the unifying language is still at an early stage of development, a limited number of applications that can exchange data between them using this preliminary version of the language already exist.

A complete scenario for an agent (a semantic application) which will "browse" the Semantic Web is the following:

Suppose a user wants to use the application to make arrangements to spend a week in London. He writes the sentence "I want to go to London on Holiday from 8 to 15 June 2007". The application will retrieve the text and process it, using RDF and ontologies.

After processing, the application will "know" that it should invoke a web service that makes flight bookings and one that makes hotel reservations. The application then interacts with both services and makes the necessary reservations, based on certain conditions that the user requires, and then returns a list of other possible services that might be of interest to the user on the trip to London.
Another important feature of the Semantic Web is digital signatures. These are encrypted data that ensures the agent that it can trust the source that sent the information. For example, if someone makes a payment through an agent, the agent must ensure that payment is made to the bank and not to the hacker across the street. Agents should be "skeptical" about the information received, until it is verified by digital signatures.
If built correctly, the semantic web will revolutionize the World Wide Web and its use, ensuring a more efficient exchange of information and the discovery of more relevant information quickly and efficiently.
A very important part of the Semantic Web (perhaps the most developed at this point) is represented by web services.

5. RDF Query Languages

RDF is considered to be the most relevant standard for data representation and exchange on the Semantic Web. This implies that huge RDF databases will be created and then searched through. Although one could use SQL and a relational database to accomplish this goal, this process would be very cumbersome, since SQL has a limited and inflexible syntax and RDF concepts are represented in the form of triplets (Subject, Predicate, Object). In order to solve this problem, new languages, called RDF Query Languages, have emerged.
An RDF Query Language can be characterized by five distinct properties: expressiveness, closure, adequacy, orthogonality, safety.
Expressiveness indicates how powerful queries can be formulated in a given language. Typically, a language should at least provide the means offered by relational algebra, i.e. be relationally complete. Usually, expressiveness is restricted to maintain other properties such as safety and to allow an efficient (and optimizable) execution of queries.
The closure property requires that the results of an operation are again elements of the data model. This means that if a query language operates on the graph data model, the query results would again have to be graphs.
A query language is called adequate if it uses all concepts of the underlying data model. This property therefore complements the closure property: For the closure, a query result must not be outside the data model, for adequacy the entire data model needs to be exploited.
The orthogonality of a query language requires that all operations may be used independent of the usage context.
A query language is considered safe, if every query that is syntactically correct returns a finite set of results (on a finite data set). Typical concepts that cause query languages to be unsafe are recursion, negation and built-in functions.
The underlying data model directly influences the set of operations that should be provided by a query language. The underlying structure of any RDF document is a collection of triples. This collection of triples is usually called the RDF graph. Each triple

states a relationship (aka. edge, property) between two nodes (aka. resource) in the graph. This abstract data model is independent of a concrete serialization syntax. Therefore query languages usually do not provide features to query serialization-specifc features, e.g. order of serialization.

RDF has a formal semantics which provides a dependable basis for reasoning about the meaning of an RDF graph. This reasoning is usually called entailment. Entailment rules state which implicit information can be inferred from explicit information. Hence, RDF query languages can consider such entailment and might convey means to distinguish implicit from explicit data.

XML data types can be used to represent data values in RDF. XML Schema also provides an extensibility framework suitable for defining new datatypes for use in RDF. Data types should therefore be supported in a RDF query language.

In general, it is not assumed that complete information about any resource is available in the RDF query. A query language should be aware of this and should tolerate incomplete or contradicting information.

An RDF query language is a computer language, specifically a query language for databases, able to retrieve and manipulate data stored in Resource Description Framework format.

SPARQL is emerging as the de facto RDF query language, and is a W3C recommendation. Released as a Candidate Recommendation in April 2006, it returned to Working Draft status in October 2006, due to open issues. It returned to Candidate Recommendation status in June 2007. On 12 November 2007 the status of SPARQL changed into Proposed Recommendation. On 15 January 2008, SPARQL was standardized.

SPARQL uses four different types of queries: *SELECT*, *ASK*, *DESCRIBE* and *CONSTRUCT*. *DESCRIBE* and *CONSTRUCT BOTH* return data in RDF/XML, but in the case of the first query, the query engine decides what data is relevant for a particular query, while in the case of the latter, the programmer decides what the relevant data is. These types of queries are very useful especially when the client application extracts a subset of the available data for further RDF processing.

SELECT queries are similar to the ones in SQL. For example, this query should list all the names of all the people represented in an RDF graph:

```
PREFIX foaf: <http://xmlns.com/foaf/0.1>
SELECT ?name
WHERE
{
  ?person foaf:name ?name.
}
```

The keyword *PREFIX* provides the XML namespaces associated with each prefix, the *SELECT* clause specifies the data that should be returned, while the *WHERE* clause represents the filter.

ASK queries are specialized versions of *SELECT* queries, but only return a YES/NO answer (Yes, if the needed data is found and no, otherwise). For example, this query will return true if there exists at least one document written by Adam Altar and false, otherwise.

```
PREFIX dc: <http://85.204.140.247/dc/#>
ASK WHERE
{
  ?document dc:creator "Adam Altar".
}
```

ASK queries are very useful in interacting with web services, particularly new found ones. A simple *ASK* query (which is much faster than a *SELECT*) can reveal whether the particular web service is of interest or not. Moreover, these queries can be serialized in the format SPARQL XML.

SPARQL XML is a very simple XML vocabulary that standardizes the way results of a SPARQL query are formatted. The format consists of a handful of elements belonging to a single namespace. Compared to RDF, which provides a variety of possibilities for serializing the same data, SPARQL XML is much more regulated. This implies much easier futher processing using XSLT or another XML parser. In conclusion, SPARQL can be used in “XML only” contexts; no RDF parsing is needed. The following example illustrates the SPARQL XML format:

```
<sparql xmlns="http://www.w3.org/2005/sparql-results#">
  <head>
    <variable name="name"/>
  </head>
  <results ordered="false" distinct="false">
    <result>
      <binding name="name">
        <literal datatype="http://www.w3.org/2001/XMLSchema#string">Gica
        </literal>
      </binding>
    </result>
   <!-- mai multe rezultate -->
   </results>
</sparql>
```

Similar to a SQL query, a SPARQL query returns tabulated results that can be easily transformed using XSLT.

The *sparql* root element contains a *head* element which lists the variables returned for each row (the columns). The *results* element contains one *results* element for each row. Additional elements (*literal*, *etc*) provide the type of resource (an RDF concept or just a simple string).

SPARQL allows users to write queries against data that can loosely be called "key-value" data or, more specifically, data that follows the RDF specification of the W3C. The entire

database is thus a set of "subject-predicate-object" triples. This is analogous to some NoSQL databases' usage of the term "document-key-value", such as MongoDB.
RDF data can also be considered in SQL relational database terms as a table with three columns - the subject column, the predicate column and the object column. Unlike relational databases, the object column is heterogeneous, the per-cell data type is usually implied (or specified in the ontology) by the predicate value. Alternately, again comparing to SQL relational, all of the triples for a given subject could be represented as a row, with the subject being the primary key and each possible predicate being a column and the object is the value in the cell. However, SPARQL/RDF becomes easier and more powerful for columns that could contain multiple values (like "children"), and where the column itself could be a joinable variable in the query, rather than directly specified.
SPARQL thus provides a full set of analytic query operations such as JOIN, SORT, AGGREGATE for data whose schema is intrinsically part of the data rather than requiring a separate schema definition. Schema information (the ontology) is often provided externally, though, to allow different datasets to be joined in an unambiguous manner. In addition, SPARQL provides specific graph traversal syntax for data that can be thought of as a graph.

6. Conclusions

Although the Semantic Web is still in its early stages, the future looks very promising. Potential benefits that can improve today's Web come from the fact that, with the development of the Semantic Web, computers will be able to "understand" the enormous amount of information that currently exists online. A computer could instantly find the Chinese restaurant closest to home and reserve a table.
Together with ontologies and web services and using languages already familiar to programmers in order to exchange messages, the RDF standard will aid in the development of The Semantic Web, and, hopefully, in a few years, we will use the Web to find exactly the relevant information for our search.

REFERENCES

[1] G. Bela, H. Piroska, Middleware *for Automated Implementation of Security Protocols*, 6th European Semantic Web Conference, Heraklion, Greece, 31 May - 4 June, Lecture Notes in Computer Science (LNCS 5554), L. Aroyo et al. (Eds.), Springer-Verlag, pp. 476-490, 2009, ISBN 978-3-642-02120-6
[2] P. Hasse, J. Broekstra, A. Eberhart, R. Volz, "*A Comparison of RDF Query Languages*", AIFB Institute, University of Karlshrue, Germany, 2007
[3] J. Hebeler et al, "*Semantic Web Programming*", Wiley Publishers (2009), ISBN: 978-0470418017
[4] H. Kreger, „*Fulfilling the Web Service Promise*", Communications of the ACM, June 2003, Vol. 46, No. 6
[5] S. McHraith, T. Cao Sun, H. Zeng, „*Semantic Web Services*", Stanford University Working Papers, No. 7/2008

[6] S. McHraith, D. Martin, "*Bringing Semantics to Web Servicies"*, Stanford University Working Papers, No. 4/203

[7] A. Polleres, "*From SPARQL to Rules (and back)*", Proceedings of the 16th international conference on World Wide Web, Madrid, Spain, pp. 787 – 796, ISBN: 978-1-59593-654-7

[8] T. Segaran, "Programming *Collective Intelligence: Building Smart Web 2.0 Applications*", O'Reilly Media, 2007, ISBN: 978-0596529321

[9] „*Studies on the Semantic Web*", IOS Press Amsterdam & AKA Verlag Heidelberg (2009+), ISSN 1868-1158

[10] Semantic Web Resource Centers: http://www.Xml.com/semweb ; http://www.semanticweb.org

[11] *http://www.w3.org/2001/sw/ - W3 Official Web Servicies Specification*

[12] *http://www.w3.org/TR/soap/ - W3 Official SOAP Specification*

[13] *http://www.w3.org/TR/wsdl/ - W3 Official WSDL Specification*

[14]*http://www.w3.org/TR/rdb-direct-mapping/#emp-addr – W3 Official Direct Mapping Specification*

[15]*http://www.revelytix.com/?q=content/spyder&phpMyAdmin=2acdy2DeLD5inPb43VaD6lMxxT5 – Spyder RDF Tool Description*

[16]*http://virtuoso.openlinksw.com/dataspace/doc/dav/wiki/Main/VOSRDF - Virtuoso RDF Tool Open-Source Version Description*

[17]*http://virtuoso.openlinksw.com/rdf-quad-store/ - Virtuoso Commercial Description*

[18]*http://d2rq.org/ - D2RQ Official Web Page*

The article has been supported by scientific research within the project entitled "PRACTICAL SCHOOL: Innovation in Higher Education and Success on the Labour Market", project identified as POSDRU/156/1.2/G/132920. The project is co-financed by the European Social Fund through the Sectorial Operational Programme for Human Resources Development 2007-2013. Investing in people!