RDB2Onto: Approach for creating semantic metadata from relational database data

Martin Šeleng, Michal Laclavík, Zoltán Balogh, Ladislav Hluchý
Institute of Informatics, Slovak Academy of Sciences,
Dúbravská cesta 9, 845 07 Bratislava, Slovakia
e-mail: martin.seleng@savba.sk

Abstract

In this paper we present the approach for creating semantic metadata from relational database data. When building ontology-based information systems, it is often needed to convert or replicate data from existing information systems (such as databases) to the ontology based information systems, if we want the ontology-based systems to work with real data. RDB2Onto tool converts selected data from a relational database to a RDF/OWL ontology document based on a defined template. Such filled in templates can be then stored to the ontology-based knowledge memory. In the paper we also evaluate the tool against existing solutions, such as RDQUERY or D2RQ.

Keywords: RDF/OWL, semantic, relational database, ontologies, semantics, information systems

1. Introduction

Building ontology based information systems, it is frequently necessary to convert or replicate data from existing information systems such as databases to the ontology based information systems, if the ontology based systems want to work with real data. Usually data in existing information systems are stored in a Relational Database. Such problem arises also in the NAZOU project[1] where some knowledge acquisition and maintenance tools store results data only in the RDB database. Due to the common presentation frame work [5] the result data need to appear also in its ontology form and this is the place where RDB2Onto plays its role. In addition, a large quantity of data can be found on the web automatically generated from relational databases, often referred to as the Deep Web [1]. If we want to create web content based on semantic web technologies such as OWL[2], we have to solve conversion of RDB data to ontology data, while several approaches exist. Most of them are usually based on creating new quite complicated mapping languages like D2R MAP [4], D2R [3] or R2O [2]. In our approach we try to give a more simple solution based on SQL queries and RDF/OWL templates which are filled in with results of SQL query.

2. Overview of the Approach

The goal of the tool is to provide Relational Database Data to Ontology Individuals Mapping. The tool works on a domain ontology model and a relational database. The overall idea is to map SQL query to RDF/OWL XML template. Such OWL data are then sent to an ontology model. The tool is being implemented in Java using Jena[3] or Sesame[4] library for ontology manipulation and MySQL database for testing but it is possible to use any other relational database using JDBC connector. Architecture of the tool is shown in Figure 1.

[1] http://nazou.fiit.stuba.sk/
[2] http://www.w3.org/TR/owlfeatures/
[3] http://www.sf.net
[4] http://www.openrdf.org/

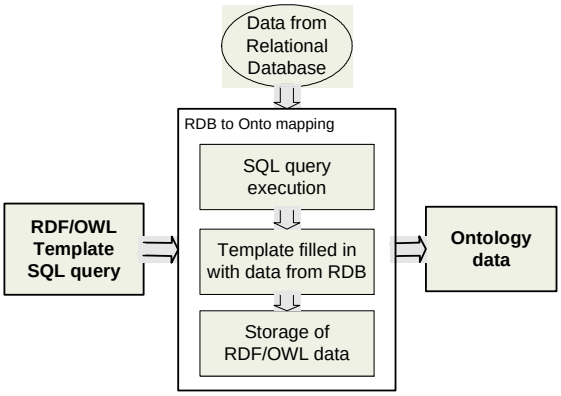

Figure 1*: RDB2Onto Architecture*

It contains 3 basic steps which are explained on the following example: There is a *document* table with following fields: *id*, *url*, *original_doc_path*, *converted_doc_path*, *download_date*, *lang* in relational database. In this example SQL query will look as follows:

```
SELECT
        id, url, original_doc_path, converted_doc_path,
        download_date, IF(lang = 'sk', 'Slovak', 'English')
        AS lang
FROM
        document
```

The SQL query is executed and for each row of the query results it fills in the XML-based OWL template. Each element enclosed with {} brackets is replaced with adequate value from SQL query for a given row and composed OWL data are stored to the ontology model.

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
     xmlns:jo="http://nazou.fiit.stuba.sk/nazou/ontologies/v0.6.17/offer-job#"
     xmlns:inst="http://nazou.fiit.stuba.sk/nazou/ontologies/v0.6.17/offer-job-inst#"
     xmlns:c="http://nazou.fiit.stuba.sk/nazou/ontologies/v0.6.17/classification#"
     xmlns:ofr="http://nazou.fiit.stuba.sk/nazou/ontologies/v0.6.17/offer#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
     <rdf:Description rdf:about="offer-job-inst:jo_{id}">
     <rdf:type rdf:resource="offer-job:JobOffer"/>
     <ofr:hasSource rdf:resource="offer-job-inst:source_{id}"/>
     <ofr:hasOfferCreator rdf:resource="offer-job-inst:OfferCreator_NAZOU_RDB2Onto"/>
     </rdf:Description>
     <rdf:Description rdf:about="offer-job-inst:source_{id}">
     <rdf:type rdf:resource="offer:OfferSource"/>
     <ofr:acquisitionDate>{download_date}</ofr:acquisitionDate>
     <ofr:originalURI>{url}</ofr:originalURI>
     <ofr:localURI>{original_doc_path}</ofr:localURI>
     <ofr:localConvertedURI>{converted_doc_path}</ofr:localConvertedURI>
     <ofr:language rdf:resource="region:{lang}"/>
     </rdf:Description>
</rdf:RDF>
```

3. Existing Solutions

3.1 RDQuery[5]

RDQuery is a wrapper system which makes relational databases accessible for Semantic Web applications using an RDF query language (RDF-QL). RDQuery currently supports RDQL [7] and its successor SPARQL [6], which will hopefully be recommended soon by the W3C as the de facto standard query language for RDF. Nevertheless, RDQuery may easily be adapted to future developments adding specific parsers for other query languages.

[5] http://sourceforge.net/projects/rdquery/

3.2 D2RQ[6]

D2RQ is implemented as a Jena graph, the basic information representation object within the Jena framework [8]. A D2RQ graph wraps one or more local relational databases into a virtual, readonly RDF graph. D2RQ rewrites RDQL queries and Jena API calls into application-datamodel-specific SQL queries. The result sets of these SQL queries are transformed into RDF triples which are passed up to the higher layers of the Jena framework.

4. Evaluation

In our evaluation we will focus only on speed measurements because the tools can't do a mistake. The probability of a mistake depends only on SQL or SPARQL query and RDF/OWL template, which are exactly created by the humans. All experiments will be conducted on Sesame repository and MYSQL[7] database. If we use our SQL query (Fig. 1) translated to SPARQL language, which most users are not familiar with (this is main disadvantage of RDQuery tool), we achieve the following results for RDQuery tool.

	Execution Time [s]	
SPARQL Query	Query Translation	Mapping Process
1 Query	0.018	0.052

Table 1: *Results achieved by RDQuery*

If we want to evaluate the D2RQ tool, we must create a template as in RDB2Onto. This template can be created automatically by generate-mapping tool and then it can be edited by a user. Then we use dump-rdf tool which will generate RDF/OWL file and then we send this file to Sesame repository. The results are shown in the following table.

	Execution Time [s]	
SQL Query	Query Translation	Mapping Process
1 query	0.006	0.030

Table 2: *Results achieved by D2RQ tool*

The query translation is very fast because there is no query translation (it is only SQL query to MYSQL database).

[6] http://sourceforge.net/projects/d2rq-map/
[7] http://www.mysql.com/

Using our tool (RDB2Onto), SQL query translation will be very fast, similar to D2RQ tool, as there is no query translation. Also the performance of the mapping process will be comparable to that of D2RQ tool. The results are shown in Table 3.

	Execution Time [s]	
SQL Query	Query Translation	Mapping Process
1 Query	0.007	0.034

Table 3: *Results achieved by RDB2Onto*

5. Conclusion

Comparing the results in Table 1-3 we can see that our solution has comparable performance. RDB2Onto has worse performance than D2RQ tool but its execution time is much better than that of RDQuery tool. Advantage of RDB2Onto tool is its simplicity. It is possible to receive any complicated data from relational database upon an SQL request. The solution can be simply configured for specific data mapping. The technologies such as [2,3,4] provide infrastructure and languages for relational data mapping by setting data dependencies between RDB data and ontology. Such solutions can be sometimes too complicated and they can require too much effort to learn, set up and use. RDB2Onto requires only knowledge of SQL and RDF/OWL and thus can be applied with minimum effort.

Acknowledgments

This work is partially supported by projects NAZOU SPVV 1025/2004, RAPORT APVT-51-024604, VEGA 2/7098/27.

References

[1] Bergman MK. The Deep Web: Surfacing hidden value. White paper. Sept 2001

[2] J. Barrasa, Ó. Corcho, A. Gómez-Pérez: R2O, an Extensible and Semantically Based Data-baseto-ontology Mapping Language, Second Workshop on Semantic Web and Databases (SWDB2004). Toronto, Canada. August 2004.

[3] Bizer C. D2R MAP – A DB to RDF Mapping Language. 12th International World Wide Web Conference, Budapest. May 2003

[4] Barrasa J, Corcho O, Gómez-Pérez A. FundFinder – A case study of Database-to-ontology mapping. Semantic Integration Workshop, ISWC 2003. Sanibel Island, Florida. Sept 2003

[5] Návrat, P., Bieliková, M., Rozinajová, V.: Methods and Tools for Acquiring and Presenting Information and Knowledge in the Web. In: CompSysTech 2005, B. Rachev, A. Smrikarov (Eds.), Varna, Bulgaria, June 2005. – pp. IIIB.7.1-IIIB.7.6.

[6] E. Prud'hommeaux and A. Seaborne. SPARQL Query Language for RDF. http://www.w3.org/TR/2006/CR-rdf-sparql-query-20060406/, 2006. W3C Candidate Recommendation.

[7] A. Seaborne. RDQL - A Query Language for RDF. http://www.w3.org/Submission/2004/SUBM-RDQL-20040109/, 2004. W3C Member Submission.

[8] Jeremy J. Carroll *et al*.: Jena: Implementing the Semantic Web Recommendations. In Proceedings of the 13th World Wide Web Conference New York City, May 2004.