SPARQL to SQL query translation using R2RML mappings

Project Proposal for the MSc in Advanced Computing Technologies

by Sebastian Holzschuher

Department of Computer Science and Information Systems

Birkbeck College, University of London

April 2015

Academic Declaration

This proposal is substantially the result of my own work except where explicitly indicated in the text. I give my permission for it to be submitted to the JISC Plagiarism Detection Service. I have read and understood the sections on plagiarism in the Programme Handbook and the College website.

Table of Contents

1 Introduction

The Semantic Web is an initiative of the World Wide Web consortium (W3C) that aims at defining standards and technologies for annotating and linking information in a machine-readable format on the web. The current World Wide Web (WWW) can be considered a very large document collection in which hyperlink references establish a connection between individual documents or web pages. The content of web pages is crawled and indexed for efficient retrieval via keyword searches. This approach is sufficient for satisfying most users' information needs. It provides them with a ranked result list from which they can access and inspect several documents relevant to their query. In contrast, the Semantic Web initiative aims at creating a Web of Data which enables humans and machines alike to discover specific knowledge.

The W3C defined several standards to achieve the implementation of its vision of the Web of Data. As part of this project, RDF, SPARQL and the RDB2RDF standard R2RML will be used. These technologies are described in more detail in Appendix A: Semantic Web technologies.

2 Project Aim

2.1 Problem statement

As indicated in the previous section, the Semantic Web is the vision of linking data on the WWW and providing technologies for the automated processing of this Linked Data by computers (Berners-Lee, et al., 2001).

Heath and Bizer (2011) describe six different approaches for publishing Linked Data. As a first option, RDF triples can be stored in static files which are uploaded to, and made available via a web server. Alternatively, triples are inserted into a triple store and made available via interfaces for consumption on the web. For content that is provided via APIs or server-side scripts, the authors suggest implementing wrappers and scripts that return RDF triples based on the request sent to the API or server. A fifth option is annotating knowledge within the HTML content of web pages with RDFa syntax and thereby enabling software agents to extract triples automatically. The drawback of the last approach is that only information which is embedded in the content of web pages can be published this way. However, a lot of information is stored in the deep web, a term used to describe web content that is not indexed by search engines (Wright, 2009). One part of the deep web is web pages that are dynamically generated and populated with content from a database backend. Relational databases store information for other applications than web pages as well. Generally, they are utilised as a repository for structured and normalised data due to their efficient storage and retrieval capabilities. As a result, the Semantic Web research community acknowledged the requirement for generating RDF triples from the data already existing in relational databases (Perez de Laborda, et al., 2006; Lv, et al., 2010; Unbehauen, et al., 2012; Sequeda & Miranker, 2013). Consequently, the RDB2RDF working group developed Direct Mapping and R2RML as standard mapping languages for transforming relational data to RDF. However, the standard only specifies the two mapping languages and does not provide specific tools for translating database entities and their relationships into RDF triples or querying a relational database via SPARQL based on a Direct Mapping or R2RML mapping document.

2.2 Current and potential approaches

Prior to the publication of the Direct Mapping and R2RML standards as languages, three main approaches for dealing with relational data in the context of the Semantic Web existed. Lv et al. (2006) describe them as RDF dumps, triple stores and wrappers.

The first option refers to exporting relational data to static RDF files. Triples are generated based on a mapping from relational entities and relationships to RDF concepts and properties. This approach bears the advantage that the resulting RDF files can be queried directly with SPARQL and hence query translation is not required. However, various researchers developed their own mapping syntaxes and methodologies (Bizer, 2003; Perez de Laborda & Conrad, 2006) preventing interoperability and data exchange. Another disadvantage of this approach is the duplication of data which requires additional storage as well as regular updates from the database to the RDF files in order to keep the two data sources synchronised.

Triple stores provide an alternative solution by storing RDF data in a database schema rather than in plain files. The idea behind this approach is to take advantage of existing RDBMS technology such as transaction management, data integrity and performance (Chebotko, et al., 2009). Nonetheless, it is still necessary to maintain two data sets, the relational master data source and the translated RDF version. Storing RDF data in relational tables also adds the complexity of translating SPARQL queries to SQL queries and thereby introduces additional processing overhead.

Unlike the previous two approaches, wrapper tools do not materialise RDF triples, but provide virtual RDF views that can be queried via SPARQL. In this case additional storage and data synchronisation are not required. However, the issue of adopting proprietary mapping languages for generating the virtual RDF views remains. Furthermore, SPARQL queries need to be translated into SQL queries based on the corresponding mapping since the RDF representation is not materialised.

In summary, the main disadvantage of the presented options was the lack of a standard mapping language. This shortcoming has been addressed by the specification of Direct Mapping and R2RML and their subsequent recommendation by the W3C.

In view of the new mapping standards, all of the above approaches are still valid options for creating and publishing RDF serialisations of relational data, and subsequently querying the RDF data with SPARQL. The difference for the first two options, RDF dumps and triple stores, is that Direct Mapping or R2RML is used instead of a non-standard mapping syntax. Nevertheless, both remain unfavourable due to the duplication of data and the resulting need for additional storage and synchronisation between the two data representations. In most cases it can be assumed that the relational database continues to be the master data source that is used by an application for creating and modifying data. It is therefore desirable to expose an RDF representation of this data dynamically rather than refreshing a separate materialised data store. By allowing SPARQL queries to be executed against the virtual RDF views obtained via the R2RML mappings the latest real-time information is taken into account for answering a query. The only disadvantage concerning the wrapper solution is the additional processing overhead that is required for the SPARQL to SQL query translation. However, it can be argued that this issue is somewhat alleviated by the superior query performance of wrappers compared to RDF triple stores on large datasets (Bizer & Schultz, 2011).

Current implementations of wrappers or rewriters that support R2RML mappings are DB2Triples, Morph, RDF-RDB2RDF, SparqlMap, Ultrawrap, Virtuoso and XSparql (Hausenblas, 2012; Unbehauen, et al., 2012).

2.3 Proposed solution

The aim of the project is the development of a software application that enables users to execute SPARQL queries against a relational database by providing a combination of three inputs: Connection details to a RDBMS; a valid SPARQL query; a complete R2RML mapping document.

It is envisaged that the tool will not materialize RDF triples, but translate the SPARQL query to an equivalent SQL query against the database schema based on the R2RML mappings. This approach allows for querying real-time data as highlighted in the previous section and avoids data redundancy by not materialising RDF triples.

The application is intended to provide a graphical user interface as well as a command line tool to enable users to submit a completed R2RML mapping file, the database connection details and the SPARQL query. After processing the inputs, the application returns the query result in an applicable RDF format (RDF/XML, Turtle, N-Triples). Alternatively the program raises an error in case one of the three inputs did not match the correct format.

For the purpose of this project, a R2RML mapping document will be created manually. The mapping will be based on the IMDB[1] database. It is intended that existing ontologies like FOAF[2] and MO[3] are used for establishing the mapping.

3 Project Objectives

3.1 Architecture

The application is intended to be developed as a Java client application that connects to a PostgreSQL database via JDBC. Users are able to submit a R2RML mapping document via a file browse dialog, enter the database connection details in a form and enter the SPARQL query in a multiline text field. A button allows users to run their query once all inputs have been provided.

As shown in Figure 3.1 Application Architecture the application consists of three main components: A R2RML parser, a SPARQL parser and the query translation component. The former two provide inputs for the latter component. The query translation component is the implementation of an algorithm for translating a SPARQL query to a SQL query. It is also issuing the SQL query to the relational database and retrieving the returned result. The result set is then passed to the R2RML processor which transforms the relations to triples based on the R2RML mapping.

In a final step, the application prompts the user to specify a storage location for the generated RDF results file or displays them via the user interface.

[1] http://www.imdb.com/interfaces
[2] http://www.foaf-project.org/
[3] http://www.movieontology.org/

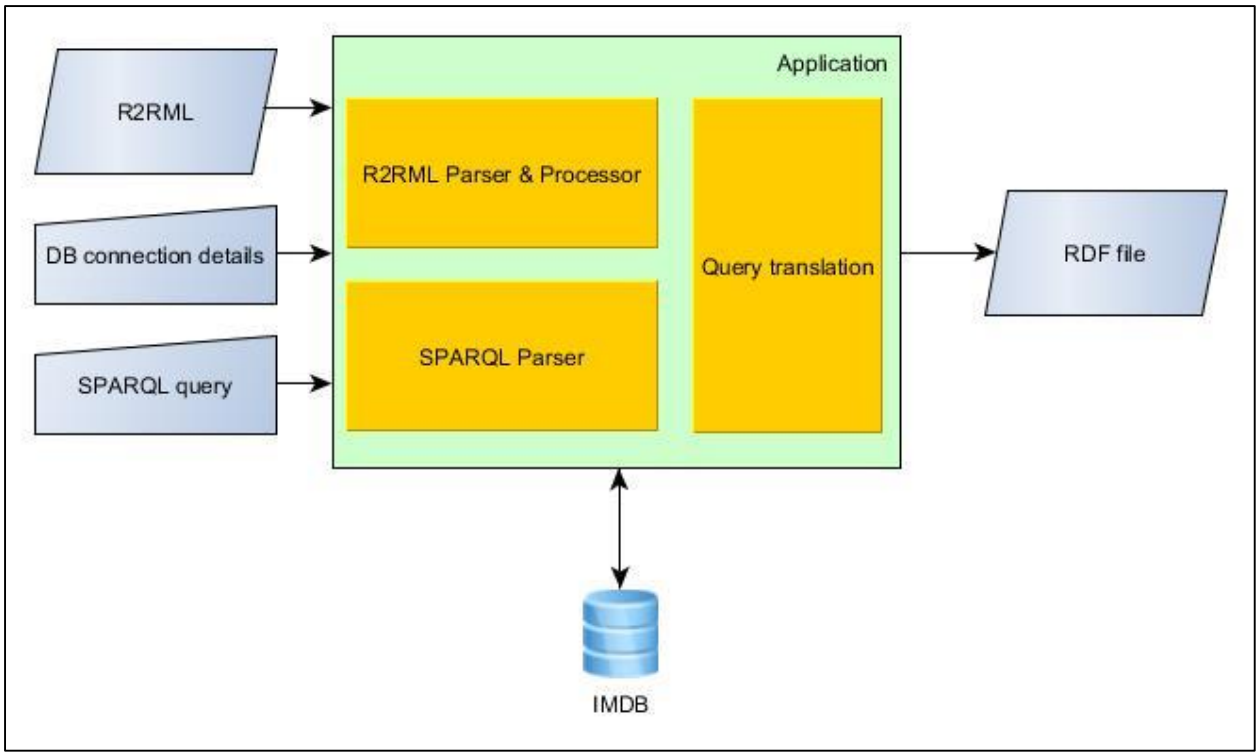

Figure 3.1 Application Architecture

3.2 Methodology

The development methodology chosen for this project is the spiral model. This approach allows the iterative development of prototypes and a reassessment of project risks at each iteration (Boehm, 1988).

It is planned that individual application components are developed and tested independently before continuing with another component. Furthermore, an investigation will be undertaken whether the two parser components are available as open source software and can be incorporated into the application without further custom development.

The incremental development cycles guarantee that risks with regards to milestones and deadlines are noticed quickly.

3.3 Technologies

The application will be developed as a Java client or command line application. The database backend used for storing the IMDB database is PostgreSQL 9.4. The data is loaded from plain text files provided on the IMDB website[4]. For the actual data import into the database schema a freely available Python package called IMDbPy[5] is used.

3.4 Project Plan

The project has seven main deliverables:

1. The manual creation of a R2RML mapping document

[4] http://www.imdb.com/interfaces
[5] http://imdbpy.sourceforge.net/

2. Determining a set of SPARQL test queries for validation and testing purposes
3. Identifying or developing the R2RML parser component
4. Identifying or developing the SPARQL parser component
5. Developing the query translation algorithm
6. Implementing the query translation component
7. Developing the graphical user interface

In order to complete the R2RML mapping document, the number of IMDB tables taken into consideration might be reduced as there are several dozen table objects in the IMDB schema. A second subtask for the deliverable is choosing suitable ontologies in order to customise the mapping.

As part of the second deliverable a fixed number of SPARQL queries will be documented. These serve as test cases for the query translation component and should be representative of the expressiveness provided by SPARQL. However, it might be necessary to limit the scope with regards to constructs supported by the query translation component. For instance, a decision might be made to limit the scope to SELECT queries only.

The third and fourth deliverable are either developed or, if development proves too time consuming, are obtained from existing open source projects.

The query translation algorithm and its implementation are developed iteratively. It is intended to start with limited scope for building a first prototype and then expanding functionality to more complex queries and mappings.

The last deliverable will be the graphical user interface as it is the least important aspect of the project.

The main deliverables and their timelines as well as the associated milestones are summarised in Table 3.1 Project deliverables.

Deliverable	Effort estimate	Schedule	Identified risk	Risk mitigation strategy
R2RML mapping file	3 days	09/06 to 11/06	None	n/a
Example SPARQL queries	2 days	12/06 to 13/06	None	n/a
Milestone 1				
R2RML parser	15 days	14/06 to 28/06	Exceeding allocated development time threatens project timeline	Research and incorporate open source component
SPARQL parser	15 days	29/06 to 13/07	Exceeding allocated development time threatens project timeline	Research and incorporate open source component
Milestone 2				
Query translation algorithm	25 days	14/07 to 07/08		

Deliverable	Effort estimate	Schedule	Identified risk	Risk mitigation strategy
Query translation component	15 days	08/08 to 22/08		
Milestone 3				
GUI development	15 days	23/08 to 06/09	Previous deliverables might require more time than allocated	Limit application to command line use

Table 3.1 Project deliverables

References

Berners-Lee, T., Hendler, J. & Lassila, O., 2001. The Semantic Web. *Scientific American*, May, pp. 29-37.

Bizer, C., 2003. *D2R MAP – A Database to RDF Mapping Language.* Budapest, Hungary, Proceedings of the Twelfth International World Wide Web Conference - Posters, WWW 2003.

Bizer, C. & Schultz, A., 2011. The Berlin SPARQL Benchmark. In: A. Sheth, ed. *Semantic Services, Interoperability and Web Applications: Emerging Concepts.* Hershey, PA: IGI Global, pp. 81-103.

Boehm, B., 1988. A Spiral Model of Software Development and Enhancement. *IEEE Computer,* 21(5), pp. 61-72.

Chebotko, A., Lu, S. & Fotouhi, F., 2009. Semantics preserving SPARQL-to-SQL translation. *Data & Knowledge Engineering,* 68(10), pp. 973-1000.

Cyganiak, R., Wood, D. & Lanthaler, M., 2014. *RDF 1.1 Concepts and Abstract Syntax.* [Online]
Available at: http://www.w3.org/TR/rdf11-concepts/
[Accessed 15 March 2015].

Das, S., Sundara, S. & Cyganiak, R., 2012. *R2RML: RDB to RDF Mapping Language.* [Online]
Available at: http://www.w3.org/TR/r2rml/
[Accessed 15 March 2015].

Hausenblas, M., 2012. *Implementations - RDB2RDF.* [Online]
Available at: http://www.w3.org/2001/sw/rdb2rdf/wiki/Implementations
[Accessed 12 April 2015].

Heath, T. & Bizer, C., 2011. Linked Data: Evolving the Web into a Global Data Space (1st edition). In: *Synthesis Lecture on the Semantic Web: Theory and Technology.* s.l.:Morgan & Claypool, pp. 1-136.

Lv, L., Jiang, H. & Ju, L., 2010. *Research and Implementation of the SPARQL-TO-SQL Query Translation Based on Restrict RDF View.* Sanya, IEEE.

Perez de Laborda, C. & Conrad, S., 2006. *Bringing Relational Data into the SemanticWeb using SPARQL and Relational.OWL.* Atlanta, GA, USA, IEEE.

Perez de Laborda, C., Matthäus, Z. & Stefan, C., 2006. *RDQuery - Querying Relational Databases on-the-fly with RDF-QL.* Podebrady, Czech Republic, Posters and Demos of the 15th International Conference on Knowledge Engineering and Knowledge Mangement, EKAW.

Priyatna, F., Corcho, O. & Sequeda, J., 2014. *Formalisation and Experiences of R2RML-based SPARQL to SQL query translation using Morph.* Seoul, Korea, Proceedings of the 23rd international conference on World wide web, Pages 479-490.

Sequeda, J. F. & Miranker, D. P., 2013. Ultrawrap: SPARQL Execution on Relational Data. *Web Semantics: Science, Services and Agents on the World Wide Web,* Volume 22, pp. 19-39.

Unbehauen, J., Stadler, C. & Auer, S., 2012. Accessing Relational Data on the Web with SparqlMap. In: H. Takeda, Y. Qu, R. Mizoguchi & Y. Kitamura, eds. *Semantic Technology.* Nara, Japan: Springer Berlin Heidelberg, pp. 65-80.

Unbehauen, J., Stadler, C. & Auer, S., 2013. *Optimizing SPARQL-to-SQL Rewriting.* Vienna, Austria, Proceedings of International Conference on Information Integration and Web-based Applications & Services, IIWAS 13.

W3C SPARQL Working group, 2013. *SPARQL 1.1 Overview.* [Online]
Available at: http://www.w3.org/TR/sparql11-overview/
[Accessed 15 March 2015].

Wright, A., 2009. *Exploring a 'Deep Web' That Google Can't Grasp.* [Online]
Available at:
http://www.nytimes.com/2009/02/23/technology/internet/23search.html?pagewanted=1&_r=0&th&emc=th
[Accessed 8 March 2015].

Appendix A: Semantic Web technologies

Resource Description Framework (RDF)

RDF is a framework for the purpose of modelling knowledge in the form of labelled graphs. Graphs consist of RDF triples, each of which is made up of a subject, predicate and object. Triples either express a relationship between resources or describe a resource itself (Cyganiak, et al., 2014).

SPARQL

SPARQL is a query language for RDF triples, whether they are stored in an RDF triple store or as graphs on the web. Its syntax appears to be similar to the Structured Query Language (SQL), however, the two languages are different. While SQL is set based, SPARQL follows bag based semantics. Results are not returned as tuples but as RDF terms or triples matching the graph pattern provided in the WHERE clause of a SPARQL query (W3C SPARQL Working group, 2013).

R2RML

The development of the WWW and its wide spread use led to three tier application architectures consisting of thin clients, an application server and a database server. As a result, a lot of information is stored in relational databases and is usually only accessible via forms or other query interfaces. A W3C working group published R2RML as a mapping syntax for expressing a relational schema as a set of graphs (Das, et al., 2012). This is achieved by translating relational records into RDF triples that are stored in a triple store or virtually materialised for further processing.