Extending R2RML to a source-independent mapping language for RDF

Anastasia Dimou, Miel Vander Sande, Pieter Colpaert, Erik Mannens, and Rik Van de Walle

Ghent University - iMinds - Multimedia Lab
Gaston Crommenlaan 8, bus 201, B-9050 Ledeberg-Ghent, Belgium
firstname.lastname@ugent.be

Abstract. Although reaching the fifth star of the Open Data deployment scheme demands the data to be represented in RDF and linked, a generic and standard mapping procedure to deploy raw data in RDF was not established so far. Only the R2RML mapping language was standardized but its applicability is limited to mappings from relational databases to RDF. We propose the extension of R2RML to also support mappings of data sources in other structured formats (indicatively CSV, TSV, XML, JSON). Broadening further its scope, the focus is put on the mappings and their optimal reuse. The language becomes source-agnostic, and resources are integrated and interlinked at a primary stage.

1 Introduction

Today, the idea of the (Linked) Open Data is widely spread and adopted. However, while reaching the fourth star of the Open Data deployment scheme[1] is easily attainable, achieving the fifth demands a well-considered approach and significantly greater effort. Current solutions are either highly customized to each case's specific needs or they follow a schematic and/or syntactic mapping approach. This fails to fully depict the semantics as it remains tied to the source file's structure. To this end, only R2RML[2] became a W3C recommendation aiming to formalize the mappings from relational databases to RDF (RDB2RDF). In practice though, one publishes data available in different source formats which, in turn, requires a more generic approach.

A generic language that maps the data independently of the source structure (*schema-agnostic*) and puts the focus on the mappings is a prominent advancement. Thereby, one deals with all different source files in a uniform way; in contrast with other languages that handle the mappings of different source formats separately. Therefore, the initial learning costs remain limited and the potential for the custom-defined mapping's reuse augments. As a result, the per-file mapping model followed so far gets surpassed, leading to contingent data integration and interlinking at a primary stage. In this paper, we propose an extension of the R2RML aiming to broaden its scope to cover also mappings from different structured data formats –CSV, TSV, XML and JSON files– to RDF.

[1] http://5stardata.info
[2] http://www.w3.org/TR/r2rml

2 State of the art

Beyond R2RML which has already several implementations[3], other RDB2RDF mapping languages were defined [1]. In the same context, there are corresponding languages to support CSV-to-RDF mappings (CSV2RDF), e.g., the XLWrap's mapping language [2], the Mapping Master's M2 [3] and Vertere[4]. On the other hand, in the case of mappings from XML to RDF (XML2RDF), the different tools rely mostly on existing XML solutions. To be more precise, XSLT-based approaches were explored, as the Krextor [4] and the AstroGrid-D[5] mapping tools, while other implementations deploy mappings using XPath and XQuery, e.g., the Tripliser[6] and the XSPARQL [5]. These solutions for XML sources lead to mappings on the syntactic level rather than on the semantic level or fail to provide a solution applicable to a broader domain. Beyond the standard Extract-Map-Load (EML) mappings, dynamic query translation was also explored, e,g, in the case of Tarql[7] (CSV2RDF) and XSPARQL (mapping and integration of XML, RDB and RDF resources).

In general, most tools deploy mappings from a certain source format to RDF (*source-centric approaches*). There are only a few tools that provide mappings from various source formats to RDF –DataLift [6], the DataTank [7], Karma [8], Open Refine[8] and Virtuoso Sponger[9] are the most well known– but only the DataTank uses a mapping language. For the latter's needs, Vertere was extended not only to cover CSV2RDF mappings but mappings from other structured data sources as well, namely databases, XML and JSON. Since R2RML became a W3C standard and due to its analogous nature to Vertere, the extension of R2RML is considered a prominent solution and its applicability verified.

3 Extending R2RML for a more generic use

An extension of the R2RML language is proposed, aiming to broaden its scope beyond RDB2RDF mappings, to cover every structured data format (a *Global-As-View approach*), and to address the limitations of existing languages. The R2RML's RDF graphs are used to express mappings independently of the source format. Therefore, the same custom mappings are reused whether the source files are in the same format or not, only by redetermining the references to the source values to be mapped, as the expected custom mapping definitions remain the same. The vocabulary extending the R2RML is available at `http://mmlab.be/users/andimou/rml.ttl`. The expansion is achieved as follows:

[3] http://www.w3.org/2001/sw/rdb2rdf/wiki/Implementations

[4] https://github.com/knudmoeller/Vertere-RDF

[5] http://www.gac-grid.de/project-products/Software/XML2RDF.html

[6] http://daverog.github.io/tripliser/

[7] https://github.com/cygri/tarql

[8] http://openrefine.org/

[9] http://virtuoso.openlinksw.com/dataspace/doc/dav/wiki/Main/VirtSponger

Extending RDF triples mapping. *Triples map* is extended not only to map each row in the logical table, but each resource in the logical source. To this end, the `rr:logicalTable` and `rr:tableName` become a sub-property of `rml:logicalSource` and `rml:sourceName` respectively, while `rr:elementName` for XML sources and `rr:objectName` for JSON sources are introduced. In the example, *books* is a logical table's, a JSON object's or an XML element's name.

```
<#RDB_CSV_map> rml:logicalSource [ rr:tableName "BOOKS" ];
  rr:subjectMap [ rr:template "http://data.example.com/books/{ISBN}" ];
  rr:predicateObjectMap [ rr:predicate ex:id; rr:objectMap [ rml:resource "ID" ] ].
<#XML_map> rml:logicalSource [ rml:elementName "/books" ];
  rr:subjectMap [ rr:template "http://data.example.com/books/{book/ISBN}"];
  rr:predicateObjectMap [ rr:predicate ex:id; rr:objectMap [ rml:resource "book/ISBN@id" ] ].
<#JSON_map> rml:logicalSource [ rml:objectName "books" ];
  rr:subjectMap [ rr:template "http://data.example.com/books/{book.ISBN}"];
  rr:predicateObjectMap [ rr:predicate ex:id; rr:objectMap [ rml:resource "book.id" ] ].
```

Extending resources' mapping. In the same context, *term maps* are extended to generate RDF terms from any logical resource, either this is a table row, an XML element or a JSON object. The *column-valued term map* is extended to cover every *resource term map*. Therefore, the R2RML's `rr:column` property becomes a sub-property of the `rml:resource` which is a valid column name for relational databases and CSV files, a valid XPath expression for an XML node's or attribute's absolute path and a valid path pattern in JavaScript syntax for objects in JSON source files, as in the aforementioned example.

Multiple entities per row. Most of the mapping languages (including R2RML) follow the *entity-per-row* model and consider that each row's RDF triples are mapped to the same subject. In its extended version, R2RML can map sets of columns to different subjects, which are then related among each other with a *predicate-object triples map*. For example, a row may have several columns with information about an event and a few of them refer to its location, e.g., latitude and longitude. Using this single row a *triples map* may be defined for the event while another *triples map* may be defined for the location where the event takes place (this mapping definition might be reused for other locations' mapping) and the two of them are related with a *predicate-object triples map*, as in the following example:

```
<#Event_map> rml:logicalSource [ rml:elementName "/events" ];
  rr:subjectMap [ rr:template "http://data.example.com/events/{event/id}" ];
  rr:predicateObjectMap
    [ rr:predicate ex:location; rr:objectMap [ rr:parentTriplesMap <#Location_map> ] ] ,
    [ rr:predicate ex:transport; rr:objectMap [ rr:parentTriplesMap <#Transport_map> ;
      rr:joinCondition [ rr:child "event/bus/num"; rr:parent "BUS_NUM" ] ] ].
<#Location_map> rml:logicalSource [ rml:elementName "/events" ];
  rr:subjectMap [
  rr:template "http://data.example.com/location/{event/location/lat},{event/location/long}"].
<#Transport_map> rml:logicalSource [ rr:tableName "TRANSPORTATIONS" ];
  rr:subjectMap [ rr:template "http://data.example.com/transport/{TYPE}/{BUS_NUM}"];
  rr:predicateObjectMap [ rr:predicate ex:name; rr:objectMap [ rml:resource "BUS_NAME" ] ].
```

Extended the logical sources. According to the `rr:sqlQuery`, the `rml:xmlQuery` is adapted and both are sub-properties of `rml:query` to serve a query against a source file. In the same context the `rml:queryLanguage` is defined to determine which language is used (indicatively, a W3C standard in the case of XML).

Integrated mapping. Extending the *reference object map*, one can use the subjects of another *triples map* as the objects generated by a *predicate-object map*. Since the *triples maps* may be based on different logical sources, the potential to create triples based on integrated sources emerges. At the aforementioned event example, an element node may refer to the number of the bus going to the event location, but the bus names are associated to the bus numbers at a separate table which is mapped by another *triples map*. The mappings of both of them are defined and a *predicate-object terms map* may be used to relate them.

4 Conclusions and Future Work

A generic mapping language is proposed to handle the mappings from different source formats to RDF. The uppermost goal of such an extension is to keep the focus on the mappings to be expressed rather than on the data and their original structure. With this work, we bring into discussion its feasibility, possible barriers and aspects that should be taken into consideration. In the future the arising generic mapping language will be used at the DataTank, instead of Vertere, to cover mappings from different source formats to RDF and, in the same time, to confront with the standard mapping language for the RDB2RDF mappings.

References

1. Hert, M., Reif, G., Gall, H.C.: A comparison of RDB-to-RDF mapping languages. In: Proceedings of the 7th International Conference on Semantic Systems. I-Semantics '11, New York, NY, USA, ACM (2011) 25–32
2. Langegger, A., Wöß, W.: XLWrap – Querying and Integrating Arbitrary Spreadsheets with SPARQL. In: Proceedings of the 8th International Semantic Web Conference. ISWC '09, Berlin, Heidelberg, Springer-Verlag (2009) 359–374
3. O'Connor, M.J., Halaschek-Wiener, C., Musen, M.A.: Mapping Master: a flexible approach for mapping spreadsheets to OWL. In: Proceedings of the 9th International Semantic Web Conference on The Semantic Web - Volume Part II. ISWC'10, Berlin, Heidelberg, Springer-Verlag (2010) 194–208
4. Lange, C.: Krextor - an extensible framework for contributing content math to the Web of Data. In: Proceedings of the 18th Calculemus and 10th international conference on Intelligent computer mathematics. MKM'11, Berlin, Heidelberg, Springer-Verlag (2011) 304–306
5. Bischof, S., Decker, S., Krennwallner, T., Lopes, N., Polleres, A.: Mapping between rdf and xml with xsparql. Journal on Data Semantics **1**(3) (2012) 147–185
6. Scharffe, F., Atemezing, G., Troncy, R., Gandon, F., Villata, S., Bucher, B., Hamdi, F., Bihanic, L., Képéklian, G., Cotton, F., Euzenat, J., Fan, Z., Vandenbussche, P.Y., Vatant, B.: Enabling Linked Data publication with the Datalift platform. In: Proc. AAAI workshop on semantic cities, Toronto, Canada (2012)
7. Vander Sande, M., Colpaert, P., Van Deursen, D., Mannens, E., Van de Walle, R.: The DataTank: an open data adapter with semantic output. In: 21st International Conference on World Wide Web, Proceedings. (2012)
8. Gupta, S., Szekely, P., Knoblock, C., Goel, A., Taheriyan, M., Muslea, M.: Karma: A system for mapping structured sources into the Semantic Web. In: 9th Extended Semantic Web Conference (ESWC2012). (May 2012)