RML-star: A Declarative Mapping Language for RDF-star Generation

Thomas Delva[1], Julián Arenas-Guerrero[2], Ana Iglesias-Molina[2], Oscar Corcho[2], David Chaves-Fraga[2], and Anastasia Dimou[1]

[1] IDLab, Department of Electronics and Information Systems, Ghent University - imec, Belgium
{thomas.delva,anastasia.dimou}@ugent.be
[2] Ontology Engineering Group, Universidad Politécnica de Madrid, Spain
{julian.arenas.guerrero,ana.iglesiasm,oscar.corcho,david.chaves}@upm.es

Abstract. RDF-star was recently proposed as a convenient representation to annotate statements in RDF with metadata by introducing the so-called RDF-star triples, bridging the gap between RDF and property graphs. However, even though there are many solutions to generate RDF graphs, there is no systematic approach so far to generate RDF-star graphs from heterogeneous data sources. In this paper, we propose *RML-star*, an extension of the RML mapping language to generate RDF-star. We introduce the extension of the RML ontology and the associated specification with representative examples.
URL: `https://w3id.org/kg-construct/rml-star`

Keywords: RML · R2RML · RDF-star · Knowledge Graphs.

1 Introduction

RDF-star was proposed as a compact representation to annotate statements in RDF with metadata [4]. For instance, the following declares that Bob claims Alice was born in 1996: `:bob :claims <<:alice :birthYear 1996>>.` Following the uptake of the proposed solution, a W3C Community Group was formed[3] and a W3C Draft Report [5] was recently released with improvements over the original proposal. By now, several RDF-related programming libraries, e.g., Eclipse RDF4J, Apache Jena, RDF.rb, and N3.js, and RDF graph database systems, e.g., Blazegraph, AnzoGraph, Stardog and GraphDB, have adopted RDF-star[4].

However, no mapping language supports the generation of RDF-star graphs so far. Most data are still heterogeneous, represented in different formats (e.g., relational databases, CSV, JSON, or XML). One of the most common approaches nowadays to integrate them into RDF graphs is the use of declarative mapping

[3] `https://www.w3.org/community/rdf-dev/`
[4] `https://blog.liu.se/olafhartig/`

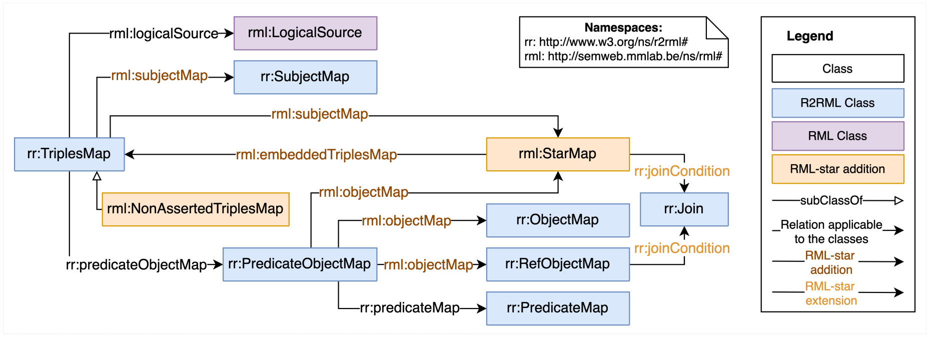

Fig. 1: The RML-star extension (Chowlk notation [3]). Orange classes and dark orange object properties show the additions to the RML ontology, light orange object properties represent extensions (i.e., change in domain and/or range).

languages such as R2RML [1] and RML [2]. R2RML is the W3C Recommendation mapping language to generate RDF graphs from relational databases. RML is a superset of R2RML that generates RDF graphs from data formats beyond relational databases, such as CSV, JSON, or XML. Extending a mapping language to specify how RDF-star datasets can be generated from heterogeneous data sources can potentially increase the amount of available RDF-star datasets and, thus, foster the adoption of the RDF-star proposal.

In this paper, we propose *RML-star*, an extension of RML to generate RDF-star graphs from heterogeneous data sources. We introduce a set of new classes and properties that allow describing how RDF-star datasets can be created from heterogeneous data sources in a systematic manner, using the same mapping language as to generate RDF datasets. We also introduce the RML-star specification that explains in detail how these extensions should be used and implemented.

2 RML-star

The aim of RML-star is to generate RDF-star triples by applying a set of additions and extensions over RML. The changes over the RML vocabulary include two new classes and three object properties, and the modification of one object property (Figure 1). The specification of RML-star with the corresponding ontology is available online at `https://w3id.org/kg-construct/rml-star`.

Throughout this section, we rely on an example to demonstrate RML-star. We consider two data sources: the CSV file in Listing 1 and the JSON file in Listing 2. The RML-star mapping for these data sources is given in Listings 3 and 4. Finally, in Listing 5 we show the generated RDF-star graph.

RML. Before we explain the RML-star extensions, we summarize how an RML mapping is defined. RML consist of a set of *Triples Maps* which include a *Logical Source* (lines 2 and 11 of Listing 3) to access the data sources, a *Subject Map* to generate the subjects of the triples (lines 3-4 and 12-13 of Listing 3), and multiple *Predicate-Object Maps* to generate the predicates and objects (lines 5-9 and 14-17 of Listing 3). *Predicate-Object Maps* are in turn composed of *Predicate Maps* and *(Referencing) Object Maps* (lines 6 and 7-9 of Listing 3 respectively). A *Referencing Object Map* uses the *Subject Map* of another *Triples Map* to generate the objects. Since a *Referencing Object Map* may involve two different data sources, join conditions can be specified.

```
PERSON , BIRTHYEAR, CLAIMER, CONFIDENCE
alice  , 1996     , bob    , 0.9
charlie, 2002     , daniel , 0.3
```

Listing 1: Contents of the logical source `:birthyears` (CSV).

```
[ { "PATIENT":  "alice",
    "HOSPITAL": "Juan Ramon Jimenez" },
  { "PATIENT":  "charlie",
    "HOSPITAL": "AZ Maria-Middelares" } ]
```

Listing 2: Contents of the logical source `:hospitalrecords` (JSON).

```
1  :innerTM a rml:NonAssertedTriplesMap ;
2    rml:logicalSource :birthyears ;
3    rml:subjectMap [
4      rr:template ":{PERSON}" ] ;
5    rr:predicateObjectMap [
6      rr:predicate :birthYear ;
7      rml:objectMap [
8        rml:reference "BIRTHYEAR" ;
9        rr:dataType xsd:integer ]] .

10 :outerTM a rr:TriplesMap ;
11   rml:logicalSource :birthyears ;
12   rml:subjectMap [
13     rr:template ":{CLAIMER}" ] ;
14   rr:predicateObjectMap [
15     rr:predicate :claims ;
16     rml:objectMap [
17       rml:embeddedTriplesMap :innerTM ]] .
```

Listing 3: Example of an RML-star mapping. It creates embedded triples that are not asserted.

```
:outerOuterTM a rr:TriplesMap ;                   1
  rml:logicalSource :birthyears ;                 2
  rml:subjectMap [                                3
    rml:embeddedTriplesMap :outerTM ] ;           4
  rr:predicateObjectMap [                         5
    rr:predicate :confidence ;                    6
    rml:objectMap [                               7
      rml:reference "CONFIDENCE" ;                8
      rr:dataType xsd:float ]] .                  9

:joiningTM a rr:TriplesMap ;                      10
  rml:logicalSource :hospitalrecords ;            11
  rml:subjectMap [                                12
    rml:embeddedTriplesMap :innerTM ;             13
    rr:joinCondition [                            14
      rr:child "PATIENT" ;                        15
      rr:parent "PERSON" ]] ;                     16
  rr:predicateObjectMap [                         17
    rr:predicate :recordedBy ;                    18
    rml:objectMap [                               19
      rml:reference "HOSPITAL" ]] .               20
```

Listing 4: Mapping extension of Listing 3, containing nested triples and multiple data sources.

```
:bob :claims << :alice :birthyear 1996 >> .
:daniel :claims << :charlie :birthyear 2002 >> .
<< :bob :claims << :alice :birthyear 1996 >> >> :confidence 0.9 .
<< :daniel :claims << :charlie :birthyear 2002 >> >> :confidence 0.3 .
<< :alice :birthyear 1996 >> :recordedBy "Juan Ramon Jimenez" .
<< :charlie :birthyear 2002 >> :recordedBy "AZ Maria Middelares" .
```

Listing 5: RDF-star triples generated by the RML-star mappings in Listings 3 and 4 from data sources in Listings 1 and 2.

Star Map. We introduce the *Star Map* class (`rml:StarMap`) to generate RDF-star triples. A Star Map can be either at the place of a Subject Map (lines 3-4 and 12-16 of Listing 4) or an Object Map (lines 16-17 of Listing 3), generating RDF-star triples in either the subject or object positions, following the RDF-star specification [5]. A Star Map can form either a subject or an object. For that reason, it belongs to the domain of `rml:subjectMap` and `rml:objectMap` properties. The original properties, `rr:subjectMap` and `rr:objectMap` had cardinality restrictions that prevent extending them to include Star Map in their domain. These additions are used exactly as the original ones in any other sense.

The object property `rml:embeddedTriplesMap` connects the Star Map to the Triples Map that defines how the RDF-star triples will be generated. A simple example of a Star Map is shown on lines 16-17 of Listing 3: it embeds triples generated by the Triples Map `:innerTM` in the objects of the triples generated by the Triples Map `:outerTM`. This results in the triples shown on lines 1-2 of Listing 5 when given Listing 1 as input.

Non-Asserted Triples Map. An *asserted RDF-star triple* is a triple that is an element of an RDF-star graph, as opposed to an *embedded RDF-star triple*, that only appears in the subject or object of another RDF-star triple. In RML-star, all generated triples are considered by default asserted RDF-star triples. To specify that a generated triple is embedded but not asserted, we introduce the *Non-Asserted Triples Map* (`rml:NonAssertedTriplesMap`) as a subclass of Triples Map (`rr:TriplesMap`). This Triples Map has the same expressiveness as every other Triples Map and just adds the information of being non-asserted. For instance, `:innerTM` (line 1 of Listing 3) is declared to be a Non-Asserted Triples Map and, as a result, the `:birthYear` triples it generates are not present in Listing 5 as asserted triples: they only occur as embedded triples.

This structure allows the recursion of Triples Maps to nest as many embedded triples as needed. For example, the Triples Map `:outerOuterTM` generates triples that have embedded triples generated by `:outerTM` as their subject, and `:outerTM` in turn generates triples with embedded triples from `:innerTM` as their object. As a result, `:outerOuterTM` generates triples containing two levels of embedded triples (lines 3-4 of Listing 5).

An *Embedded Triples Map* can generate triples using different data sources. Thus, the Star Map needs to have join conditions to combine such data sources. To achieve this, the property `rr:joinCondition` is extended to include Star Map in its domain. This property, in contrast to `rr:objectMap` and `rr:subjectMap`, is easily extended due to the lack of restrictions in the original vocabulary. On lines 12-16 of Listing 4 a Star Map is declared which joins the data sources in Listings 1 and 2 on equal values of the `PERSON` column and the `PATIENT` attribute. It creates the triples on lines 5-6 of Listing 5.

3 Conclusions and Next Steps

In this paper, we present RML-star, an extension of RML, which allows generating RDF-star graphs from heterogeneous data sources. We include a set of new classes and properties while maintaining the general structure of R2RML and RML. With this proposal, we aim at promoting the adoption of RDF-star and pave the way so other mapping languages provide similar extensions. RML-star is discussed within the Knowledge Graph Construction W3C Community Group[5]

[5] `https://w3id.org/kg-construct`

and will be part of the specifications' suite developed by the group. Thanks to this solution, we devise a promising future work line on the development of efficient and scalable systems to generate RDF-star graphs.

Acknowledgments

This research is financially supported by Ministerio de Ciencia e Innovación, Spain, under grant Knowledge Spaces (PID2020-118274RB-I00) and by an FPI grant (BES-2017-082511).

References

1. Das, S., Sundara, S., Cyganiak, R.: R2RML: RDB to RDF Mapping Language. W3C Recommendation, W3C (2012), http://www.w3.org/TR/r2rml/
2. Dimou, A., Vander Sande, M., Colpaert, P., Verborgh, R., Mannens, E., Van de Walle, R.: RML: A Generic Language for Integrated RDF Mappings of Heterogeneous Data. In: Proceedings of the 7th Workshop on Linked Data on the Web. CEUR Workshop Proceedings, vol. 1184 (2014)
3. Feria, S.C., García-Castro, R., Poveda-Villalón, M.: Converting UML-based ontology conceptualizations to OWL with Chowlk. In: The Semantic Web: ESWC 2021 Satellite Events. pp. 44–48. Springer International Publishing (2021)
4. Hartig, O.: Foundations of RDF* and SPARQL* (An Alternative Approach to Statement-Level Metadata in RDF). In: Proceedings of the 11th Alberto Mendelzon International Workshop on Foundations of Data Management and the Web. CEUR Workshop Proceedings, vol. 1912 (2017)
5. Hartig, O., Champin, P.A., Kellogg, G., Seaborne, A.: RDF-star and SPARQL-star. W3C Draft Community Group Report, W3C (2021), https://w3c.github.io/rdf-star/cg-spec/