RML 1.2: Aligning RML with RDF 1.2 Triple Terms and Reifying Triples

Julián Arenas-Guerrero[1,*], Sebastián Ferrada[2,3,4], Sebastián Vergara[2,3,4] and Olaf Hartig[5]

[1]Universidad Politécnica de Madrid, Madrid, Spain
[2]Department of Computer Science and Data and AI Initiative, Universidad de Chile, Santiago, Chile
[3]Millennium Institute for Foundational Research on Data, Santiago, Chile
[4]National Center for Artificial Intelligence Research, Santiago, Chile
[5]Linköping University, Linköping, Sweden

Abstract

RDF 1.2 introduces triple terms and the rdf:reifies property as a native mechanism for statement-level metadata. However, declarative mapping languages such as RML, which generate RDF from heterogeneous data sources, have not yet been aligned with this update: prior RDF-star extensions predate RDF 1.2's abstract data model and, furthermore, do not support its new directional language-tagged strings. This paper introduces RML 1.2, a backward-compatible extension of RML-Core for generating triple terms and directional language-tagged strings. We accompany the extension of RML with an open-source Python implementation and a suite of test cases.

1. Introduction

Since the introduction of RDF, making statements about statements (i.e., annotating triples with metadata) has relied on the reification vocabulary and other cumbersome alternatives [1]. This limitation, together with the desire for closer interoperability with labeled property graphs, motivated the creation of the W3C RDF-star Community Group in 2020 with the aim of consolidating the seminal RDF-star proposal [2]. This group then evolved into the RDF & SPARQL 1.2 Working Group in 2022, tasked with standardizing a native mechanism for adding statement-level metadata. The result of this effort is the recent W3C Candidate Recommendation of RDF 1.2 Concepts [3], which introduces triple terms (triples as first-class RDF terms, to be used as the object of other triples) together with the corresponding rdf:reifies property, and an extension of language-tagged string with a notion of base direction.

While the RDF & SPARQL 1.2 Working Group is updating a total of 20 W3C Recommendations, spanning the RDF and SPARQL families, other RDF-related languages such as the R2RML W3C Recommendation [4] and the RML specification [5] are not covered. These languages, which map heterogeneous types of data to RDF, were previously extended for RDF-star [1, 6]. However, since these earlier extensions predate the RDF 1.2 standardization of triple terms, reifying triples, and directional language-tagged strings, their constructs do not directly align with the current RDF 1.2 abstract data model.

This paper proposes RML 1.2, an extension of the redesigned RML-Core vocabulary [5] that aligns RML with version 1.2 of RDF. RML 1.2 introduces four features: rml:TripleTermMap for generating triple terms from existing triples maps, the rml:NonAssertedTriplesMap class for expressing when the generated triples are used solely as triple terms (and thus, not asserted), rml:reifyingMap as a shortcut for the rdf:reifies reification pattern, and rml:DirectionMap for directional language-tagged strings. All other RML concepts remain unchanged, keeping the extension backward-compatible with existing RML mappings. An open-source implementation is also presented, along with a suite of test cases to validate the proposed constructs.

DMKG'26: 2nd International Workshop on Data Management for Knowledge Graphs, October 2026, Bari, Italy
*Corresponding author.
julian.arenas.guerrero@upm.es (J. Arenas-Guerrero)
0000-0002-3029-6469 (J. Arenas-Guerrero); 0000-0002-9834-8376 (S. Ferrada); 0009-0005-3511-060X (S. Vergara); 0000-0002-1741-2090 (O. Hartig)

2. RML 1.2 Overview and Example

RML 1.2 extends RML-Core with a minimal set of constructs to (i) generate RDF 1.2 triple terms and reifying triples and (ii) create directional language-tagged strings. Existing RML concepts such as triples maps, logical sources, and term maps are unchanged. Specifically, RML 1.2 introduces the following:

- rml:TripleTermMap and rml:tripleTermMap to create triple terms from the triples produced by a referenced triples map.
- rml:NonAssertedTriplesMap to declare that the triples generated are used only to form triple terms and are not asserted as standalone RDF triples.
- rml:reifyingMap as a shortcut for mappings that produce reifying triples.
- rml:DirectionMap and rml:directionMap (and its shortcut rml:direction) for directional language-tagged strings, to be used together with rml:languageMap (or rml:language).

RML 1.2 Full processors must support these features and generate RDF 1.2 Full datasets with triple terms and rdf:reifies; RML 1.2 Basic processors generate only basic RDF terms.

2.1. Example Input Database and RDF Output

To illustrate the new features, we use a relational database with the following `scores` table with one row:

film	score	src
1	9.2	IMDB

We aim to produce the following RDF 1.2 triples from this database (written in Turtle 1.2):

```
VERSION "1.2"
PREFIX : <http://data.example.org/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

:film1 :score 9.2 .
_:r1 rdf:reifies <<( :film1 :score 9.2 )>> ;
    :source "IMDB" .
```

2.2. Triple-Term Map and Reifying Triple

To create the first triple, we can use RML-Core with a standard triples map:

```
PREFIX : <http://data.example.org/>
PREFIX rml: <http://w3id.org/rml/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

<#TMScore>
   a rml:TriplesMap ; ## Use rml:NonAssertedTriplesMap if the triple is not asserted
   rml:logicalSource <#ScoresTableLS> ;
   rml:subjectMap [
      rml:template "http://data.example.org/film{film}"
   ] ;
   rml:predicateObjectMap [
      rml:predicate :score ;
      rml:objectMap [
         rml:reference "score" ;
         rml:datatype xsd:decimal
      ]
   ] .
```

To produce the triple term and the reifying triple, we use a triple-term map, as introduced by RML 1.2:

```
<#TMReifScore>
   a rml:TriplesMap ;
   rml:logicalSource <#ScoresTableLS> ;
   rml:subjectMap [
      rml:template "r{film}" ;
      rml:termType rml:BlankNode
   ] ;
   rml:predicateObjectMap [
      rml:predicate rdf:reifies ;
      rml:objectMap [
         rml:tripleTermMap <#TMScore>
      ]
   ] ;
   ## Alternatively, use the shortcut: rml:reifyingMap <#TMScore> ;
   rml:predicateObjectMap [
      rml:predicate :source ;
      rml:objectMap [
         rml:reference "src"
      ]
   ] .
```

Here:

- <#TMScore> declares how to create the score triple as an asserted triple.
- The first object map appearing in <#TMReifScore> is a triple-term map that refers to <#TMScore> and, thus, generates the triple term with the score triple.
- <#TMReifScore> produces the reifying triple (rdf:reifies) and attaches metadata (:source).

To generate only the triple term (without making it an asserted triple), the mapping could type <#TMScore> as an rml:NonAssertedTriplesMap instead of an rml:TriplesMap.

Graph maps on the base triples map do not contribute to triple term generation. They only determine the graph(s) in which the standalone triples of the base triples map are asserted (if asserted).

Both rml:RefObjectMap and rml:TripleTermMap refer to another triples map to produce RDF terms. rml:RefObjectMap derives RDF terms solely from the subject map of the referenced triples map (which contains exactly one), whereas rml:TripleTermMap derives RDF terms also from the predicate-object maps within the referenced triples map, which may range from zero to many. A triple-term map pointing to a triples map with no predicate-object maps MUST NOT generate triple terms. Conversely, if the base triples map has multiple predicate-object maps, all of them contribute to producing triple terms.[1]

2.3. Directional Language-Tagged Strings

RML 1.2 allows object maps to generate literals with both a language tag and a base direction (e.g., English left-to-right), following RDF 1.2. To illustrate this feature, we extend the example table, which contains one row:

film	score	src	review
1	9.2	IMDB	Great performance!

An object map for directional language-tagged strings looks as follows:

[1]It must be noted that the expected use case is that triple-term maps reference triples maps with a single predicate-object map. Otherwise, a reifier could be linked to several triple terms (generated by the multiple predicate-object maps).

```
[] rml:predicateObjectMap [
   rml:predicate :review ;
   rml:objectMap [
      rml:reference "review" ;
      rml:language "en" ;
      rml:directionMap [
         rml:constant "ltr"
      ]
   ]
] .
```

This produces the RDF term "Great performance!"@en--ltr. Every object map with an rml:directionMap property MUST also have an rml:languageMap property. Valid direction values are "ltr" and "rtl".

3. Ontology Module

3.1. Creating Triple Terms and Reifying Triples

A *triple-term map* is an object map whose generated RDF term is an RDF 1.2 triple term. An object map is a triple-term map if it has an rml:tripleTermMap property; it may also be typed explicitly as rml:TripleTermMap. A triple-term map:

- MUST be used as an rml:objectMap value.
- MUST have exactly one rml:tripleTermMap property, pointing to a triples map whose generated triples define the subject, predicate, and object of the triple term, and which is called the *base triples map* of the triple-term map.
- MAY have one or more rml:joinCondition properties: the child maps are evaluated over the logical source of the triples map containing the triple-term map, and the parent maps over the logical source of the base triples map. If a triple-term map has no join condition, the base triples map's term maps are evaluated on the current logical iteration of the triples map containing it.[2] The logical source of the base triples map is not used in this case.

The directed graph defined by rml:tripleTermMap references across all triples maps in a mapping document MUST be acyclic. This acyclicity requirement permits nested triple terms, i.e., a base triples map whose predicate-object map contains another rml:TripleTermMap, as long as no cycle is formed.

The relationship between a triple-term map and its base triples map is analogous to that between a referencing object map (rml:RefObjectMap) and its parent triples map [5]. The following example illustrates a triple-term map with a join condition:

```
[] rml:objectMap [
   rml:tripleTermMap <#TMScore> ;
   rml:joinCondition [
      rml:child "film" ;
      rml:parent "film"
   ]
] .
```

As a shortcut, a triples map may have an rml:reifyingMap property referring to a base triples map, which is equivalent to adding an rdf:reifies predicate-object map with an rml:TripleTermMap having no join conditions referencing the indicated base triples map. Each occurrence of this property MUST be treated exactly as if the triples shown in the table below were present in the mapping graph instead:

[2] RML-Core requires the logical sources to be effectively equal when a referencing object map has no join condition [5]. We propose aligning referencing object maps with the behaviour defined here for triple-term maps.

Triple involving shortcut property	Replacement triples
`?tmReif rml:reifyingMap ?tmBase .`	`?tmReif rml:predicateObjectMap [ rml:predicate rdf:reifies ; rml:objectMap [ rml:tripleTermMap ?tmBase ] ] .`

3.2. Creating Directional Language-Tagged Strings

A *direction map* is a term map that generates the base direction component of a directional language-tagged string. It is specified on an rml:ObjectMap using the rml:directionMap property. Constraints:

- the generated direction value MUST be one of the tokens “ltr” or “rtl”.
- if rml:directionMap or rml:direction is present on an object map, the same object map MUST also carry a language specification via rml:languageMap or rml:language.

The generated RDF term is a literal with a language tag and a base direction, created as follows:

- The lexical value is produced by applying the object term map to the current row.
- The datatype IRI is rdf:dirLangString.
- The language tag is generated by rml:languageMap or rml:language.
- The base direction is generated by rml:directionMap or rml:direction.

rml:direction is a shortcut property equivalent to a constant-valued rml:directionMap. Each occurrence of this property MUST be treated exactly as if the triples shown in the table below were present in the mapping graph instead:

Triple involving shortcut property	Replacement triples
`?om rml:direction ?dir .`	`?om rml:directionMap [ rml:constant ?dir ] .`

3.3. Classes

rml:TriplesMap and rml:NonAssertedTriplesMap are disjoint classes. A mapping rule that satisfies the structural requirements of a triples map and is not explicitly classified as rml:NonAssertedTriplesMap MUST be processed as an rml:TriplesMap. Consequently, the triples generated by that map MUST be asserted in the output dataset. All RML predicates that have rml:TriplesMap as rdfs:domain or rdfs:range, now must have owl:unionOf (rml:TriplesMap rml:NonAssertedTriplesMap) as rdfs:domain or rdfs:range. A triples map typed as both rml:TriplesMap and rml:NonAssertedTriplesMap is invalid.

The class rml:TripleTermMap is a subclass of rml:ObjectMap. As such, it inherits the term map evaluation rules of RML-Core and may appear wherever an rml:ObjectMap is expected.

The class rml:DirectionMap is a subclass of rml:TermMap. It represents term maps whose generated value is the base direction of a directional language-tagged string. A direction map MUST generate exactly one of the values “ltr” or “rtl” and MUST be referenced by an rml:ObjectMap through rml:directionMap. If an object map contains an rml:directionMap or rml:direction property, it MUST also specify a language through rml:languageMap or rml:language.

Table 1 summarizes the conditions for these classes.

3.4. Properties

Table 2 describes the new properties introduced by RML 1.2.

- For rml:tripleTermMap, the directed graph of rml:tripleTermMap references across all triples maps in a document MUST form an acyclic graph. A document with a cycle in this graph is invalid.
- If an object map contains an rml:directionMap or rml:direction property, it MUST also contain an rml:languageMap or rml:language property. The generated direction value MUST be one of “ltr” or “rtl”.

Table 1
RML 1.2 classes

Class	Represents	Minimum conditions
rml:TripleTermMap	Triple-term maps, i.e., term maps whose generated RDF term is an RDF 1.2 triple term.	A resource that (i) is an rml:ObjectMap and (ii) has exactly one rml:tripleTermMap property whose value is an rml:TriplesMap or rml:NonAssertedTriplesMap.
rml:NonAssertedTriplesMap	Triples maps whose generated triples are used only as triple terms (e.g., as targets of rdf:reifies) and are not asserted as standalone triples.	A resource representing a mapping rule typed as rml:NonAssertedTriplesMap whose generated triples are used exclusively via rml:TripleTermMap and MUST NOT be inserted into the output dataset as ordinary RDF triples.
rml:DirectionMap	Direction maps, i.e., term maps whose generated value is the base direction of a directional language-tagged string.	A resource that is an rml:TermMap and generates exactly one of the direction tokens "ltr" or "rtl". It is referenced by an rml:ObjectMap through rml:directionMap.

Table 2
RML 1.2 properties

Property	Represents	Context	Card.
rml:tripleTermMap	The link from a triple-term map to the triples map whose triples are turned into RDF 1.2 triple terms.	On an rml:TripleTermMap, points to the rml:TriplesMap or rml:NonAssertedTriplesMap whose generated triples define the content of the triple term.	1
rml:reifyingMap	A shortcut for specifying reifying triples following the rdf:reifies pattern.	On an rml:TriplesMap or rml:NonAssertedTriplesMap `tmReif`, points to the base rml:TriplesMap or rml:NonAssertedTriplesMap `tmBase` whose triples are reified; semantically equivalent to adding an rdf:reifies predicate-object map with an rml:TripleTermMap referencing `tmBase`.	$0 \dots N$
rml:directionMap	The base direction component of a directional language-tagged string, expressed as a term map.	On an rml:ObjectMap that produces a literal, points to a term map whose values are direction tokens ("ltr", "rtl").	$0 \dots 1$
rml:direction	Shortcut for a constant-valued rml:directionMap.	On an rml:ObjectMap, indicates a fixed base direction value; equivalent to a term map with rml:constant via rml:directionMap.	$0 \dots 1$

rml:tripleTermMap has domain rml:TripleTermMap and range rml:TriplesMap or rml:NonAssertedTriplesMap: it links a triple-term object map to the triples map whose generated triples define the triple term's content. rml:reifyingMap has domain rml:TriplesMap or rml:NonAssertedTriplesMap and range rml:TriplesMap or rml:NonAssertedTriplesMap: it links a reifying triples map to its base triples map, and is semantically equivalent to an rdf:reifies predicate-object map with an rml:TripleTermMap referencing the base triples map. rml:directionMap has domain rml:ObjectMap and range rml:TermMap: it points to a term map whose value is a direction token; its shortcut rml:direction takes a plain literal value and has the same domain (MUST be one of "ltr" or "rtl"). When there is an rml:directionMap or rml:direction predicate on an object map, the same object map MUST also carry an rml:languageMap or rml:language language specification.

Additionally, the domain of rml:joinCondition (reused from RML-Core, hence not included in Table 2) is extended to also include rml:TripleTermMap. It now has the union of rml:RefObjectMap and rml:TripleTermMap) as its domain.

4. Implementation, Test Cases and Future Work

With the purpose of verifying and validating the RML 1.2 constructs, we implemented an open-source lightweight Python processor[3] and a collection of ten test cases. Each test case includes a CSV input source, an RML 1.2 mapping document, an expected N-Triples output, and a manifest describing the test case. The collection covers both positive behaviors (asserting/reifying triples, non-asserted triples maps as triple terms, nested triple-term maps, the rml:reifyingMap shortcut, directional literals) and validation rules (invalid direction tokens, missing language tags, cyclic rml:tripleTermMap references, and empty-target triple-term maps), confirming the constructs are implementable and testable. We are currently developing the formalization of these constructs and implementing them into Morph-KGC [1].

Acknowledgments

This publication is based upon work started within COST Action CA23147 GOBLIN - Global Network on Large-Scale, Cross-domain and Multilingual Open Knowledge Graphs, supported by COST (European Cooperation in Science and Technology, https://www.cost.eu).

Ferrada and Vergara were supported by ANID – Millennium Science Initiative Program – Code ICN17_002; by Fondecyt Grant 11251322; and by CENIA FB210017, Basal ANID.

Hartig was supported by the Swedish Research Council (Vetenskapsrådet, project 2025-06246).

Declaration on Generative AI

During the preparation of this work, the authors used Claude for grammar and spelling checks. OpenAI Codex, using GPT-5.5, was used to support the development and debugging of the Python-based RML 1.2 test suite. The authors reviewed, adapted, and validated all AI-assisted outputs and take full responsibility for the final content, code, and results.

References

[1] J. Arenas-Guerrero, A. Iglesias-Molina, D. Chaves-Fraga, D. Garijo, O. Corcho, A. Dimou, Declarative generation of RDF-star graphs from heterogeneous data, Semantic Web 16 (2025) 125–134. doi:10.3233/SW-243602.

[2] O. Hartig, 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, volume 1912, CEUR Workshop Proceedings, 2017. URL: http://ceur-ws.org/Vol-1912/paper12.pdf.

[3] G. Kellogg, O. Hartig, P.-A. Champin, A. Seaborne, RDF 1.2 Concepts and Abstract Data Model, W3C Candidate Recommendation, 2026. https://www.w3.org/TR/2026/CR-rdf12-concepts-20260407/.

[4] S. Das, S. Sundara, R. Cyganiak, R2RML: RDB to RDF Mapping Language, W3C Recommendation, 2012. http://www.w3.org/TR/r2rml/.

[5] A. Iglesias-Molina, D. Van Assche, J. Arenas-Guerrero, B. De Meester, C. Debruyne, S. Jozashoori, P. Maria, F. Michel, D. Chaves-Fraga, A. Dimou, The RML Ontology: A Community-Driven Modular Redesign After a Decade of Experience in Mapping Heterogeneous Data to RDF, in: Proceedings of the 22nd International Semantic Web Conference, Springer Nature Switzerland, 2023, pp. 152–175. doi:10.1007/978-3-031-47243-5_9.

[6] L. Sundqvist, Extending VKG Systems with RDF-star Support, 2022. URL: https://ontop-vkg.org/publications/2022-sundqvist-rdf-star-ontop-msc-thesis.pdf.

[3] https://github.com/Vergaraaa1/rml12-demo