

RMLEditor: A Graph-Based Mapping Editor for Linked Data Mappings

Pieter Heyvaert

Anastasia Dimou, Aron-Levi Herregodts, Ruben Verborgh
Dimitri Schuurman, Erik Mannens and Rik Van de Walle



How can I map
existing data to Linked Data?

There are GUI tools for that.

Can non-Semantic Web experts work with these tools?

Well, if they are willing to read the full specification of the underlying technologies...

Our RMLEditor brings the editing of mappings to non-Semantic Web experts.

Overview

mapping process to generate Linked Data

data owners vs. Semantic Web experts during this process

existing tools help the mapping process

RMLEditor

- features

- user validation

Overview

mapping process to generate Linked Data

data owners vs. Semantic Web experts during this process

existing tools help the mapping process

RMLEditor

- features

- user validation

Knowledge is required to create mappings

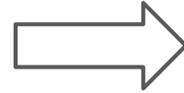
knowledge
about the data



knowledge about
the mapping language



knowledge
about the schemas



Linked Data

Need to know what the data is about

know how the data is related to each other

Need to know how the mapping language works

mapping languages offer a declarative way to generate Linked Data

based on existing data

set of rules

improvement over custom scripts/tools

examples

- R2RML

- RML

Need to know the schemas

vocabularies

ontologies

to annotate the data

Focus on data and mapping language

knowledge
about the data



knowledge about
the mapping language



knowledge
about the schemas



Linked Data

Overview

mapping process to generate Linked Data

data owners vs. Semantic Web experts during this process

existing tools help the mapping process

RMLEditor

- features

- user validation

Data owners know their data



Data owners don't know about mapping languages

data owners have limited background in the Semantic Web

need to learn the **approach** of the mapping language

need to learn the **syntax** of the mapping language

writing the mapping statements is **error-prone**

```
<#Triples_Mapping>
  rml:logicalSource <#DATA_EdgeInput> ;

  rr:subjectMap [
    rr:template "http://ex.com/{@node}" ;
    rr:class foaf:Person ] ;

  rr:predicateObjectMap [
    rr:predicate foaf:name;
    rr:objectMap [
      rr:parentTriplesMap <#LS_Mapping> ;
      rr:joinCondition [
        rr:child "@title";
        rr:parent "@id" ] ] ] ;
```

...

Data owners don't want
to use mapping languages



Semantic Web experts generate Linked Data

hired by the data owners

have knowledge of the mapping language

They don't have
knowledge about the data



Semantic Web experts



knowledge
about the data



knowledge about
the mapping language



Linked Data



data owners

Overview

mapping process to generate Linked Data

data owners vs. Semantic Web experts during this process

existing tools help the mapping process

RMLEditor

- features

- user validation

Tools to help the mapping process

knowledge
about the data



~~knowledge about
the mapping language~~



Linked Data

Tools to help the mapping process

knowledge
about the data



use mapping tool



Linked Data

GUIs are developed on mapping languages

no syntax errors

Better

but knowledge of the language is still required

subjectMap

parentTriplesMap

predicateObjectMap

Can you understand these terms without reading the full specification?

Tools require less knowledge of the language

knowledge
about the data



use mapping tool



Linked Data



Goal:

tools require no knowledge of the language

knowledge
about the data



use mapping tool



Linked Data



Overview

mapping process to generate Linked Data

data owners vs. Semantic Web experts during this process

existing tools help the mapping process

RMLEditor

- features

- user validation

Used by multiple projects and companies

including



Overview

mapping process to generate Linked Data

data owners vs. Semantic Web experts during this process

existing tools help the mapping process

RMLEditor

features

user validation

Three panels

The screenshot displays the RML EDITOR interface, which is divided into three main panels:

- Input Panel:** Shows the XML source file 'spots.xml' with a tree view of its structure. The tree includes 'spots.xml', 'spots', 'spot []', 'id', and 'title'. Below the tree is a text editor containing the following XML code:

```
<?xml version="1.0" encoding="UTF-8"?>
<spots>
  <spot>
    <id>6598</id>
    <title>Play Beach</title>
  </spot>
  <spot>
    <id>6599</id>
    <title>Paintball Gent</title>
  </spot>
</spots>
```
- Modeling Panel:** Displays a graph visualization of the RML. It features two nodes: an orange circle labeled '...oc/{id}' (representing 'spot') and a blue circle labeled 'ex:{_id}' (representing 'event:place'). Both nodes are connected to a central gray rectangle labeled 'event:place'. Below each node is a gray rectangle labeled 'dcterms:title', which is connected to an orange 'title' box (under the orange node) and a blue 'title' box (under the blue node). Navigation arrows are visible on the left side.
- Results Panel:** A table showing the results of the RML execution. The table has three columns: Subject, Predicate, and Object.

Subject	Predicate	Object
ex:6719	event:place	ex:loc/6795
ex:6719	dcterms:title	Spannende speurt
ex:6719	a	schema:Event
ex:8470	dcterms:title	Lichtfestival Gent
ex:8470	a	schema:Event
ex:8480	event:place	ex:loc/6691
ex:8480	dcterms:title	Vuurwerk
ex:8480	a	schema:Event
ex:8537	event:place	ex:loc/6601
ex:8537	dcterms:title	Festival de la lumie
ex:8537	a	schema:Event
ex:8547	dcterms:title	Feux d'artifice
ex:8547	a	schema:Event
ex:8594	event:place	ex:loc/6700
ex:8594	dcterms:title	Light Festival Gher
ex:8594	a	schema:Event

Independent of mapping language

graph visualization to hide it

uses RML under the hood

mapping language can be replaced



Map multiple data sources at the same time

data is spread across multiple data sources

Map data sources in different formats

databases

CSV files

XML files

JSON files

File Edit Mapping View

RML EDITOR

events.csv

Detail Lowest Low Moderate High Highest

events.csv

- _id
- title
- subtitle
- summary
- category_0_tid
- Spots_0_id

_id	title	subtitle	summary
6719	Spannende s	Moord met ee	Tweedelige
8470	Lichtfestival C	Een lichtparc	Een verlich
8480	Vuurwerk		Zorg dat u

- events.csv
- _id
- title
- subtitle
- summary
- category_0_tid
- Spots_0_id

spots.xml

_id	title	subtitle	summary
6719	Spannende s	Moord met ee	Tweedelige
8470	Lichtfestival C	Een lichtparc	Een verlich
8480	Vuurwerk		Zorg dat u

spots.xml

Detail Lowest Low Moderate High Highest

spots.xml

spots

spot []

id

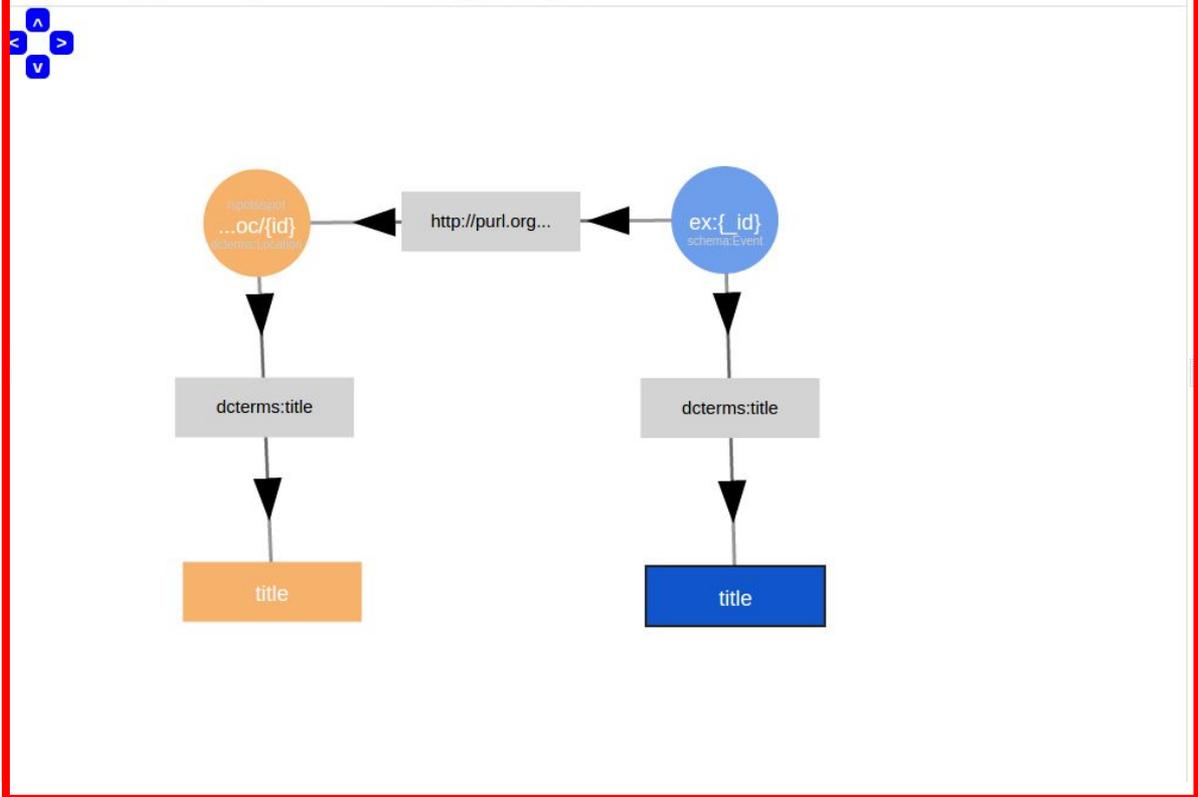
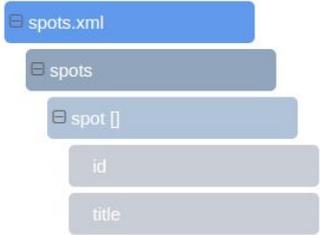
title



```
<?xml version="1.0" encoding="UTF-8"?>
<spots>
  <spot>
    <id>6598</id>
    <title>Play Beach</title>
  </spot>
  <spot>
    <id>6599</id>
    <title>Paintball Gent</title>
  </spot>
</spots>
```

spots.xml

Detail Lowest Low Moderate High Highest



```
<?xml version="1.0" encoding="UTF-8"?>
<spots>
  <spot>
    <id>6598</id>
    <title>Play Beach</title>
  </spot>
  <spot>
    <id>6599</id>
    <title>Paintball Gent</title>
  </spot>
</spots>
```

Use multiple vocabularies and ontologies

not restricted to a single schema

can be changed during mapping process

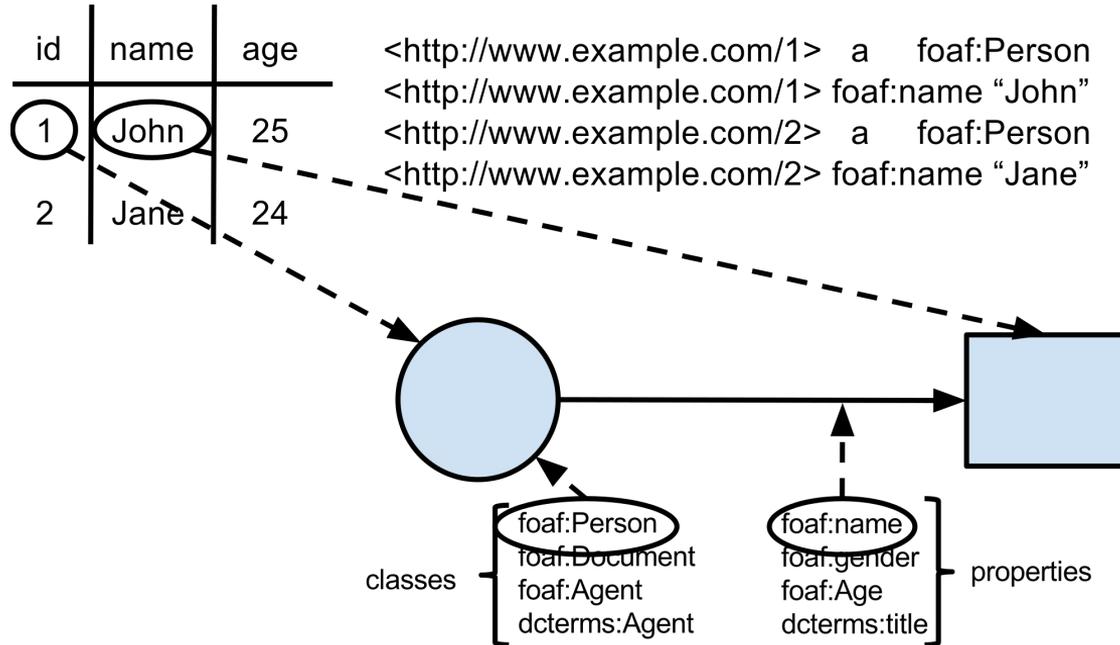
Allow multiple modeling approaches

certain use cases benefit from using specific approach

data-driven

schema-driven

Data-driven start with data to model knowledge



Data-driven via RMLEditor

The screenshot displays the RMLEditor interface. On the left, a tree view shows the structure of the XML file 'spots.xml', with a red box highlighting the 'spot' elements. Below the tree, the XML content is visible:

```
<?xml version="1.0" encoding="UTF-8"?>
<spots>
  <spot>
    <id>6598</id>
    <title>Play Beach</title>
  </spot>
  <spot>
    <id>6599</id>
    <title>Paintball Gent</title>
  </spot>
</spots>
```

On the right, a mapping diagram illustrates the relationships between the XML data and the RML schema. The diagram shows a flow from the XML data to the RML schema elements:

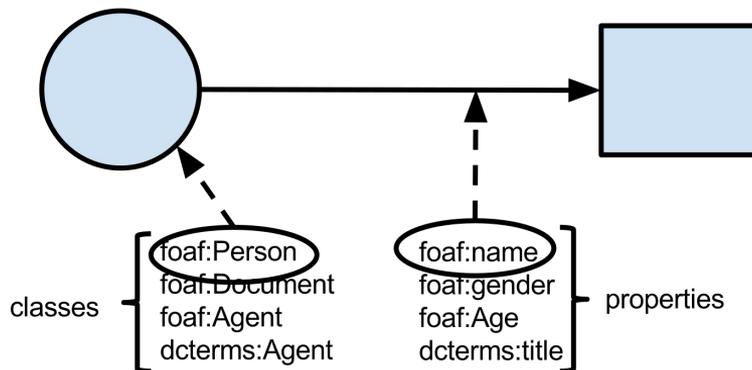
- The XML data (represented by an orange circle) is mapped to the RML schema element 'event:place' (represented by a grey rectangle).
- The RML schema element 'event:place' is mapped to the RML schema element 'ex:{_id}' (represented by a blue circle).
- The RML schema element 'ex:{_id}' is mapped to the RML schema element 'dcterms:title' (represented by a grey rectangle).
- The RML schema element 'dcterms:title' is mapped to the RML schema element 'title' (represented by a blue rectangle).

The diagram also shows a direct mapping from the XML data to the RML schema element 'title' (represented by an orange rectangle).

Schema-driven start with schema to model knowledge

id	name	age
1	John	25
2	Jane	24

<http://www.example.com/1> a foaf:Person
<http://www.example.com/1> foaf:name "John"
<http://www.example.com/2> a foaf:Person
<http://www.example.com/2> foaf:name "Jane"



Schema-driven via RMLEditor

The screenshot displays the RMLEditor interface with a menu bar (File, Edit, Mapping, View) and a toolbar. The main window shows a tree view of the XML document 'spots.xml' with nodes for 'spots', 'spot []', 'id', and 'title'. The central area features a graph diagram with nodes and arrows, enclosed in a red border. The graph includes nodes for 'event.place', 'dcterms:title', and 'title', along with schema nodes like '...oc/{id}' and 'ex:{_id}'. The bottom panel shows the XML source code.

```
<?xml version="1.0" encoding="UTF-8"?>
<spots>
  <spot>
    <id>6598</id>
    <title>Play Beach</title>
  </spot>
  <spot>
    <id>6599</id>
    <title>Paintball Gent</title>
  </spot>
</spots>
```

Allow non-linear workflows

keep overview of mapping model and relationships

linear workflows force user's steps

updating details in specific step is cumbersome

Linear workflows require to retrace steps to update specific step

Step 1  Step 2  Step 3  Step 4  **Step 5**

update step 3

Step 1  Step 2  **Step 3**  Step 4  Step 5

Step 1  Step 2  Step 3  Step 4  **Step 5**

verify step 1

Step 1  Step 2  Step 3  Step 4  Step 5

Step 1  Step 2  Step 3  Step 4  **Step 5**

Non-linear workflows via panels

The screenshot displays the RML EDITOR interface, which is used for editing and visualizing RML (RDF Mapping Language) files. The interface is divided into several panels:

- File Explorer (Left):** Shows a tree view of the file structure for 'spots.xml', including 'spots' and 'spot []' with sub-elements 'id' and 'title'.
- XML Editor (Bottom Left):** Displays the XML content of the file:

```
<?xml version="1.0" encoding="UTF-8"?>
<spots>
  <spot>
    <id>6598</id>
    <title>Play Beach</title>
  </spot>
  <spot>
    <id>6599</id>
    <title>Paintball Gent</title>
  </spot>
</spots>
```
- Diagram (Center):** A visual representation of the RML workflow. It shows two input nodes: an orange circle labeled '...oc/{id}' (representing a local object) and a blue circle labeled 'ex:{id}' (representing a schema object). Both are connected to a central grey rectangle labeled 'event:place'. From 'event:place', arrows point to two 'dcterms:title' rectangles. Each 'dcterms:title' rectangle is connected to a final 'title' rectangle (one orange, one blue).
- Table (Right):** A table showing the data instances for the RML. The columns are Subject, Predicate, and Object.

Subject	Predicate	Object
ex:6719	event:place	ex:loc/6795
ex:6719	dcterms:title	Spannende speurt
ex:6719	a	schema:Event
ex:8470	dcterms:title	Lichtfestival Gent
ex:8470	a	schema:Event
ex:8480	event:place	ex:loc/6691
ex:8480	dcterms:title	Vuurwerk
ex:8480	a	schema:Event
ex:8537	event:place	ex:loc/6601
ex:8537	dcterms:title	Festival de la lumie
ex:8537	a	schema:Event
ex:8547	dcterms:title	Feux d'artifice
ex:8547	a	schema:Event
ex:8594	event:place	ex:loc/6700
ex:8594	dcterms:title	Light Festival Gher
ex:8594	a	schema:Event

Export mappings

mappings can be executed outside the RMLEditor

use reusability of mapping language

in the format specified by the mapping language

not tied to the visualization

Summary features

three panels

- allow multiple modeling approaches

- allow non-linear workflows

independent of mapping language

map multiple data sources at the same time

- possibly in different data formats

use multiple vocabularies and ontologies

export mappings

Overview

mapping process to generate Linked Data

data owners vs. Semantic Web experts during this process

existing tools help the mapping process

RMLEditor

features

user validation

(Non)-Semantic Web experts as participants

10 Semantic Web experts

5 with experience in Linked Data publishing

5 without experience in Linked Data publishing

5 non-Semantic Web experts

Each participant completed two use cases

data-driven: start with the data

schema-driven: start with the schema

experts with experience in LD publishing

also did the use cases by directly writing RML statements

Important results

supported approaches

graph visualization

interlinking between data sources

Data-driven and schema-driven approaches supported

choice driven by personal preference

Graph visualization helps editing

participants found it beneficial

only limited number of use cases

done by SW experts were incomplete or inaccurate

non-Semantic Web experts are able to define mappings

Improved interlinking via graphs

creating links between data in different data sources

certain use cases missed interlinking when using RML directly

20% less when using RMLEditor

spots.xml



Detail

Lowest

Low

Moderate

High

Highest

spots.xml

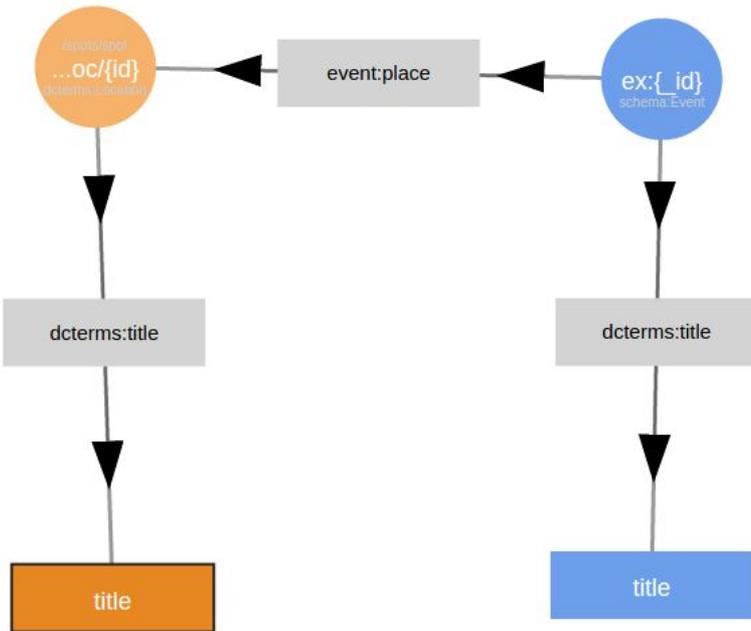
spots

spot []

id

title

```
<?xml version="1.0" encoding="UTF-8"?>
<spots>
  <spot>
    <id>6598</id>
    <title>Play Beach</title>
  </spot>
  <spot>
    <id>6599</id>
    <title>Paintball Gent</title>
  </spot>
</spots>
```



Roundup

data owners and Semantic Web have different areas of expertise

existing tools help the mapping process, but have their limitations

RMLEditor brings editing of mappings to non-Semantic Web experts

See it in action at demo #16

Generate your Linked Data with the RMLEditor

and help us improve our tool

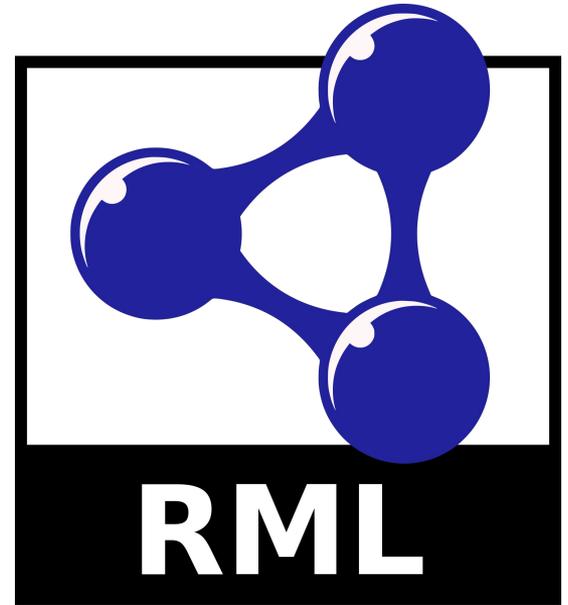
alpha version

free academic license

[contact us for access](#)

Pieter Heyvaert

pheyvaer.heyvaert@ugent.be
visit rml.io



Comparison tools

	fluidsOps editor	DataOps	Sheet2RDF	ontopPro	Karma	RDF123	RMLEditor
non-linear workflow							
data-driven approach							
schema-driven approach							
multiple data sources							
multiple data formats							
no knowledge language							

Details levels to improve editing

highest: all information

high: less details nodes

moderate: no predicates

low: no literals

lowest: no blank nodes