Generating Linked Open Data from heterogeneous Open Data sources with

rml.io

Anastasia Dimou
anastasia.dimou@ugent.be
@natadimou
Ghent University – iMinds – Multimedia Lab
Semantic Web enabled applications rely on data represented as Linked Open Data semantically annotated using ontologies and vocabularies.
Most of the data that we would like to be able to query as Linked Open Data exists in formats other than RDF.
12% of webpages contain any structured data such as microformat, microdata and RDFa derived from 6% of all websites
There are…

over 11,000 APIs according to ProgrammableWeb.org

only 74 return results in RDF

But more than 5000 return results in JSON or XML
Semantic Web

LOD enables SW apps

SW apps demands LOD

Linked Open Data

Open Data
Many languages, tools and approaches were proposed to convert data from different data sources to RDF.
Existing mapping solutions map

*per-format and per-source*

→ focus more on handling the source rather than modeling the domain

OR

provide *case-specific* solutions

→ better model the domain
R2RML mappings defines R2RML processor

- Data OWNER / PUBLISHER
- DB
- CSV
- XML
- JSON
- RDF
- RDF
- RDF
- RDF
The mappings are...

**independently defined** disregarding possible prior definitions
links to other resources (re)using same ontologies for similar data

**manually aligned/interlinked**
by reconstructing the same URIs
by post-mapping interlinking
A well-considered policy is required when mapping data to RDF in the context of a certain knowledge domain that shifts the focus FROM modeling the data of a source TO modeling the domain-level knowledge using the available data source(s)
uniform mapping definitions to describe mapping rules for heterogeneous sources

interoperable mapping definitions that would allow the re-use of mapping rules across different implementations

reusable mapping definitions that would allow the re-use of mapping rules for representing data in the same or different formats
R2RML mappings defines R2RML processor

Data OWNER / PUBLISHER

- DB
- CSV
- XML
- JSON

- RDF
- RDF
- RDF
- RDF
Data OWNER / PUBLISHER defines

Mappings definitions

processor

DB, CSV, XML, JSON

RDF
RDF Mapping Language (RML)

generic scalable mapping language for mapping heterogeneous data into RDF in an integrable and interoperable fashion

superset of the W3C recommended R2RML mapping language

http://rml.io
RDF Mapping Language

RML.io
RDF Mapping Language (RML)

- Triples Map
- Subject Map
- Predicate-Object Map
- Logical Source
- Source
- Iterator
- Reference Formulation
- Term Map
- Predicate Map
- Object Map
- Referencing Object Map
- Join Condition
- Parent column
- Child column
- Triangles Map
- constant
- reference
- template
- Multimedia
RDF Mapping Language

RML

RML generating triples

RML reusing mappings

RML aligning & interlinking
RDF Mapping Language (RML)

<subject>  <predicate>  <object>

<http://ex.com/Anastasia%20Dimou> a ex:Person .
RDF Mapping Language (RML)

RDF Term:
a URI
a literal
a blank node
RML subject

<table>
<thead>
<tr>
<th>NAME</th>
<th>BIRTH_DATE</th>
<th>DEATH_DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Robert Theodore McCall</td>
<td>1919-12-23</td>
<td>2010-02-26</td>
</tr>
<tr>
<td>Ronald Anderson</td>
<td>1929-12-06</td>
<td></td>
</tr>
</tbody>
</table>

```
<#ArtistMapping>
rr:subjectMap [ rr:template "http://ex.com/{NAME}" ; rr:class ex:Person ];

<http://ex.com/Robert%20Theodore%20McCall> a ex:Person
```
# RML predicate & object

<table>
<thead>
<tr>
<th>NAME</th>
<th>BIRTH_DATE</th>
<th>DEATH_DATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Robert Theodore McCall</td>
<td>1919-12-23</td>
<td>2010-02-26</td>
</tr>
<tr>
<td>Ronald Anderson</td>
<td>1929-12-06</td>
<td></td>
</tr>
</tbody>
</table>

```rml
<#ArtistMapping>
rr:predicateObjectMap [
  rr:predicate ex:birth_date;
  rr:objectMap [
    rr:column "BIRTH_DATE" ]
];

<http://ex.com/Robert%20Theodore%20McCall> ex:birth_date "1919-12-23"
```
artworks.JSON
[ ... ...
    { "Title": "Apollo 11 Crew",
      "Artist": "Ronald Anderson",
      "Ref": "NPG_70_36",
      "Sitter": [
        { "Name": "Neil Armstrong",
          "Birth Date": "1930-08-05" },
        { "Name": "Buzz Aldrin",
          "Birth Date": "1930-01-20" },
        { "Name": "Michael Collins" } ],
      "DateOfWork": "1969" },
    { "Title": "Neil Armstrong",
      "Artist": "Robert Theodore McCall",
      "Ref": "S_NPG_2010_51",
      "Sitter": [  
        { "Name": "Neil Armstrong" } ],
      "DateOfWork": "2009" },
... ... ]

artists.XML
<Artists> ... ...
  <Artist>
    <Name>Robert Theodore McCall</Name>
    <Birth_Date>1919-12-23</Birth_Date>
    <Death_Date>2010-02-26</Death_Date>
  </Artist>
  <Artist>
    <Name>Ronald Anderson</Name>
    <Birth_Date>1929-12-06</Birth_Date>
  </Artist> ... ...
</Artists>
Triples Map
Logical Source rml:logicalSource
source [ rml:source "artists.xml" ;
Reference Formulation rml:referenceFormulation ql:XPath ].

Triples Map
Logical Source rml:logicalSource
source [ rml:source "http://ex.com/artworks.json" ;
Reference Formulation rml:referenceFormulation ql:JSONPath ].
{ "Title": "Apollo 11 Crew", "Artist": "Ronald Anderson", "Ref": "NPG_70_36", "Sitter": [ { "Name": "Neil Armstrong", "Birth Date": "1930-08-05" }, { "Name": "Buzz Aldrin", "Birth Date": "1930-01-20" }, { "Name": "Michael Collins" } ], "DateOfWork": "1969" },
{ "Title": "Neil Armstrong", "Artist": "Robert Theodore McCall", "Ref": "S_NPG_2010_51", "Sitter": [ { "Name": "Neil Armstrong" } ], "DateOfWork": "2009" },
... ... ]
<http://ex.com/Robert+Theodore+McCall> ex:death_date "1929-12-06".

<Artists>...
</Artists>

<Artists>...
</Artists>

<Artist>
<Name>Robert Theodore McCall</Name>
<Birth_Date>1929-12-06</Birth_Date>
<Death_Date>2010-02-26</Death_Date>
</Artist>

<Artist>
<Name>Ronald Anderson</Name>
<Birth_Date>1929-12-06</Birth_Date>
<Death_Date>1919-12-23</Death_Date>
</Artist>...

</Artists>
RDF Mapping Language

RML

RML generating triples
RML reusing mappings
RML aligning & interlinking
Avoid…

redefining and replicating URI patterns

remodeling the same domain

Uniquely define the URI patterns that generates a resource and refer to its definition
{ "Performance": {
  "Perf_ID": "567",
  "Location": {
    "lat": "51.043611",
    "long": "3.717222"
  }
}, ...

<Events> ...
  <Exhibition id="398">
    <Location>
      <lat>51.076891</lat>
      <long>3.717222</long>
    </Location>
  </Exhibition> ...
</Events>
{ ... "Performance" :
{ "Perf_ID": "567",
 "Location": {
   "lat": "51.043611",
   "long": "3.717222" }
}, ...

<Events> ...

<Exhibition id="398">
  <Location>
    <lat>51.076891</lat>
    <long>3.717222</long>
  </Location>
</Exhibition> ...

</Events>

<#LocationMapping>
  rr:subjectMap [ 
    rr:template "http://ex.com/{lat},{long}" ];

  rr:.predicateObjectMap [ 
    rr:predicate ex:long; 
    rr:objectMap [ rml:reference "long" ] ];

  rr:.predicateObjectMap [ 
    rr:predicate ex:lat; 
    rr:objectMap [ rml:reference "lat" ] ];

ex:51.043611, 3.717222
ex:lat "3.717222",  ex:long "51.043611".

ex:51.076891, 3.717222
ex:lat "3.717222",  ex:long "51.043611".

"Performance" : 
{ "Perf_ID": "567",
 "Location": {
   "lat": "51.043611",
   "long": "3.717222" } },
... }
RDF Mapping Language

RML

RML generating triples
RML reusing mappings
RML aligning & interlinking
{ ...
  "Performance": {
    "Perf_ID": "567",
    "Venue": {
      "Name": "STAM",
      "Venue_ID": "78"
    },
    "Location": {
      "long": "3.717222",
      "lat": "51.043611"
    }
  }, ...
}

<#PerformancesMapping>
  rr:subjectMap [ 
    rr:template "http://ex.com/{Perf_ID}" ];
  rr:predicateObjectMap [ 
    rr:predicate ex:venue;
    rr:objectMap [ 
</#PerformancesMapping>

<#VenueMapping>
  rml:logicalSource [ 
    rml:source "http://ex.com/performances.json";
    rml:referenceFormulation ql:JSONPath;
    rml:iterator "$.[Performance.Venue.[*]]" ];
  rr:subjectMap [ 
    rr:template "http://ex.com/{Venue_ID}";
    rr:class ex:Venue ] .
</#VenueMapping>

ex:567 ex:venue ex:78
{ ... "Performance" : 
{ "Perf_ID": "567", 
"Venue": { 
"Name": "STAM", 
"Venue_ID": "78" }, 
... } 

<Events> ...
<Exhibition id="398">
  <Venue>STAM</Venue>
</Exhibition> ...
</Events>

ex:567 ex:venue ex:78.  
ex:398 ex:venue ex:STAM.  
ex:78 owl:sameAs ex:STAM  
ex:567 ex:venue ex:78.  
ex:398 ex:venue ex:78.  
ex:78 owl:sameAs ex:STAM
RDF Mapping Language (RML)

Processing
Extraction Module

Mapping Module

RML Processor

RDF

XML

JSON
rml.io

Anastasia Dimou

@natadimou
anastasia.dimou@ugent.be