RDA/RDF at the University of Washington Libraries

SWIB21
Semantic Web in Libraries

Crystal Clements
Theodore Gerontakos
Melissa Morgan
Benjamin Riesenber
Our focus today

- Application profiles
- Mappings/alignments of data models
- Data transformations

...proficiencies essential to current metadata practice.
Metadata in the 2020s

● Despite changes, some continuity 1990s-present:
  ○ Increase interoperability
  ○ Adhere to standards

bit.ly/rdardfuwl
One interoperability issue

Merging data modelled differently

Not a trivial problem

- Institution one’s data is RDA data
- Institutions two’s data is BIBFRAME
- Institution three’s data is Dublin Core
Proposed [partial] solution:

- Move fluently across adopted data models
- Create tools and proficiencies that allow this (profiles, mappings, transformations)
"Adhere to standards"

1. RDA
2. BIBFRAME is another standard we’re interested in today
3. MARC
Change in 2020s metadata

- Shift to linked data
- Emergence of abundant linked data infrastructure, including publishing platforms
Sinopia

Linked data platform

Allows unlimited data models

Testing ground of “fluency across data models”
Workflow 1 (complete)

- Input RDA (2017)
- Output BIBFRAME
- Publish both side-by-side in Sinopia
Workflow 2 (in progress)

- Input legacy MARC
- Output RDA/LRM/RDF and BIBFRAME
- Publish both side-by-side in Sinopia
Demonstrates…

A way to move across data models

A way to increase interoperability

A commitment to standards
Labor intensive approach

Alignment/mapping especially requires significant effort

Sharing this work would be helpful

- Shared methods
- Share mappings across institutions
- Make them findable

bit.ly/rdardfuwl
The shift of metadata practice in the 2020s includes…

- Fluency across data models
- Community of mapping/alignment
- Application profiles
- Ability to write transformation code
RDA/RDF Metadata Application Profiles
PAST

A Cataloging and Metadata Services work group at the University of Washington Libraries created RDA/RDF metadata application profiles during the Linked Data for Production: Pathway to Implementation project

RESOURCES

- Evaluating RDA Registry Application Profiles for Linked-Data Library Cataloging / presentation slides - PDF
- Managing and Outputting Sinopia Profiles from a Single Source / presentation slides - PDF
- 2019-2020 resource templates and description sets as HTML
- Proof-of-concept - RDA input form and output of RDA/RDF and BIBFRAME 2.0 / more information

bit.ly/rdardfuwl
PRESENT

New workflow for creating RDA/RDF resource templates under construction

RESOURCES

- **map_storage**: Store and track changes to implementation of RDA/RDF (and other) properties in XML instances
- **sinopia_maps**: Output RDA/RDF resource templates and documentation from “storage”
FUTURE

*Continue creating RDA/RDF metadata description sets*

- Build experience implementing RDA Registry terms for bibliographic description in RDF
- Build experience implementing metadata application profiles for linked library data

- Continue engagement with the LD4P/Sinopia community; take advantage of ongoing improvements to the Linked Data Editor
- Continue with a proof-of-concept: Create data using RDA3R/RDF, output derivative BIBFRAME descriptions...

[bit.ly/rdardfuwl]
RDA/RDF to BIBFRAME Conversion

bit.ly/rdardfuwl
How our conversion code works

- Python script takes in our RDA-to-BIBFRAME mapping in kiegel, and outputs RML
- Use that RML and the RML Mapper to convert our RDA/RDF to BIBFRAME
Example of our mappings in kiegel syntax

Our mapping for rda:P10004 (has category of work):

\[
\text{genreForm*} \\
\text{or} \\
\text{genreForm} \gg \text{GenreForm} > \text{rdfs:label}
\]

How this would translate to Turtle:

\[
\text{# with IRI value} \\
ex:exampleResource \text{ bf:genreForm} <\text{P10004value}> . \\
\text{# with literal value} \\
ex:exampleResource \text{ bf:genreForm} [ a \text{ bf:GenreForm} ; \text{ rdfs:label } "\text{P10004value}" ] .
\]
How our Python script reads kiegel

genreForm*

or

genreForm >> GenreForm > rdfs:label

→

{"IRI": ["genreForm*"], "literal": ["genreForm", ">>, "GenreForm", ">", "rdfs:label"]}
What is RML?

- RML = RDF Mapping Language
- Developed by Ghent University
- Extension of R2RML (RDB to RDF Mapping Language)
- Specifications last updated 6 Oct 2020 — [https://rml.io/specs/rml/](https://rml.io/specs/rml/)

- Simple
- Repetitive
- RML mappings themselves are RDF graphs
Building blocks of RML

**Triples Map**
Specifies the rules for translating the source data into zero or more RDF triples

**Logical Source**
Specifies the source data

**Subject Map**
Rules for generating a subject for the triple(s)

**Predicate-Object Map**
Rules for generating a predicate and object for the triple(s)

---

**RML Graph**

```xml
@prefix bf: <http://id.loc.gov/ontologies/bibframe/>.
@prefix ex: <http://example.org/rules/>.
@prefix rdaw: <http://rdaregistry.info/Elements/w/>.
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix rml: <http://semweb.mmlab.be/ns/rml#>.
@prefix rr: <http://www.w3.org/ns/r2rml#>.
@prefix ql: <http://semweb.mmlab.be/ns/ql#>.

ex:ExampleMap a rr:TriplesMap;
   rml:logicalSource {
     rml:source "RML_demo_data.xml"; # file path to our data
     rml:referenceFormulation ql:XPath; # what format our source data is in
     rml:iterator "/RDF/Description" # our parent properties
   };
   rr:subjectMap {
     rml:reference "@about"; # "reference" means we're referencing a value that exists in our data
     rr:class bf:Work # optional: add a class
   };
   rr:predicateObjectMap {
     rr:predicate bf:genreForm; # determine predicate
     rr:objectMap {
       rml:reference "P10004/@resource"; # again using "reference" to locate an existing value
       rr:termType rr:IRI # type as a literal, IRI, or blank node
     } .
   } .
```
Building blocks of RML

Python using rdflib

```python
import rdflib

RML_graph = Graph()

logical_source_bnode = BNode()
subject_map_bnode = BNode()
po_map_bnode = BNode()
object_map_bnode = BNode()

RML_graph.add((ex.ExampleMap, rdf.type, rr.TriplesMap))
RML_graph.add((ex.ExampleMap, rml.logicalSource, logical_source_bnode))
RML_graph.add((logical_source_bnode, rml.source, Literal("RML_demo_data.xml")))
RML_graph.add((logical_source_bnode, rml.referenceFormulation, ql.XPATH))
RML_graph.add((logical_source_bnode, rml.iterator, Literal("/RDF/Description")))
RML_graph.add((ex.ExampleMap, rr.subjectMap, subject_map_bnode))
RML_graph.add((subject_map_bnode, rml.reference, Literal('@about')))  # "reference" means we're referencing a value that exists in our data
RML_graph.add((subject_map_bnode, rr.class, bf.Work))
RML_graph.add((ex.ExampleMap, rr.predicateObjectMap, po_map_bnode))
RML_graph.add((po_map_bnode, rr.predicate, bf.genreForm))
RML_graph.add((po_map_bnode, rr.objectMap, object_map_bnode))
RML_graph.add((object_map_bnode, rml.reference, Literal("P10004/@resource")))
RML_graph.add((object_map_bnode, rr.termType, rr.IRI))
```

RML Graph

```python
RML_graph = Graph()

logical_source_bnode = BNode()
subject_map_bnode = BNode()
po_map_bnode = BNode()
object_map_bnode = BNode()

RML_graph.add((ex.ExampleMap, rdf.type, rr.TriplesMap))
RML_graph.add((ex.ExampleMap, rml.logicalSource, logical_source_bnode))
RML_graph.add((logical_source_bnode, rml.source, Literal("RML_demo_data.xml")))
RML_graph.add((logical_source_bnode, rml.referenceFormulation, ql.XPATH))
RML_graph.add((logical_source_bnode, rml.iterator, Literal("/RDF/Description")))
RML_graph.add((ex.ExampleMap, rr.subjectMap, subject_map_bnode))
RML_graph.add((subject_map_bnode, rml.reference, Literal('@about')))  # "reference" means we're referencing a value that exists in our data
RML_graph.add((subject_map_bnode, rr.class, bf.Work))
RML_graph.add((ex.ExampleMap, rr.predicateObjectMap, po_map_bnode))
RML_graph.add((po_map_bnode, rr.predicate, bf.genreForm))
RML_graph.add((po_map_bnode, rr.objectMap, object_map_bnode))
RML_graph.add((object_map_bnode, rml.reference, Literal("P10004/@resource")))
RML_graph.add((object_map_bnode, rr.termType, rr.IRI))
```
MARC21 TO RDA-RDF

A METADATA MAPPING PROJECT AT THE UNIVERSITY OF WASHINGTON LIBRARIES

GitHub Repository: https://github.com/uwlib-cams/MARC2RDA
Project Purpose

In order to move forward with RDA-RDF cataloging, we want to be able to transform our millions of legacy MARC21 records into RDA-RDF.

Before we can create a conversion tool to achieve this goal, we need a detailed and authoritative mapping between the two standards.
Existing Mappings

RDA Registry Mapping from the RSC

TMQ, Inc. Mapping used in RIMMF3

Original RDA Toolkit Mapping
Project Update

MARC21 to RDA-RDF Mapping Project

MAPPING BETWEEN MARC21 AND RDA-RDF

View the Project on GitHub
uwlib-cams/MARC2RDA

MARC2RDA

Mapping between MARC21 and RDA-RDF

About

This is a project of the Metadata and Cataloging Initiatives Unit of the Cataloging and Metadata Services Department at the University of Washington Libraries initiated in 2021 to create a more robust mapping between the MARC21 bibliographic format and RDA-RDF, with the goal of facilitating a future project to create a corresponding data conversion tool. Authority data is out of scope for this mapping. This project builds on the work of the RDA/MARC 21 Alignment Task Force within the RSC Technical Working Group, specifically the mapping available here.

Contributing

Changes to GitHub instructions (in PDF format to easily include screenshots) can be made here.

Communication

Please post issues and related discussions here.

Resources
null
UW Mapping Spreadsheet Columns

- NotMapped
- MARCField
- MARCFieldLabel
- MARCInd1Label
- MARCInd1Value
- MARCInd1ValueLabel
- MARCInd2Value
- MARCInd2ValueLabel
- CharacterPosition
- CharacterPositionLabel
- MARCSubfield
- MARCSubfieldLabel
- CodeValue
- CodeValueLabel

- MARCTagCondition1
- Condition1Values
- MARCTagCondition2
- Condition2Values
- RDA Registry URI
- RDA Registry Label
- Recording Method
- Notes columns:
  - Justification for Mapping
  - Transformation Notes
  - Problems with Mapping
  - Notes (Uncategorized)
- Milestone
Next Steps

- Seek out collaborators to help complete mapping
  - Please email cec23@uw.edu if you are interested in collaborating!
  - For more information, see our GitHub Repository
- Establish a common mapping for all stakeholders and publish for global reuse
- Launch another project to build a conversion tool
  - Mapping is being written with this goal in mind
Thank you

**IMAGE DETAILS:**
- Piet Schipperus (1840–1929), CC0, [via Wikimedia Commons](https://commons.wikimedia.org/wiki/File:Piet_Schipperus_(1840–1929).jpg)
- Cadop, CC0, [via Wikimedia Commons](https://commons.wikimedia.org/wiki/File:16.png)
- Jorge Mendoza, CC0, [via Wikimedia Commons](https://commons.wikimedia.org/wiki/File:16.png)
- Samer Kayali, CC0, [via Wikimedia Commons](https://commons.wikimedia.org/wiki/File:16.png)
- William Henry Fox Talbot, CC0, [via Wikimedia Commons](https://commons.wikimedia.org/wiki/File:16.png)