Tools to be used with RML

List of tools on RML.io <https://rml.io/tools/>

Implementation report comparing several RML tools: <https://rml.io/implementation-report/>

# RML Mapper, CARML, and Rocket RML

Input: (semi-)structured data sources in various structures, formats, and serializations

|  |  |  |  |
| --- | --- | --- | --- |
|  | **RML Mapper** | **CARML** | **Rocket RML** |
| GitHub | <https://github.com/RMLio/rmlmapper-java> | <https://github.com/carml/carml> | <https://github.com/semantifyit/RocketRML> |
| Paper | <http://ceur-ws.org/Vol-1184/ldow2014_paper_01.pdf> |  | <https://arxiv.org/pdf/1903.04969.pdf> |
| Rules | RML | RML | RML |
| Programming | Java (on top of RDF4J) | Java | Javascript |
| Rationale | 1. Load all data in memory;  2. Sequentially ingest data sources;  3. Start RDF generation. | 1. Load small parts of data in memory  2. Generate RDF  3. Repeat |  |
| Benefits |  | Handle high-volume data | High speed |
| Restrictions | not suitable for high-volume data | Low speed | Limit the support of data sources or rules range |
| Comments |  |  | A NodeJS implementation of the RML mapper |

# RML Streamer

Original paper: <https://dl.acm.org/doi/abs/10.1145/3323878.3325802>

Can convert files or data streams to RDF. It is available as a **docker container** ([instructions](https://github.com/RMLio/RMLStreamer/blob/development/docker/README.md)). (I am unable to run it on my windows pc, could just be my lack of docker and Flink knowledge) - Nirupama

# RML Editor

Nirupama

<https://rml.io/tools/rmleditor/>

We only find a YouTube introduction in this page. These are the features I see from the video.

Structure of the interface: Has a window to see the input data, one to see the mapping rules in a graphical representation and the output rdf in a table of triples.

Input data: Seems like you can upload CSV, JSON and XML files as input.

Mapping options:

* The mapping rules are set up as graphs, classes are circles, attributes are rectangles with a predicate connecting them to a class.
* IRIs can be manually added for all items
* You can search for ontological terms on Linked Open Vocabularies
* You can either make mapping rules and link it to data or upload the data and make the mapping rules
* I am not sure if mapping rules are reusable or downloadable. The introduction video did not show any RML code.
* It is possible to modify the input with an ‘if’ condition and use the modified column in the mapping rules graph

Publication: <https://link.springer.com/chapter/10.1007/978-3-319-34129-3_43>

YouTube Channel: <https://www.youtube.com/channel/UC8CNBPVQw4lPIo0L8Mp2iaA>

# Sdm-rdfizer:

Mark.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| GitHub | Documentation | License?  Language? | Docker? | Speed tests? | Rules? | Restrictions? | Notable benefits? |
| <https://github.com/SDM-TIB/SDM-RDFizer> | <https://github.com/SDM-TIB/SDM-RDFizer/wiki/Install&Run> | Apache License 2.0 Python | <https://hub.docker.com/r/oegdataintegration/sdmrdfizer>  (this does not work, but the docker image in the GitHub builds corrrectly) | About 30-40X faster than RMLMapper | Still don’t know |  | Exposes a simple GET interface in Docker – will be easy to pipeline this!  Seems to work “out of the box”  Using port-forwarding, you could dynamically expose ONLY THE MAPPED portions of your dataset to the open Web! |

