Creating RDF Knowledge Graphs from structured data using mapping languages

Connected Data London 2025

Michael Rauch

zazuko

Agenda

Introduction to three mapping languages: R2RML, RML, XRM

Hands-On

- Map & transform: CSV and XML to RDF
- Query with SPARQL
- Validate with SHACL and generate datamodel documentation

Goal
Get hands-on experience using mapping languages to create a RDF knowledge graph from existing structured data and learn how to implement this transformation in a reliable and maintainable way that anticipates and supports change and iterative refinement.

R2RML

A language for mappings from **relational databases** to RDF

The mappings are themselves RDF graphs

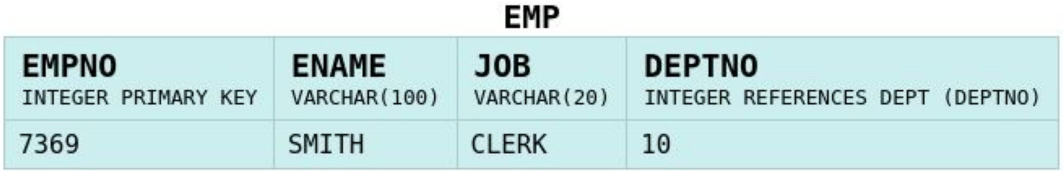
R2RML enables different types of processors, for example:

- Materialization: generate RDF dumps
- Virtualization: offer a virtual SPARQL endpoint over the relational data

https://www.w3.org/TR/r2rml/

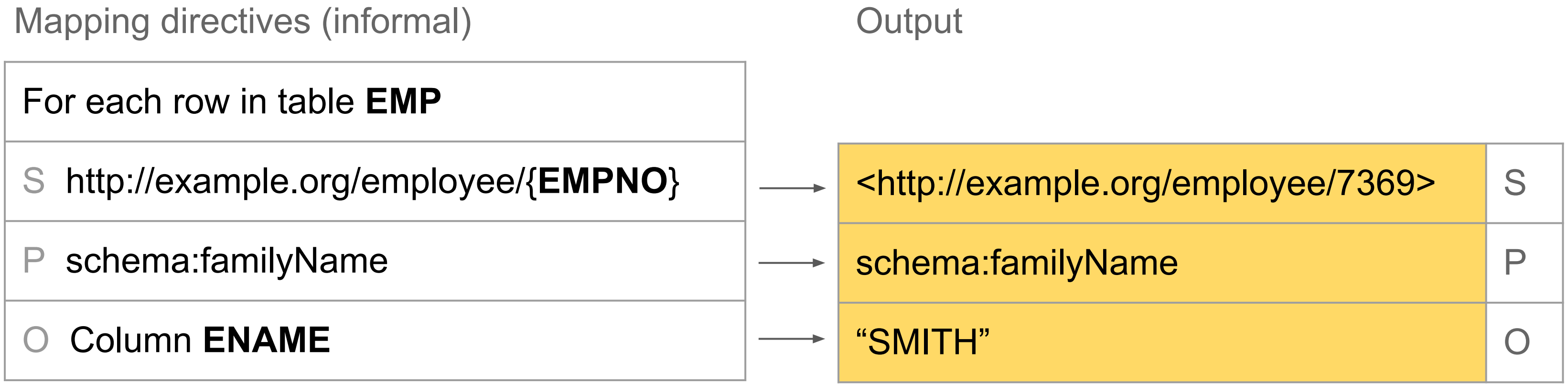
https://www.w3.org/TR/rdb-direct-mapping/

What do we need to say, to make triples from a table?



EMP

EMPNO INTEGER PRIMARY KEY	**ENAME** VARCHAR(100)	**JOB** VARCHAR(20)	**DEPTNO** INTEGER REFERENCES DEPT (DEPTNO)
7369	SMITH	CLERK	10



Mapping directives (informal)

For each row in table **EMP**

S http://example.org/employee/{**EMPNO**}

P schema:familyName

O Column **ENAME**

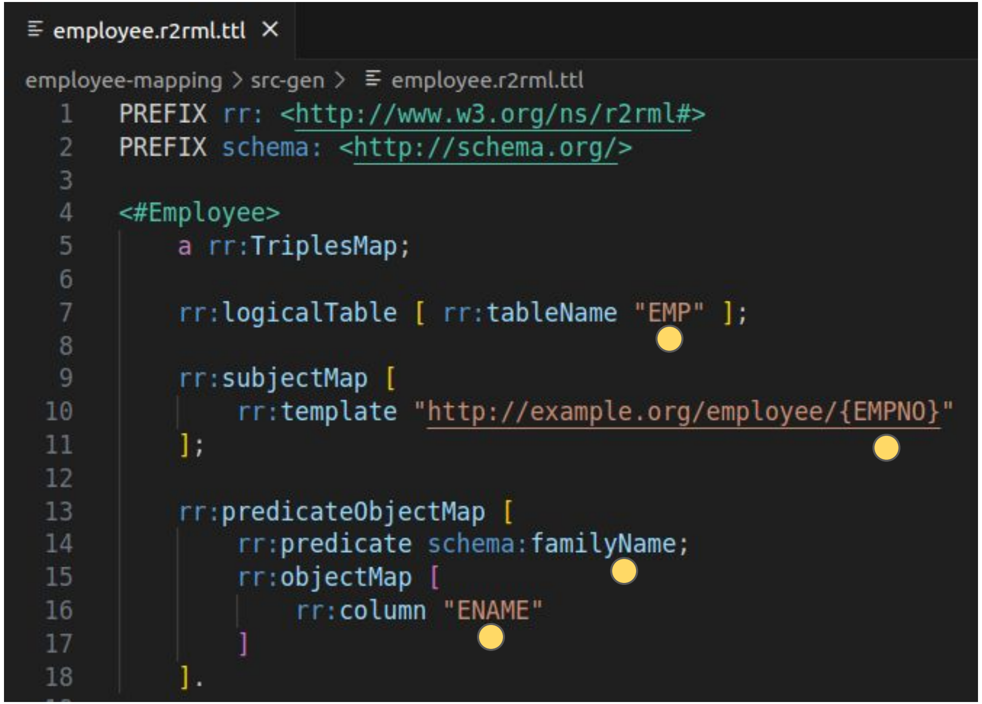
Output

<http://example.org/employee/7369> S

schema:familyName P

“SMITH” O

Formalized mapping in R2RML



```
PREFIX rr: <http://www.w3.org/ns/r2rml#>
PREFIX schema: <http://schema.org/>

<#Employee>
    a rr:TriplesMap;

    rr:logicalTable [ rr:tableName "EMP" ];

    rr:subjectMap [
        rr:template "http://example.org/employee/{EMPNO}"
    ];

    rr:predicateObjectMap [
        rr:predicate schema:familyName;
        rr:objectMap [
            rr:column "ENAME"
        ]
    ].
```

RML

RML generalizes the concepts of R2RML to **any type of data source**

For example CSV, XML, JSON, …

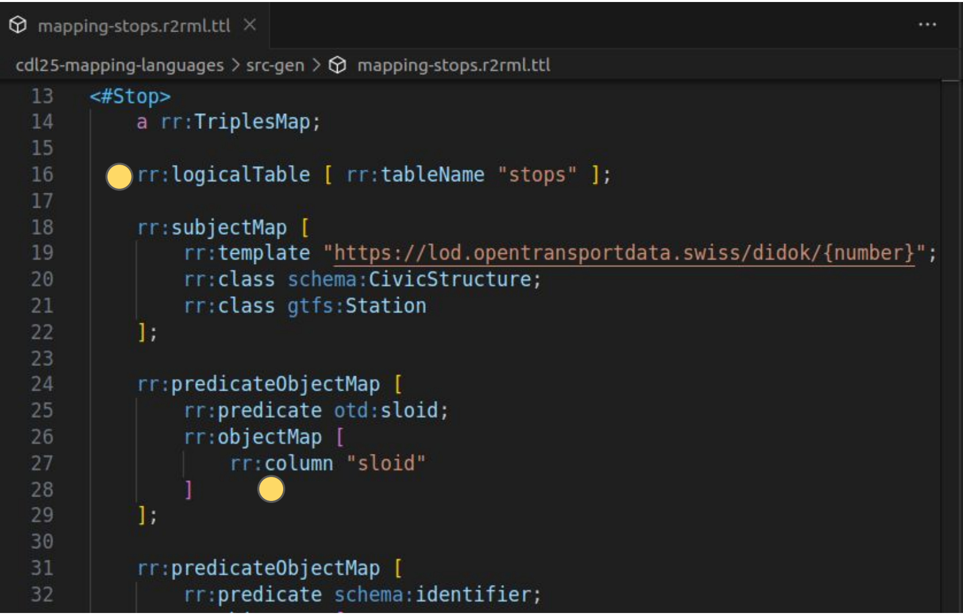
https://rml.io/specs/rml/

https://rml.io/docs/

https://github.com/kg-construct

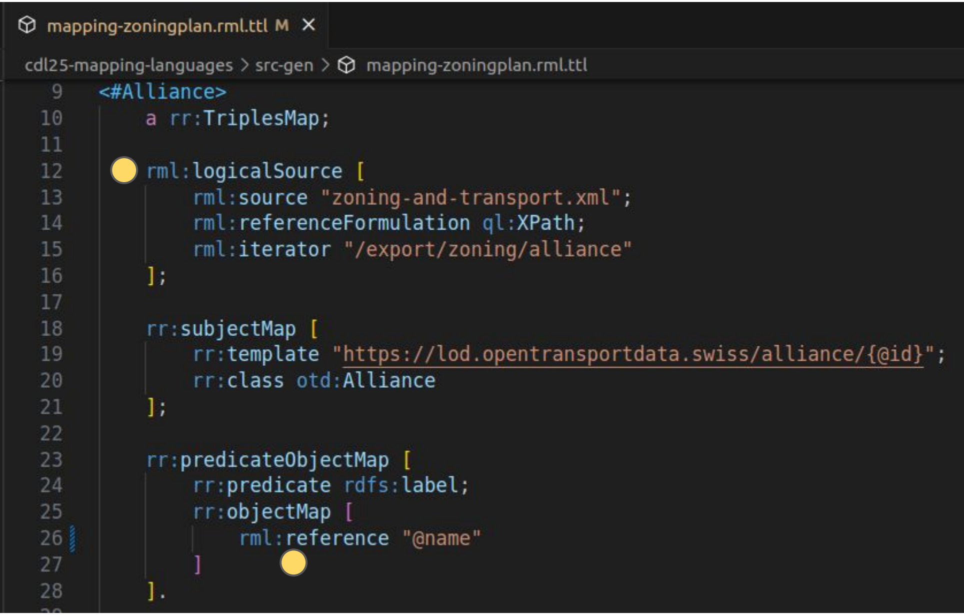
Differences between R2RML and RML

R2RML



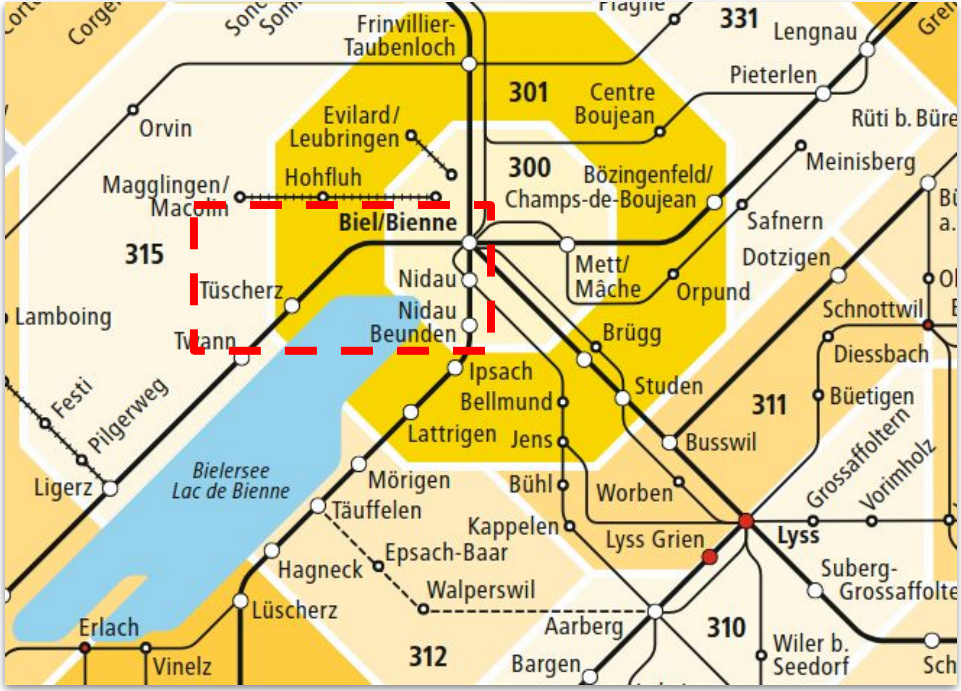
```
<#Stop>
    a rr:TriplesMap;

    rr:logicalTable [ rr:tableName "stops" ];

    rr:subjectMap [
        rr:template "https://lod.opentransportdata.swiss/didok/{number}";
        rr:class schema:CivicStructure;
        rr:class gtfs:Station
    ];

    rr:predicateObjectMap [
        rr:predicate otd:sloid;
        rr:objectMap [
            rr:column "sloid"
        ]
    ];

    rr:predicateObjectMap [
        rr:predicate schema:identifier;
```

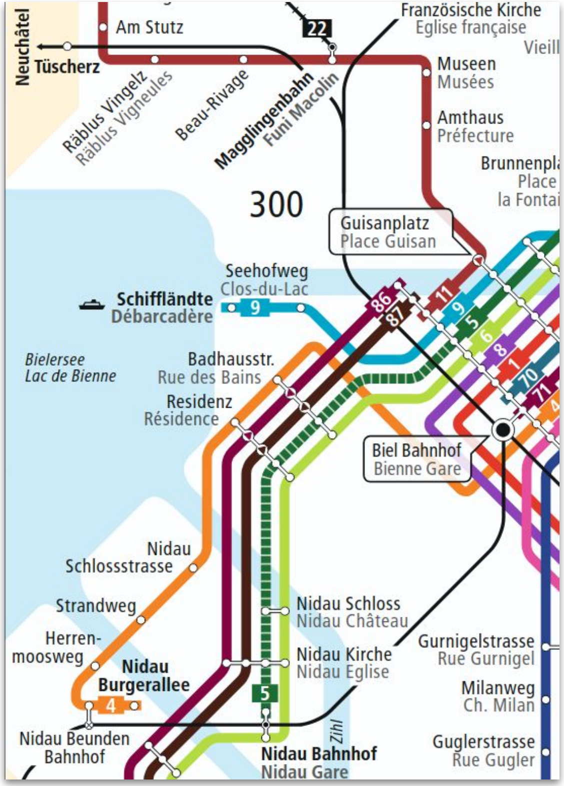
RML


mapping-zoningplan.rml.ttl M
cdl25-mapping-languages > src-gen > mapping-zoningplan.rml.ttl

```
<#Alliance>
    a rr:TriplesMap;

    rml:logicalSource [
        rml:source "zoning-and-transport.xml";
        rml:referenceFormulation ql:XPath;
        rml:iterator "/export/zoning/alliance"
    ];

    rr:subjectMap [
        rr:template "https://lod.opentransportdata.swiss/alliance/{@id}";
        rr:class otd:Alliance
    ];

    rr:predicateObjectMap [
        rr:predicate rdfs:label;
        rr:objectMap [
            rml:reference "@name"
        ]
    ].
```

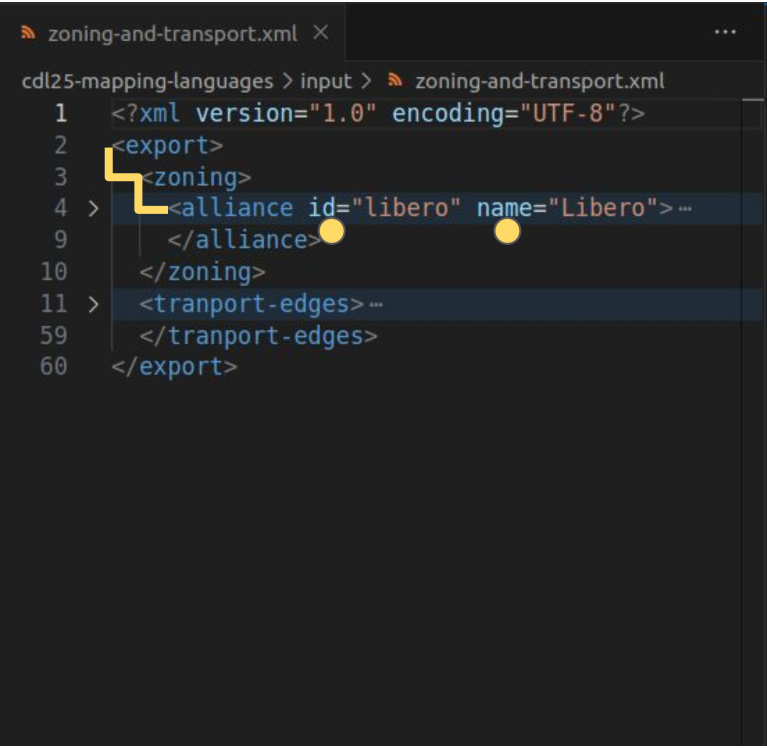
Hands-On - sample data from public transport domain





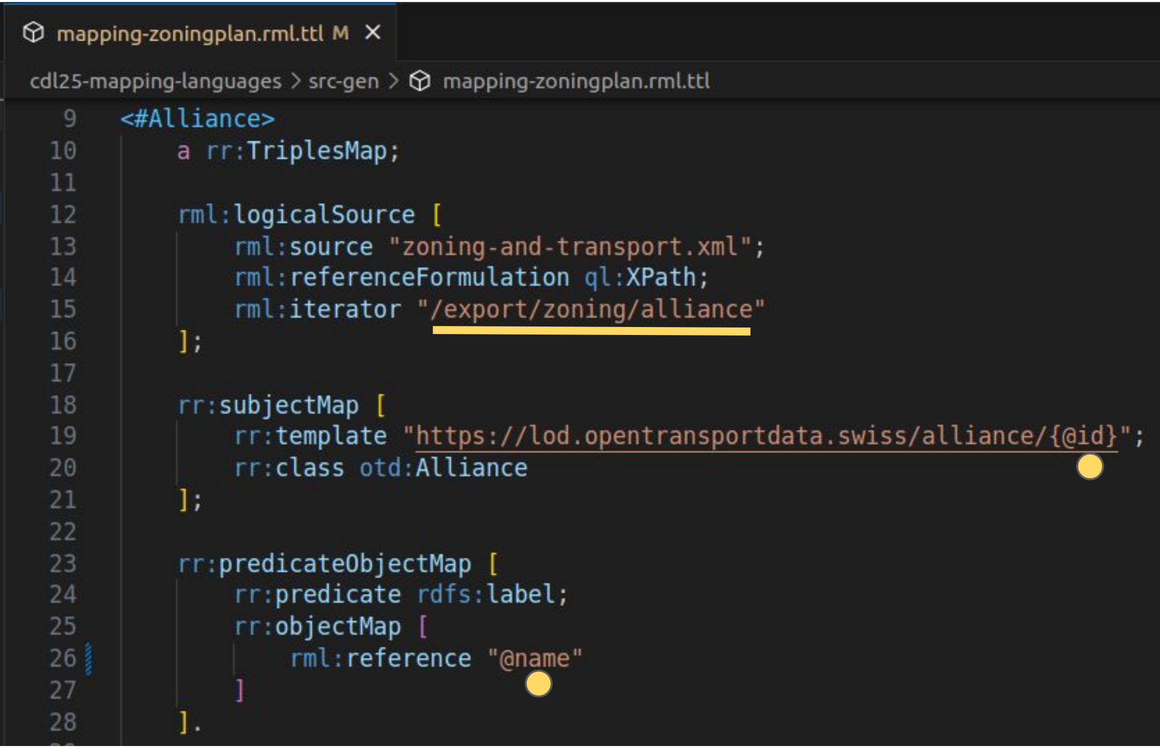
XML datasource - XPath references

XML input


zoning-and-transport.xml ×
cdl25-mapping-languages > input > zoning-and-transport.xml

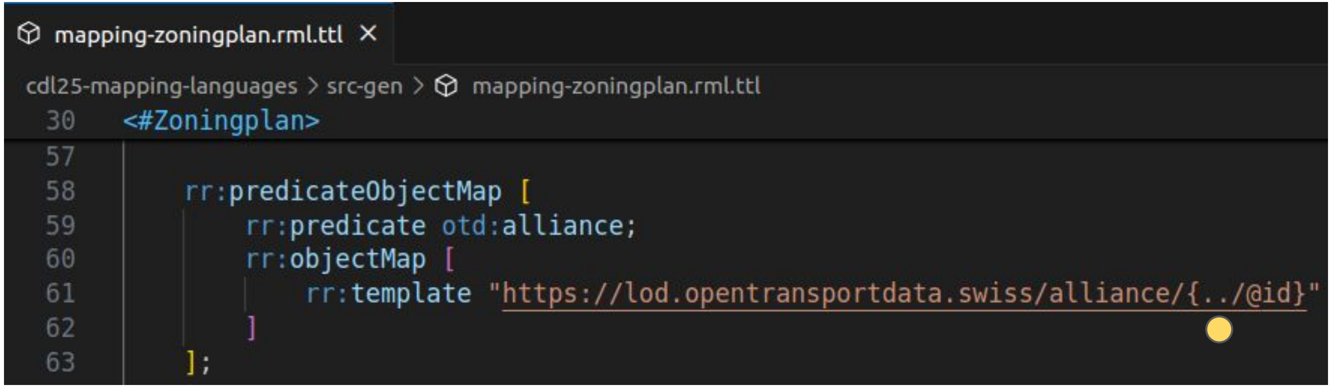
```
<?xml version="1.0" encoding="UTF-8"?>
<export>
  <zoning>
    <alliance id="libero" name="Libero">⋯
    </alliance>
  </zoning>
  <tranport-edges>⋯
  </tranport-edges>
</export>
```

RML mapping


mapping-zoningplan.rml.ttl M ×
cdl25-mapping-languages > src-gen > mapping-zoningplan.rml.ttl

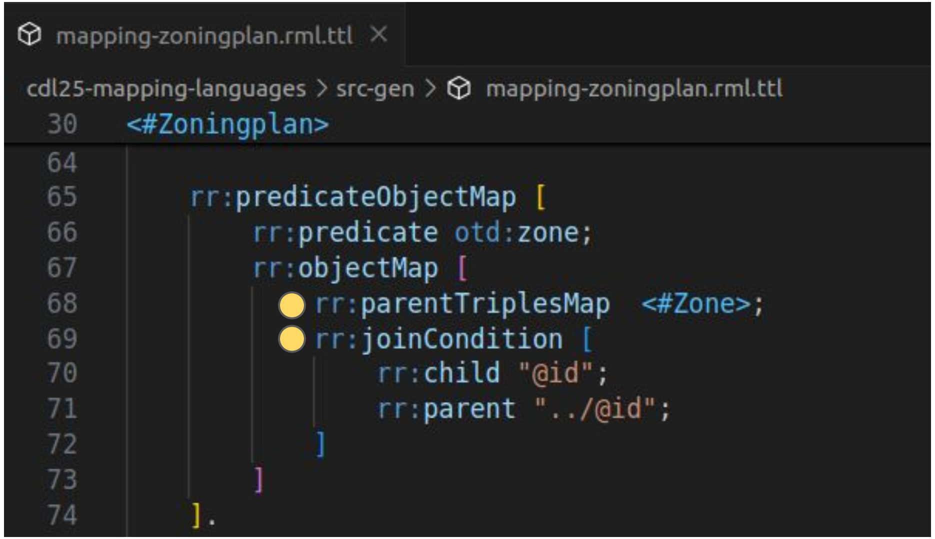
```
<#Alliance>
    a rr:TriplesMap;

    rml:logicalSource [
        rml:source "zoning-and-transport.xml";
        rml:referenceFormulation ql:XPath;
        rml:iterator "/export/zoning/alliance"
    ];

    rr:subjectMap [
        rr:template "https://lod.opentransportdata.swiss/alliance/{@id}";
        rr:class otd:Alliance
    ];

    rr:predicateObjectMap [
        rr:predicate rdfs:label;
        rr:objectMap [
            rml:reference "@name"
        ]
    ].
```

Making links with string templates (rr:template)


mapping-zoningplan.rml.ttl ×
cdl25-mapping-languages > src-gen > mapping-zoningplan.rml.ttl

```
<#Zoningplan>

    rr:predicateObjectMap [
        rr:predicate otd:alliance;
        rr:objectMap [
            rr:template "https://lod.opentransportdata.swiss/alliance/{../@id}"
        ]
    ];
```

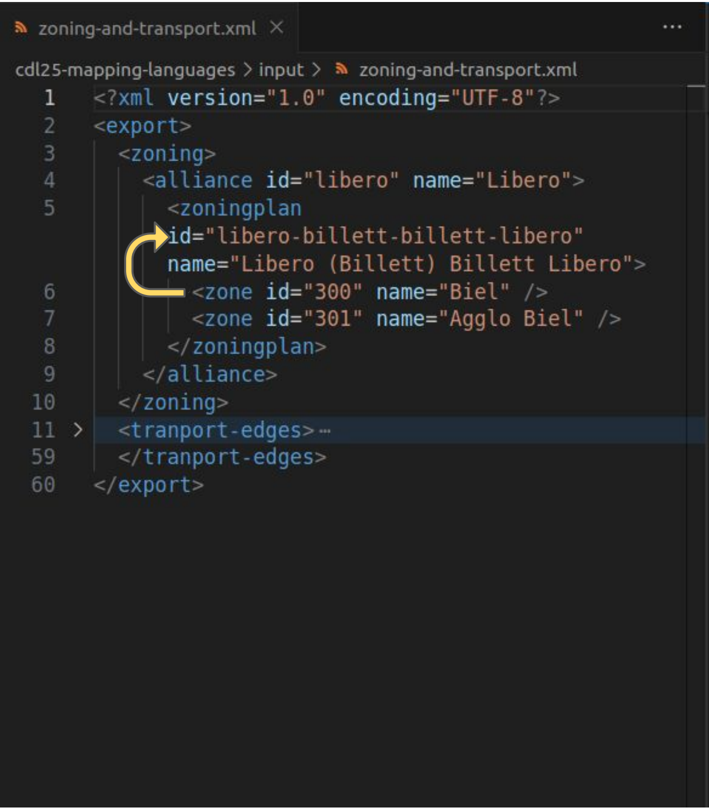
Making links by combining triples maps of different sources (rr:parentTriplesMap)


mapping-zoningplan.rml.ttl
cdl25-mapping-languages > src-gen > mapping-zoningplan.rml.ttl

```
<#Zoningplan>

    rr:predicateObjectMap [
        rr:predicate otd:zone;
        rr:objectMap [
            rr:parentTriplesMap  <#Zone>;
            rr:joinCondition [
                rr:child "@id";
                rr:parent "../@id";
            ]
        ]
    ].
```

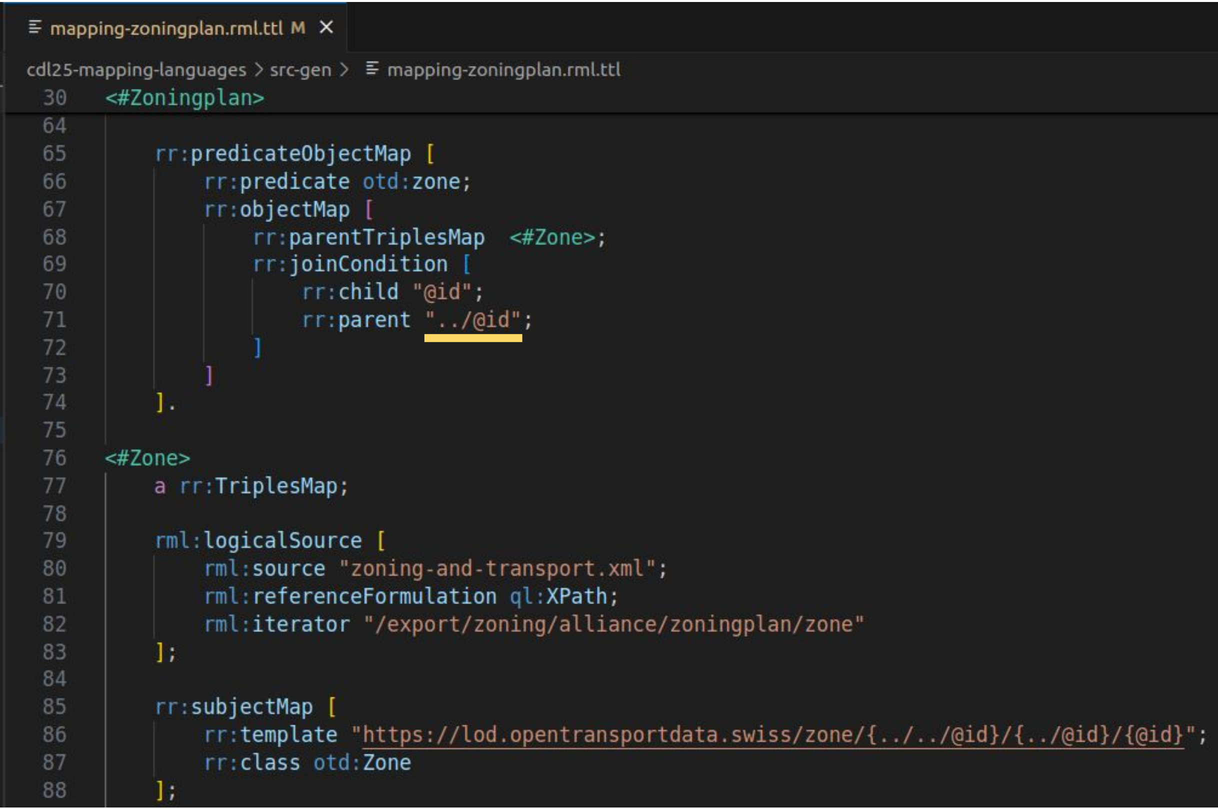
Making links by combining triples maps of different sources (rr:parentTriplesMap)

XML input



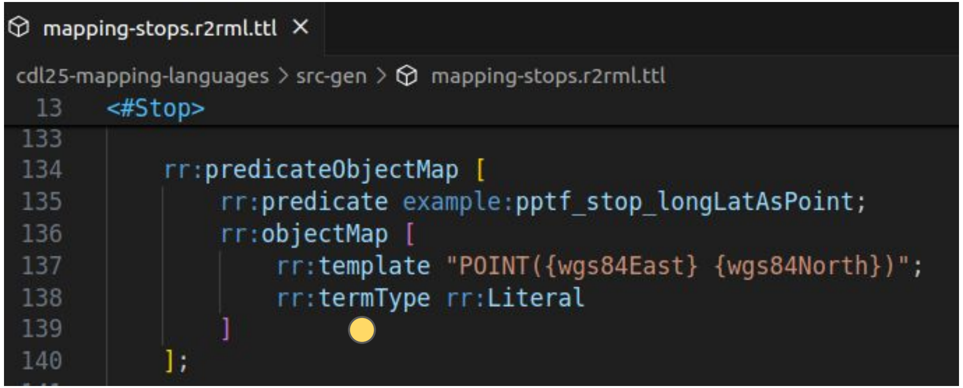
```
<?xml version="1.0" encoding="UTF-8"?>
<export>
  <zoning>
    <alliance id="libero" name="Libero">
      <zoningplan
      id="libero-billett-billett-libero"
      name="Libero (Billett) Billett Libero">
        <zone id="300" name="Biel" />
        <zone id="301" name="Agglo Biel" />
      </zoningplan>
    </alliance>
  </zoning>
  <tranport-edges>⋯
  </tranport-edges>
</export>
```

RML mapping


mapping-zoningplan.rml.ttl M
cdl25-mapping-languages > src-gen > mapping-zoningplan.rml.ttl

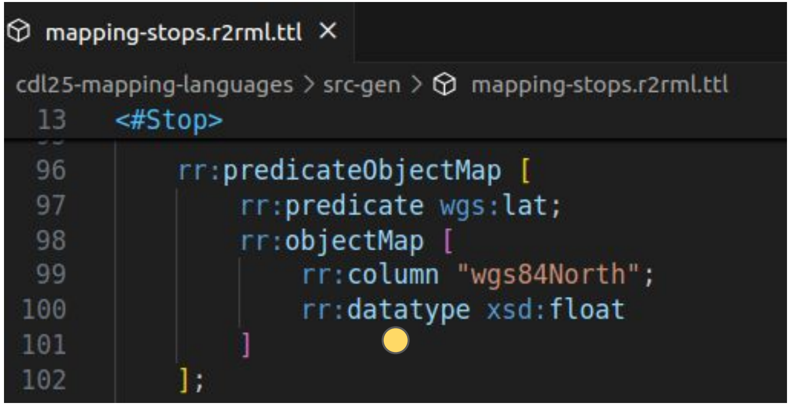
```
<#Zoningplan>

    rr:predicateObjectMap [
        rr:predicate otd:zone;
        rr:objectMap [
            rr:parentTriplesMap  <#Zone>;
            rr:joinCondition [
                rr:child "@id";
                rr:parent "../@id";
            ]
        ]
    ].

<#Zone>
    a rr:TriplesMap;

    rml:logicalSource [
        rml:source "zoning-and-transport.xml";
        rml:referenceFormulation ql:XPath;
        rml:iterator "/export/zoning/alliance/zoningplan/zone"
    ];

    rr:subjectMap [
        rr:template "https://lod.opentransportdata.swiss/zone/{../../@id}/{../@id}/{@id}";
        rr:class otd:Zone
    ];
```

IRI, Literal or Blank Node (rr:termType)


mapping-stops.r2rml.ttl ×
cdl25-mapping-languages > src-gen > mapping-stops.r2rml.ttl

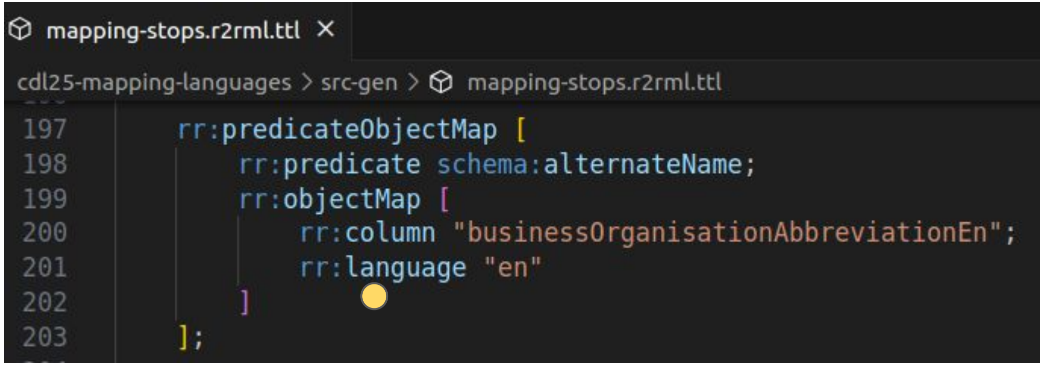
```
<#Stop>

    rr:predicateObjectMap [
        rr:predicate example:pptf_stop_longLatAsPoint;
        rr:objectMap [
            rr:template "POINT({wgs84East} {wgs84North})";
            rr:termType rr:Literal
        ]
    ];
```

Typed Literals (rr:datatype)


mapping-stops.r2rml.ttl ×
cdl25-mapping-languages > src-gen > mapping-stops.r2rml.ttl

```
<#Stop>
    rr:predicateObjectMap [
        rr:predicate wgs:lat;
        rr:objectMap [
            rr:column "wgs84North";
            rr:datatype xsd:float
        ]
    ];
```

Language Tags (rr:language)



```
    rr:predicateObjectMap [
        rr:predicate schema:alternateName;
        rr:objectMap [
            rr:column "businessOrganisationAbbreviationEn";
            rr:language "en"
        ]
    ];
```

XRM

XRM is a textual language and editor, built with the objective to make it **easier to create, refine and maintain R2RML and RML** mappings

XRM supports a subset of R2RML and RML

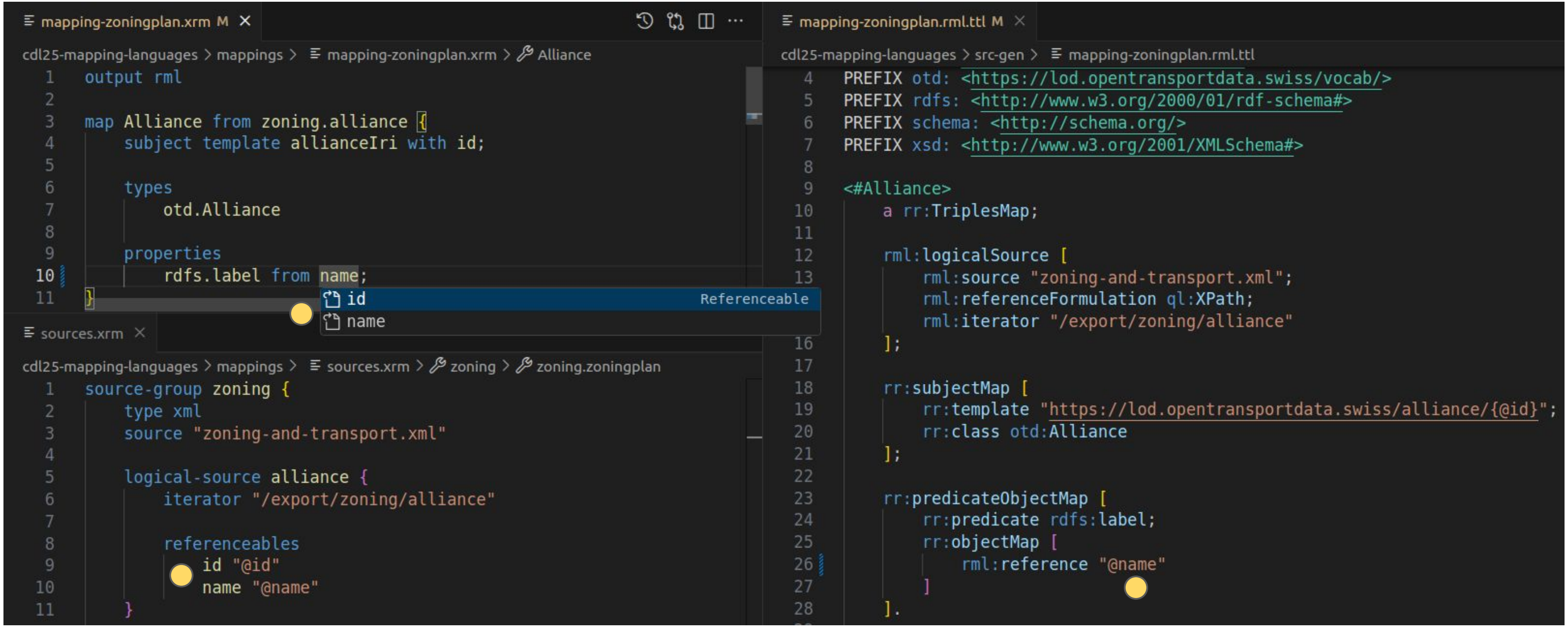
Processing toolchain remains on R2RML / RML

https://github.com/zazuko/expressive-rdf-mapper

XRM

XRM

RML


mapping-zoningplan.xrm M ×
cdl25-mapping-languages > mappings > mapping-zoningplan.xrm > Alliance
id Referenceable
name
sources.xrm ×
cdl25-mapping-languages > mappings > sources.xrm > zoning > zoning.zoningplan
mapping-zoningplan.rml.ttl M ×
cdl25-mapping-languages > src-gen > mapping-zoningplan.rml.ttl

```
output rml

map Alliance from zoning.alliance {
    subject template allianceIri with id;

    types
        otd.Alliance

    properties
        rdfs.label from name;
}
```

```
source-group zoning {
    type xml
    source "zoning-and-transport.xml"

    logical-source alliance {
        iterator "/export/zoning/alliance"

        referenceables
            id "@id"
            name "@name"
    }
```

```
PREFIX otd: <https://lod.opentransportdata.swiss/vocab/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX schema: <http://schema.org/>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

<#Alliance>
    a rr:TriplesMap;

    rml:logicalSource [
        rml:source "zoning-and-transport.xml";
        rml:referenceFormulation ql:XPath;
        rml:iterator "/export/zoning/alliance"
    ];

    rr:subjectMap [
        rr:template "https://lod.opentransportdata.swiss/alliance/{@id}";
        rr:class otd:Alliance
    ];

    rr:predicateObjectMap [
        rr:predicate rdfs:label;
        rr:objectMap [
            rml:reference "@name"
        ]
    ].
```

Summary: R2RML, RML, XRM

R2RML for **relational databases**

RML for **any type of data source**

XRM as **editor language** for R2RML and RML

Same base concept of how mapping directives are described

ETL Pipelines

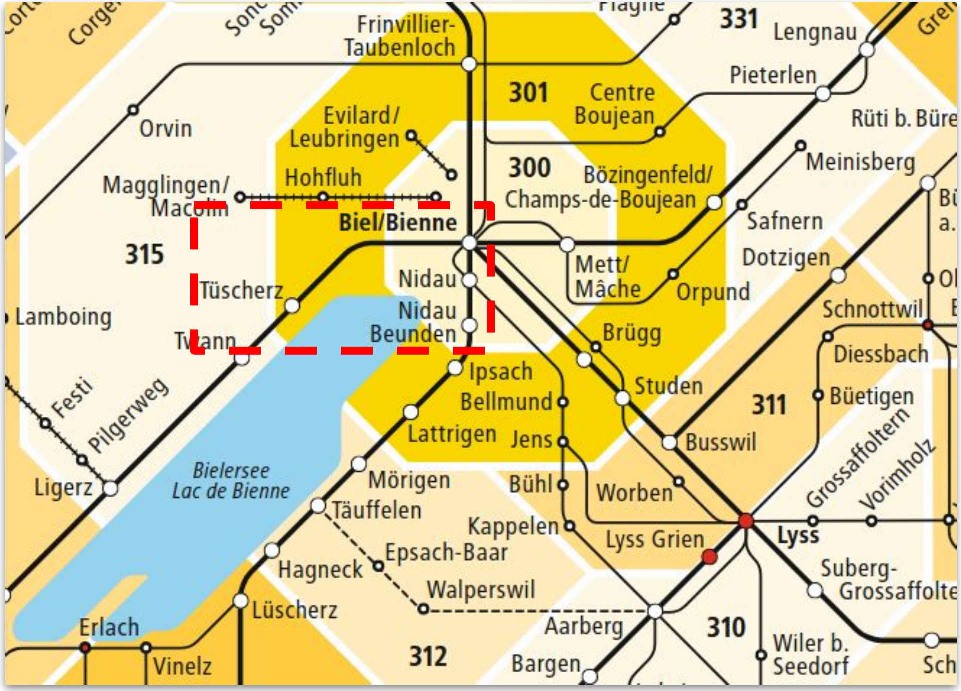
There is more to data transformation than just mapping

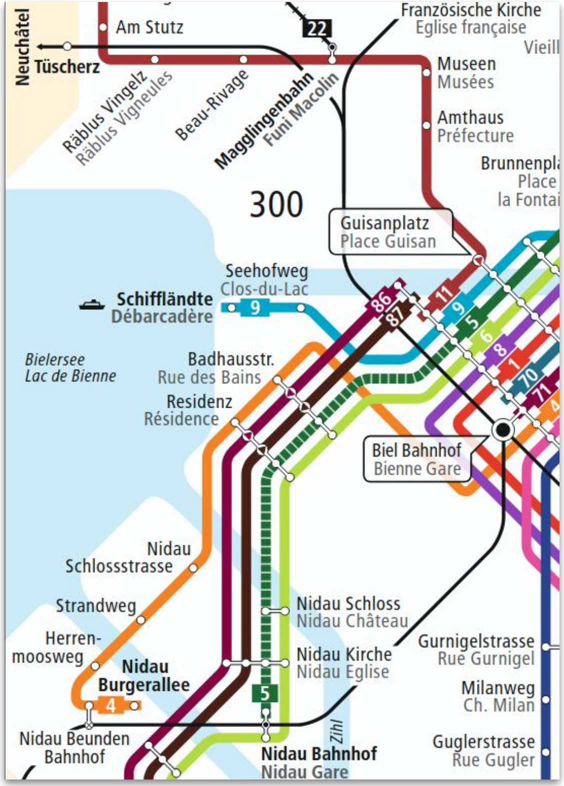
Data pre-processing and post-processing after the mapping step is common

Pipelines: https://github.com/zazuko/expressive-rdf-mapper#tutorials--documentation

Transformation functions: https://github.com/kg-construct

Hands-On - sample data from public transport domain





Hands-On

Example 1

XML file
RML mapping
CARML engine

Example 2

CSV file
R2RML mapping
Ontop VKG system DuckDB

https://github.com/zazuko/cdl25-mapping-languages/

Contact

michael.rauch@zazuko.com

https://www.linkedin.com/in/michael-rauch-mra/