Knowledge Graph Construction using Declarative Mapping Rules

David Chaves-Fraga, Ana Iglesias-Molina, Andrea Cimmino, Oscar Corcho

Ontology Engineering Group
Universidad Politécnica de Madrid, Spain

dchaves@fi.upm.es oeg-upm 01/11/20 ISWC2020

Oscar Corcho

Full professor at UPM and leader of OEG

Ana Iglesias-Molina

PhD Student

David Chaves-Fraga

PhD Student

Andrea Cimmino

Post-Doctoral researcher

https://www.w3.org/community/kg-construct/

The past, the present and... the future

RDB2RDF W3C WG

SPARQL Federation 1.1 W3C Rec.

R2RML W3C Rec.

RML & CSVW

KG construct W3C CG

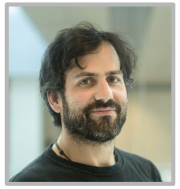
2004 ... 2007 ... 2011 ... 2013 2014 2015 2016 2017 2018 2019 2020 ...

R2O

ODEMapster

SPARQL-DQP

Morph-RDB



kyre

Morph-Streams

Morph-CSV & Helio

RMLC

Mapeathor

WOT Mapping

Morph-Skyline

Morph-GraphQL

eWoT

Madrid-GTFS Benchmark

W3C Group

Language

Benchmark

Tool

External to OEG

References

- Ana Iglesias-Molina, David Chaves-Fraga, Freddy Priyatna, Oscar Corcho: *Towards the definition of a language-independent mapping template for knowledge graph creation*. Proceedings of the Third International Workshop on Capturing Scientific Knowledge co-located with the 10th International Conference on Knowledge Capture (K-CAP 2019), 2019. Online version
- Ana Iglesias-Molina, David Chaves-Fraga, Freddy Priyatna, Oscar Corcho: *Enhancing the Maintainability of the Bio2RDF project Using Declarative Mappings*. 12th International Semantic Web Applications and Tools for Health Care and Life Sciences Conference, 2019. Online version
- David Chaves-Fraga, Freddy Priyatna, Andrea Cimmino, Jhon Toledo, Edna Ruckhaus, & Oscar Corcho (2020). GTFS-Madrid-Bench: A Benchmark for Virtual Knowledge Graph Access in the Transport Domain. *Journal of Web Semantics*, *65*. Online version GitHub
- Oscar Corcho, Freddy Priyatna, David Chaves-Fraga: *Towards a New Generation of Ontology Based Data Access*. Semantic Web Journal, 2020. Online version
- David Chaves-Fraga, Edna Ruckhaus, Freddy Priyatna, Maria-Esther Vidal, Oscar Corcho: *Enhancing OBDA Query Translation over Tabular Data with Morph-CSV*. Under Review at SWJ. Online version
- David Chaves-Fraga, Luis Pozo, Jhon Toledo, Edna Ruckhaus, Oscar Corcho: *Morph-CSV: Virtual Knowledge Graph Access for Tabular Data*. 19th International Semantic Web Conference P&D, 2020.
- Ana Iglesias-Molina, Luis Pozo-Gilo, Daniel Doña, Edna Ruckhaus, David Chaves-Fraga and Oscar Corcho. *Mapeathor: Simplifying the Specification of Declarative Rules for Knowledge Graph Construction*. 19th International Semantic Web Conference P&D, 2020.

Schedule:

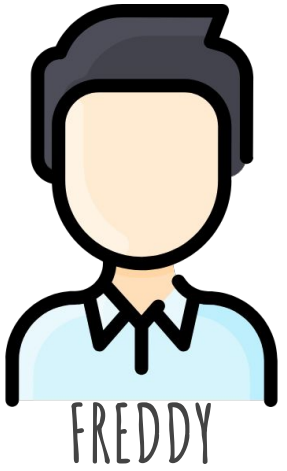
- 16:00 - 16:10 : Greetings
- 16:10 - 16:30 : Introduction
- 16:30 - 17:10 : Mapeathor
- 17:10 - 17:50 : Morph-CSV
- 17:50 - 18:05 : Break
- 18:05 - 18:45 : Helio
- 18:45 - 19:00 : Recapitulation & discussion

Introduction

Oscar Corcho
ocorcho@fi.upm.es
@ocorcho

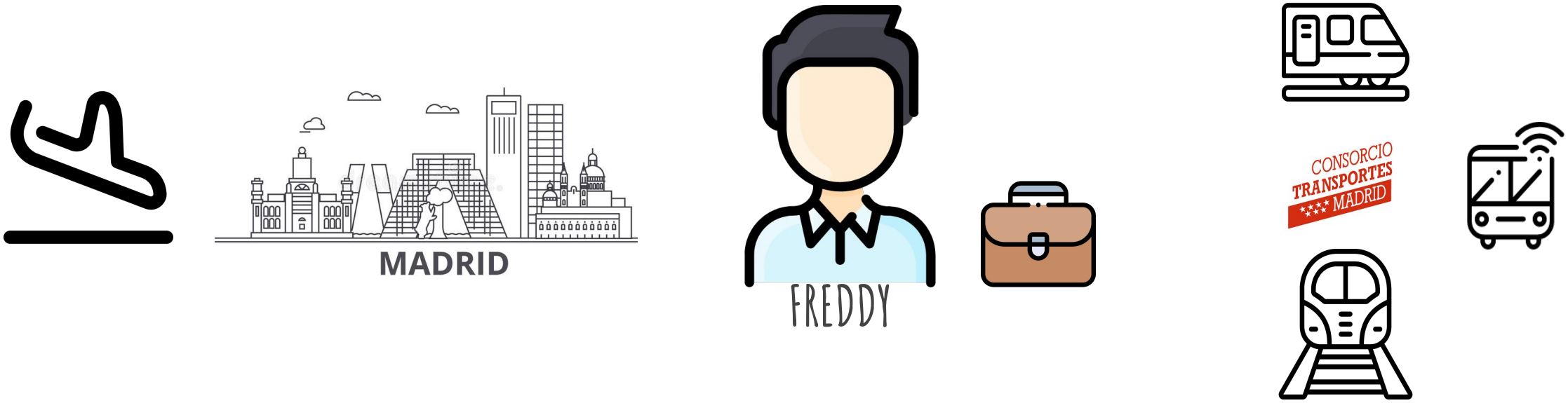
16:10-16:30

Let’s start with a story...



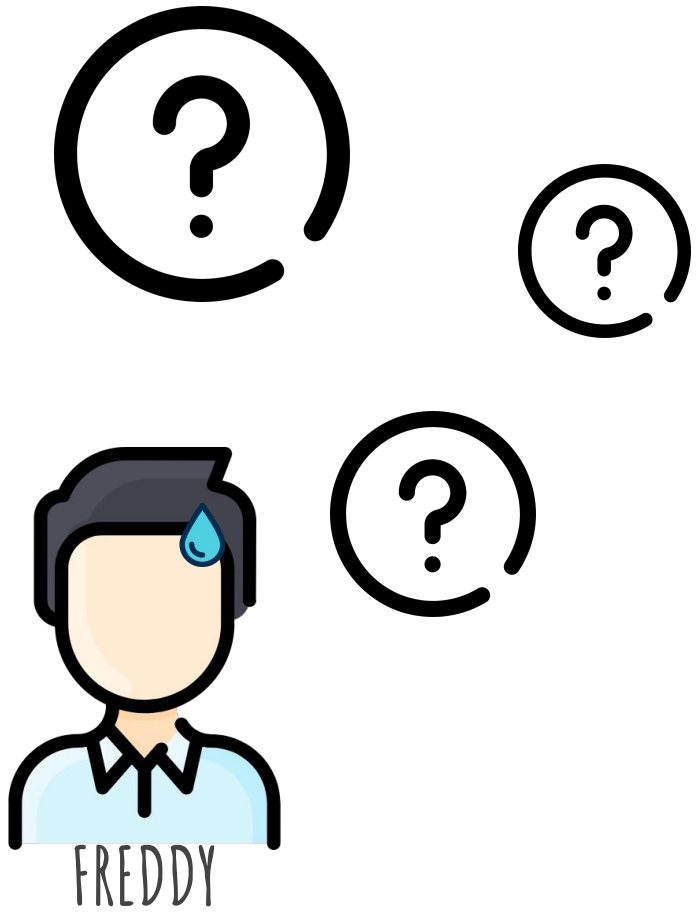
... with Freddy as our virtual guide

Freddy arrives in Madrid, and he is employed as a knowledge scientist in charge of handling Madrid's **transport data**

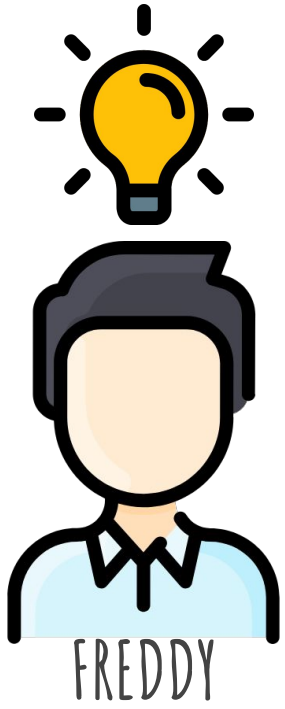


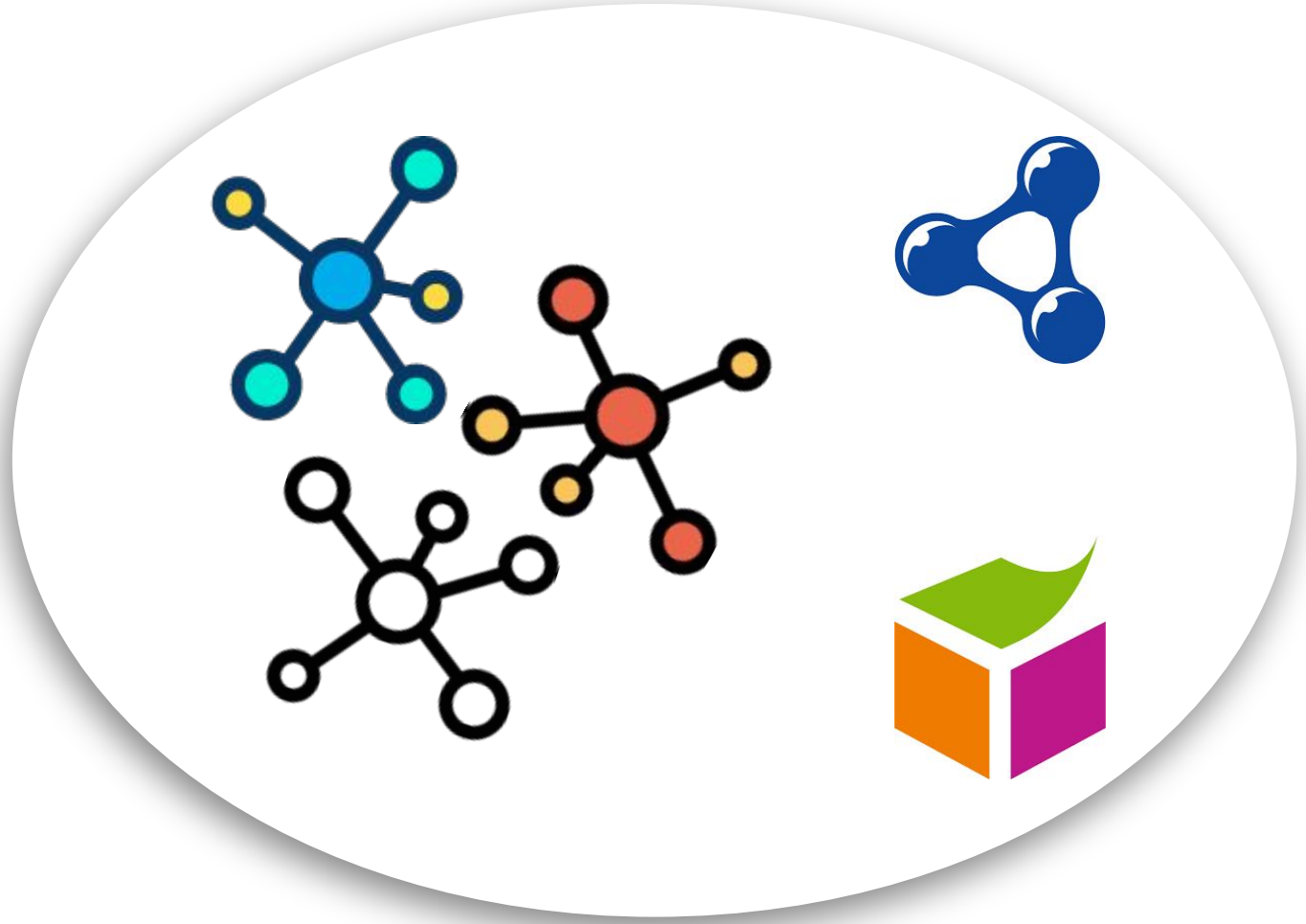
However, real transport data is **tricky**:

- **Heterogeneity**
- **Complexity**
- Data **not clean** and **normalized**
- Values, such as dates, have to be **correctly interpreted**
- Has to be **understandable** by anyone



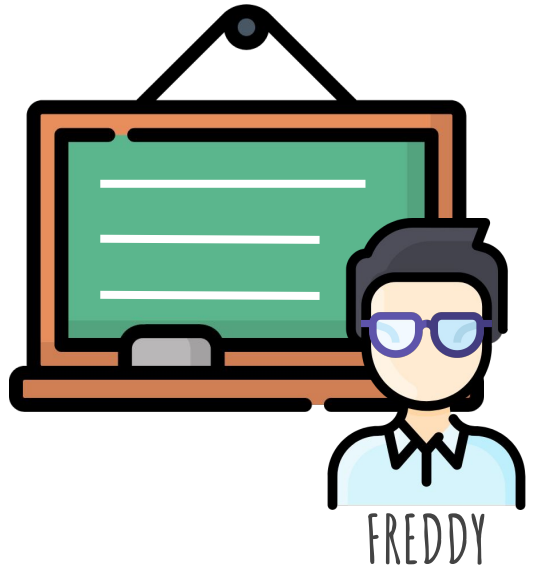
So he decided to build a **Knowledge Graph**, the solution to represent clearly the heterogeneous and make it **clear**, **accessible** and **queriable**



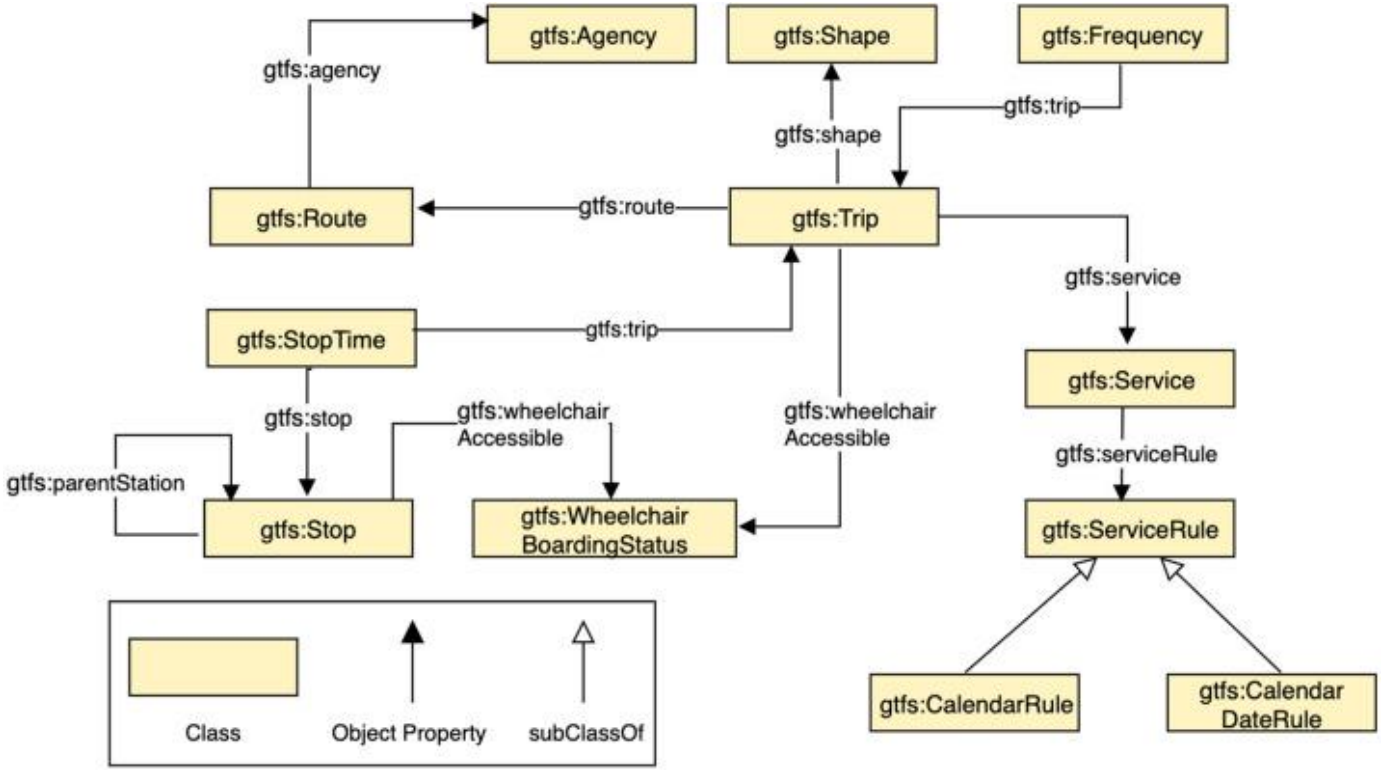


The **requirements** needed for this data integration pipeline are:

- **Ontology** that models the transport domain
- **Standard declarative mapping rules**:
  - Flexibility
  - Adaptability
  - Maintainability
  - Readability
  - Reproducibility
- Ensure **materialized** but also **virtual** views
- **Avoid** at maximum ad-hoc and **manual** steps
- **Efficient** generation
- Improved **publication** of the KG

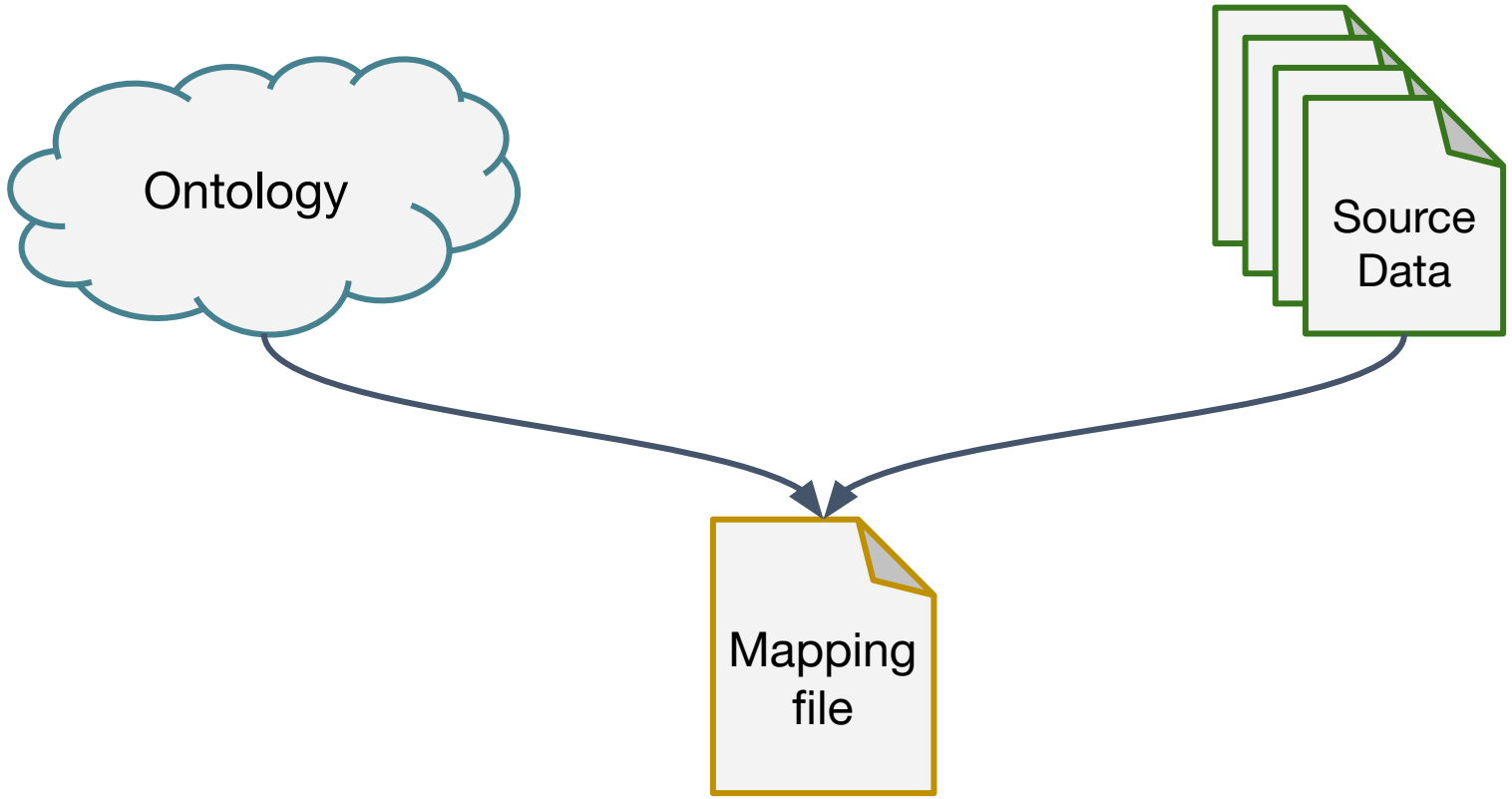


He has been given the data, so it's time to choose a suitable ontology: the **General Transit Feed Specification (GTFS)**[1]

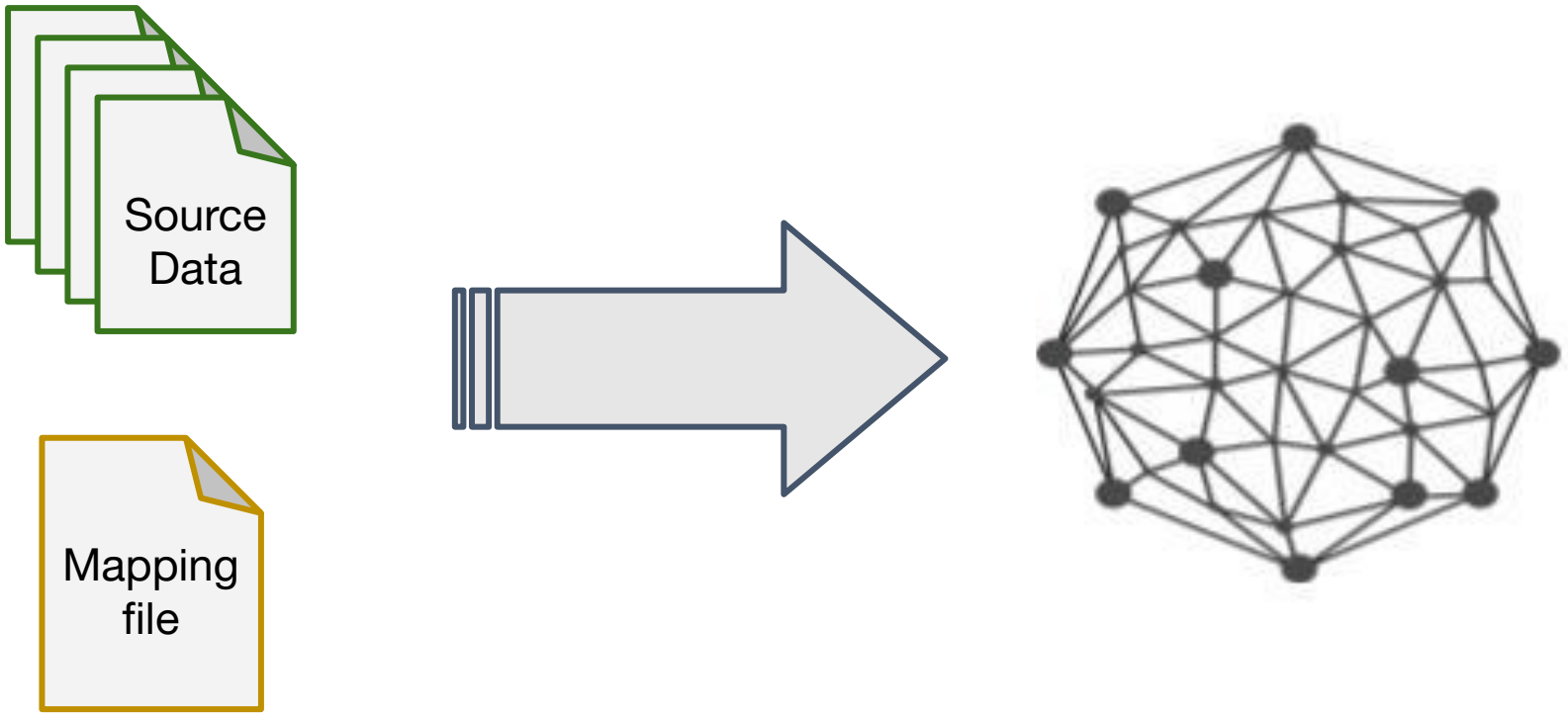


[1] https://lov.linkeddata.es/dataset/lov/vocabs/gtfs

First step: establish and create the **rules** to transform the data following the ontology's schema in **mapping files**

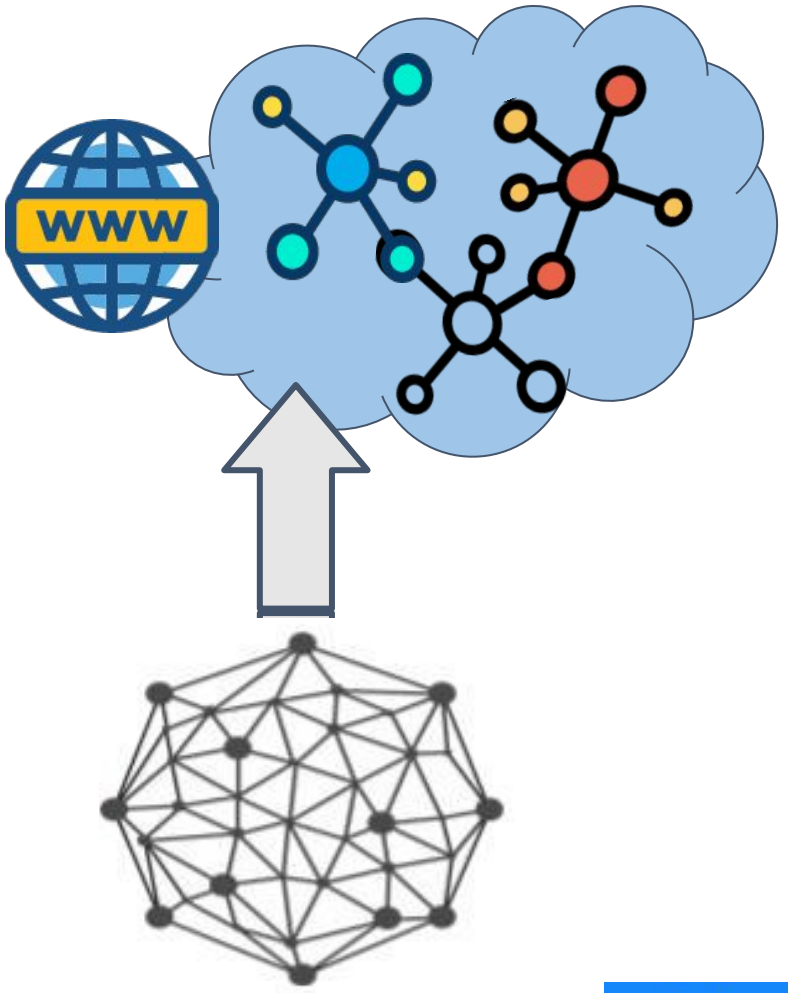


Then it's time to **transform** the data into (virtual) **Knowledge Graph**

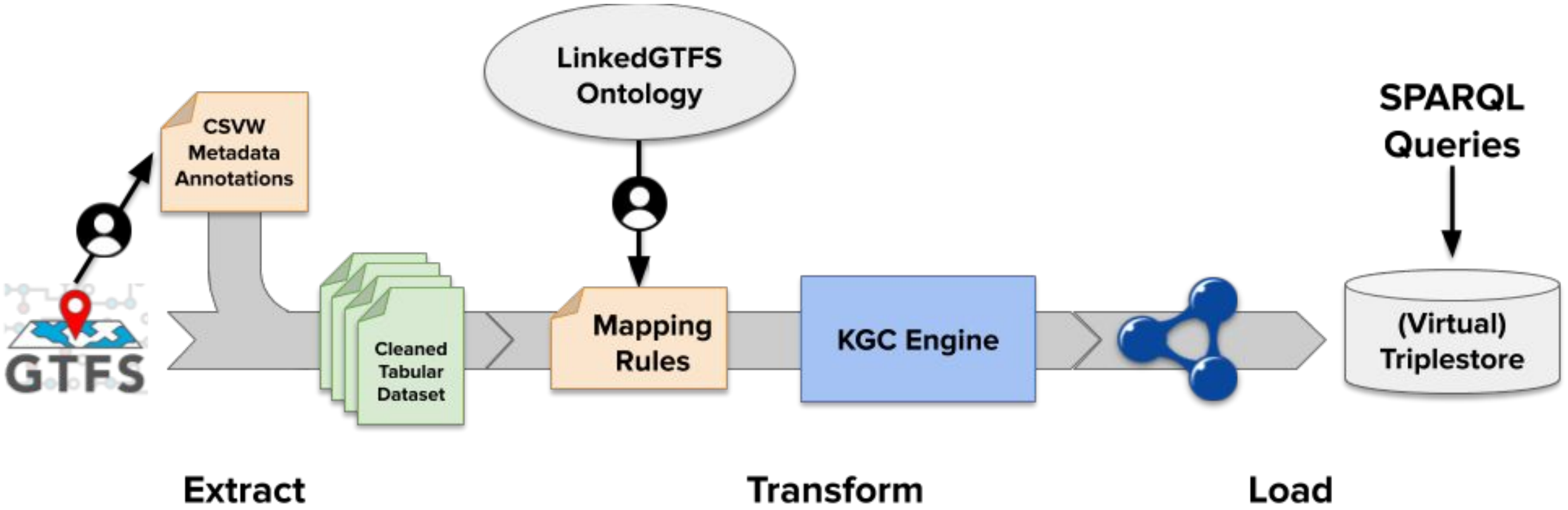


Finally, the data is ready to be **published**, making sure data is:

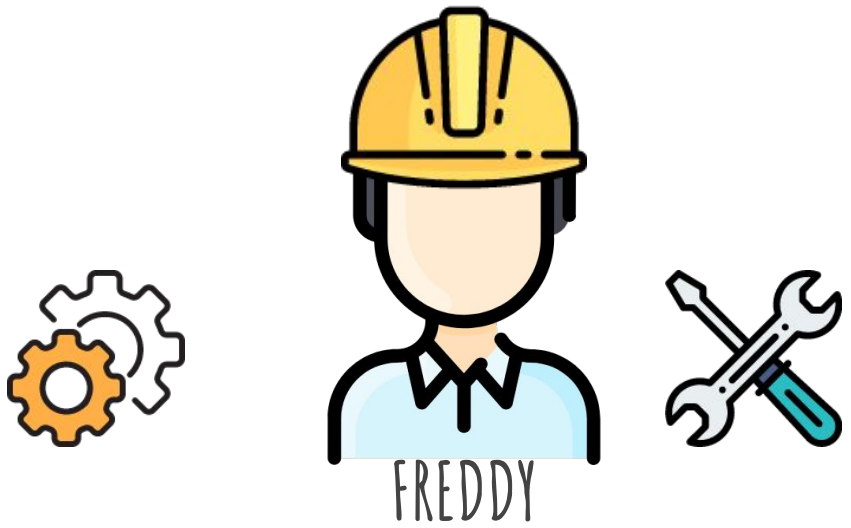
- Accessible
- Consumable & queriable
- Understandable
  - by humans
  - by machines



General overview



Now, let's get technical, we'll show you how it's done...



https://github.com/oeg-upm/kgc-tutorial-iswc2020

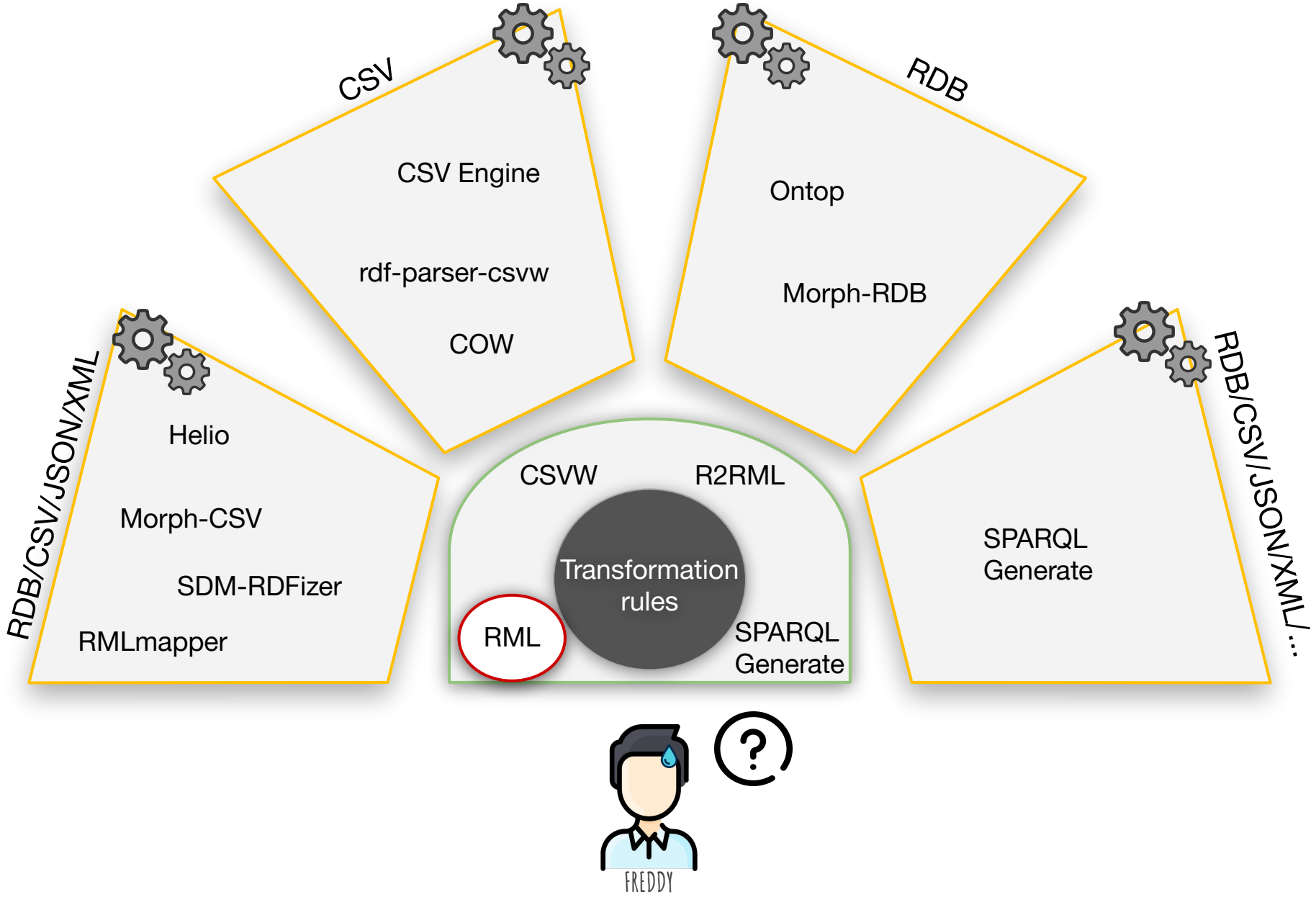
Mapeathor

Ana Iglesias-Molina

ana.iglesiasm@upm.es
@_aieme

16:30-17:10

Diversity of mapping languages



```
<STOPTIMES>
  rml:logicalSource [
     rml:source "data/STOP_TIMES.csv" ;
     rml:referenceFormulation ql:CSV ;
  ];
  rr:subjectMap [
     rr:class gtfs:StopTime;
     rr:template "http://trans.linkeddata.es/{trip_id}-{stop_id}";
  ];
  rr:predicateObjectMap [
     rr:predicateMap    [ rr:constant gtfs:arrivalTime ];
     rr:objectMap    [ rml:reference "arrival_time" ];
  ];
  rr:predicateObjectMap [
     rr:predicateMap    [ rr:constant gtfs:departureTime ];
     rr:objectMap    [ rml:reference "departure_time" ];
  ];
```

Triples Map

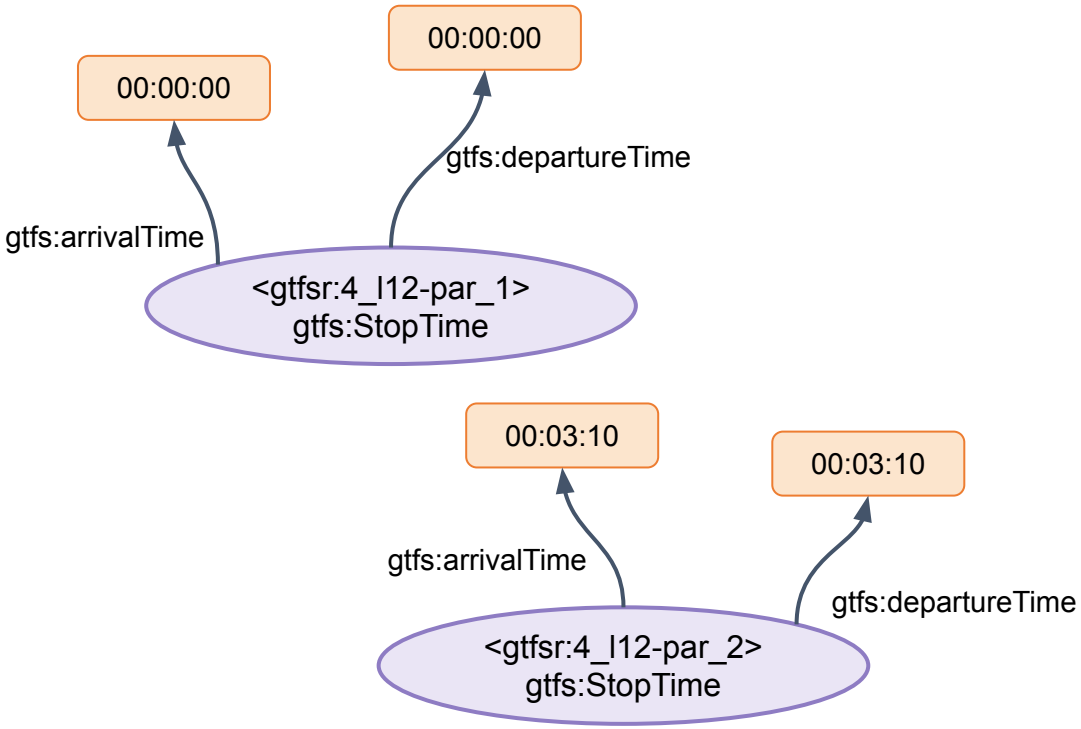
STOP_TIMES.csv

trip_id	arrival_time	departure_time	stop_id
4_I12	00:00:00	00:00:00	par_1
4_I12	00:03:10	00:03:10	par_2

```
gtfsr:4_I12-par_1    a                    ex:StopTime .
gtfsr:4_I12-par_1    gtfs:arrivalTime     “00:00:00” .
gtfsr:4_I12-par_1    gtfs:departureTime   “00:00:00” .
gtfsr:4_I12-par_2    a                    ex:StopTime .
gtfsr:4_I12-par_2    gtfs:arrivalTime     “00:03:10” .
gtfsr:4_I12-par_2    gtfs:departureTime   “00:03:10” .
```

Mapping structure (RML)

```
<STOPTIMES>
 rml:logicalSource [
    rml:source "data/STOP_TIMES.csv" ;
    rml:referenceFormulation ql:CSV ;
 ];
 rr:subjectMap [
    rr:class gtfs:StopTime;
    rr:template "http://trans.linkeddata.es/{trip_id}-{stop_id}";
 ];
 rr:predicateObjectMap [
    rr:predicateMap    [ rr:constant gtfs:arrivalTime ];
    rr:objectMap    [ rml:reference "arrival_time" ];
 ];
 rr:predicateObjectMap [
    rr:predicateMap    [ rr:constant gtfs:departureTime ];
    rr:objectMap    [ rml:reference "departure_time" ];
 ];
```



gtfsr:4_l12-par_1	a	ex:StopTime .
gtfsr:4_l12-par_1	gtfs:arrivalTime	“00:00:00” .
gtfsr:4_l12-par_1	gtfs:departureTime	“00:00:00” .
gtfsr:4_l12-par_2	a	ex:StopTime .
gtfsr:4_l12-par_2	gtfs:arrivalTime	“00:03:10” .
gtfsr:4_l12-par_2	gtfs:departureTime	“00:03:10” .

Mapping structure (RML)

```
<STOPTIMES>
  rml:logicalSource [
     rml:source "data/STOP_TIMES.csv" ;
     rml:referenceFormulation ql:CSV ;
  ];
  rr:subjectMap [
     rr:class gtfs:StopTime;
     rr:template "http://trans.linkeddata.es/{trip_id}-{stop_id}";
  ];
  rr:predicateObjectMap [
     rr:predicateMap    [ rr:constant gtfs:arrivalTime ];
     rr:objectMap    [ rml:reference "arrival_time" ];
  ];
  rr:predicateObjectMap [
     rr:predicateMap    [ rr:constant gtfs:departureTime ];
     rr:objectMap    [ rml:reference "departure_time" ];
  ];
  rr:predicateObjectMap [
     rr:predicateMap    [ rr:constant gtfs:stop ];
     rr:objectMap    [ rr:parentTriplesMap <STOPS>;
       rr:joinCondition [ rr:child "stop_id"; rr:parent "stop_id"; ];
     ];
  ];
```

STOP_TIMES.csv

trip_id	arrival_time	departure_time	stop_id
4_I12	00:00:00	00:00:00	par_1
4_I12	00:03:10	00:03:10	par_2

STOPS.csv

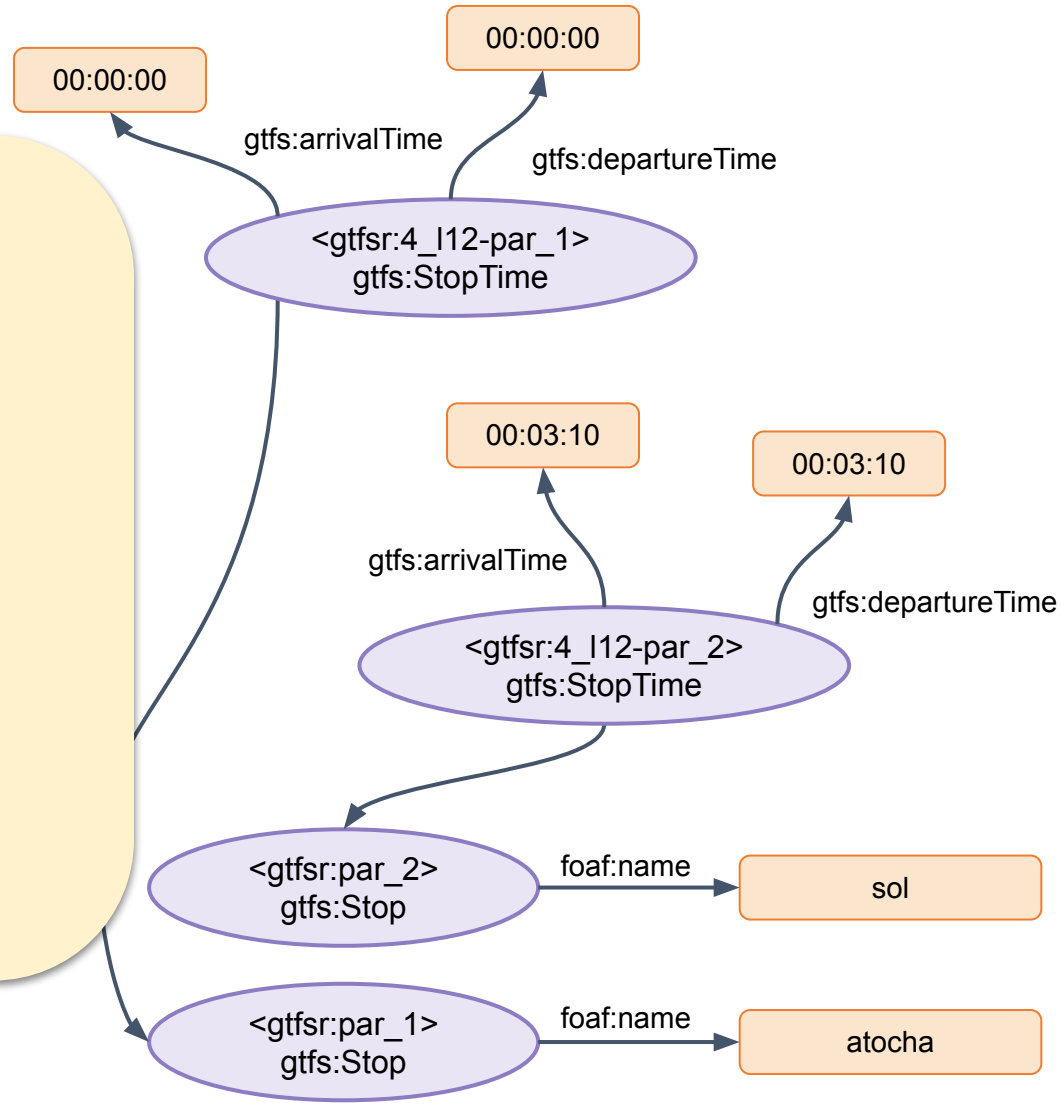
stop_id	stop_name
par_1	atocha
par_2	sol

```
<STOPS>
  rml:logicalSource [
     rml:source "data/STOP.csv" ;
     rml:referenceFormulation ql:CSV ;
  ];
  rr:subjectMap [
     rr:class ex:Sport;
     rr:template "http://trans.linkeddata.es/{stop_id}";
  ];
  rr:predicateObjectMap [
     rr:predicateMap    [ rr:constant foaf:name ];
     rr:objectMap    [ rml:reference "stop_name" ];
  ];
```

Join condition

Mapping structure (RML)

```
gtfsr:4_l12-par_1        a                    ex:StopTime .
gtfsr:4_l12-par_1        gtfs:arrivalTime     “00:00:00” .
gtfsr:4_l12-par_1        gtfs:departureTime   “00:00:00” .
gtfsr:4_l12-par_1        gtfs:stop            gtfsr:par_1 .
gtfsr:4_l12-par_2        a                    ex:StopTime .
gtfsr:4_l12-par_2        gtfs:arrivalTime     “00:03:10” .
gtfsr:4_l12-par_2        gtfs:departureTime   “00:03:10” .
gtfsr:4_l12-par_2        gtfs:stop            gtfsr:par_2 .

gtfsr:par_1              a                    gtfs:Stop .
gtfsr:par_1              foaf:name            “atocha” .
gtfsr:par_2              a                    ex:Sport .
gtfsr:par_2              foaf:name            “sol” .
```



STOPS.csv	
stop_id	**stop_name**
par_1	atocha
par_2	sol

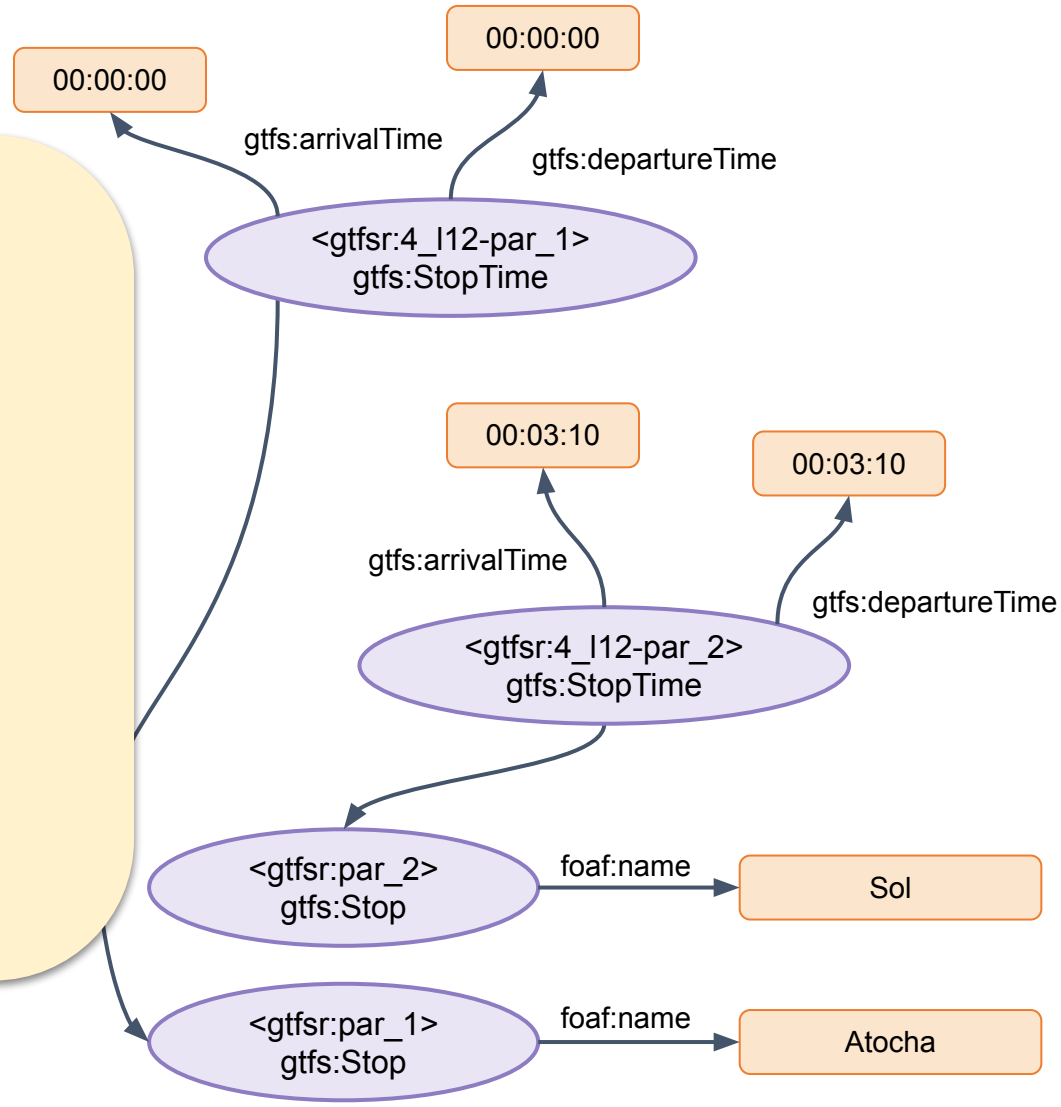
Function in Triple Map

```
<STOPS>
 rml:logicalSource [
    rml:source "data/STOP.csv" ;
    rml:referenceFormulation ql:CSV ;
 ];
 rr:subjectMap [
    rr:class ex:Stop
    rr:template "http://trans.linkeddata.es/{stop_id}";
 ];
 rr:predicateObjectMap [
    rr:predicateMap    [ rr:constant foaf:name ];
    rr:objectMap    <TITLE_FUNC> ;
 ];
```

```
<TITLE_FUNC>  a fnml:FunctionTermMap ;
 fnml:functionValue [
   rml:logicalSource [
      rml:source "data/STOPS.csv" ;
      rml:referenceFormulation ql:CSV ;
   ];
   rr:predicateObjectMap [
      rr:predicateMap    fno:executes;
      rr:objectMap    [ rml:constant grel:toTitleCase ];
   ];
   rr:predicateObjectMap [
      rr:predicateMap    grel:valueParam1;
      rr:objectMap    [ rml:reference “stop_name” ];
   ];
 ];
.
```

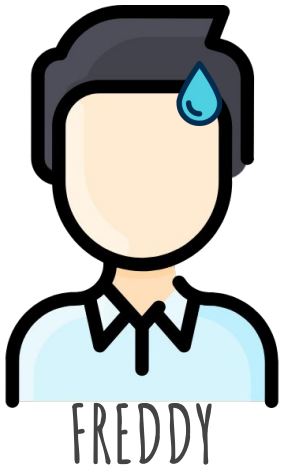
Mapping structure (RML)

```
gtfsr:4_l12-par_1      a                    ex:StopTime .
gtfsr:4_l12-par_1      gtfs:arrivalTime     “00:00:00” .
gtfsr:4_l12-par_1      gtfs:departureTime   “00:00:00” .
gtfsr:4_l12-par_1      gtfs:stop            gtfsr:par_1 .
gtfsr:4_l12-par_2      a                    ex:StopTime .
gtfsr:4_l12-par_2      gtfs:arrivalTime     “00:03:10” .
gtfsr:4_l12-par_2      gtfs:departureTime   “00:03:10” .
gtfsr:4_l12-par_2      gtfs:stop            gtfsr:par_2 .

gtfsr:par_1            a                    gtfs:Stop .
gtfsr:par_1            foaf:name            “Atocha” .
gtfsr:par_2            a                    ex:Sport .
gtfsr:par_2            foaf:name            “Sol” .
```



Freddy is confused

Is there no way of writing this more easily?



Spreadsheets as mappings

- Gathering and declaration of mapping rules in spreadsheets
- Each sheet contains an essential element describing the data:
  - Prefix
  - Source data
  - Subject
  - Predicate-Object
  - Functions

Spreadsheets as mappings

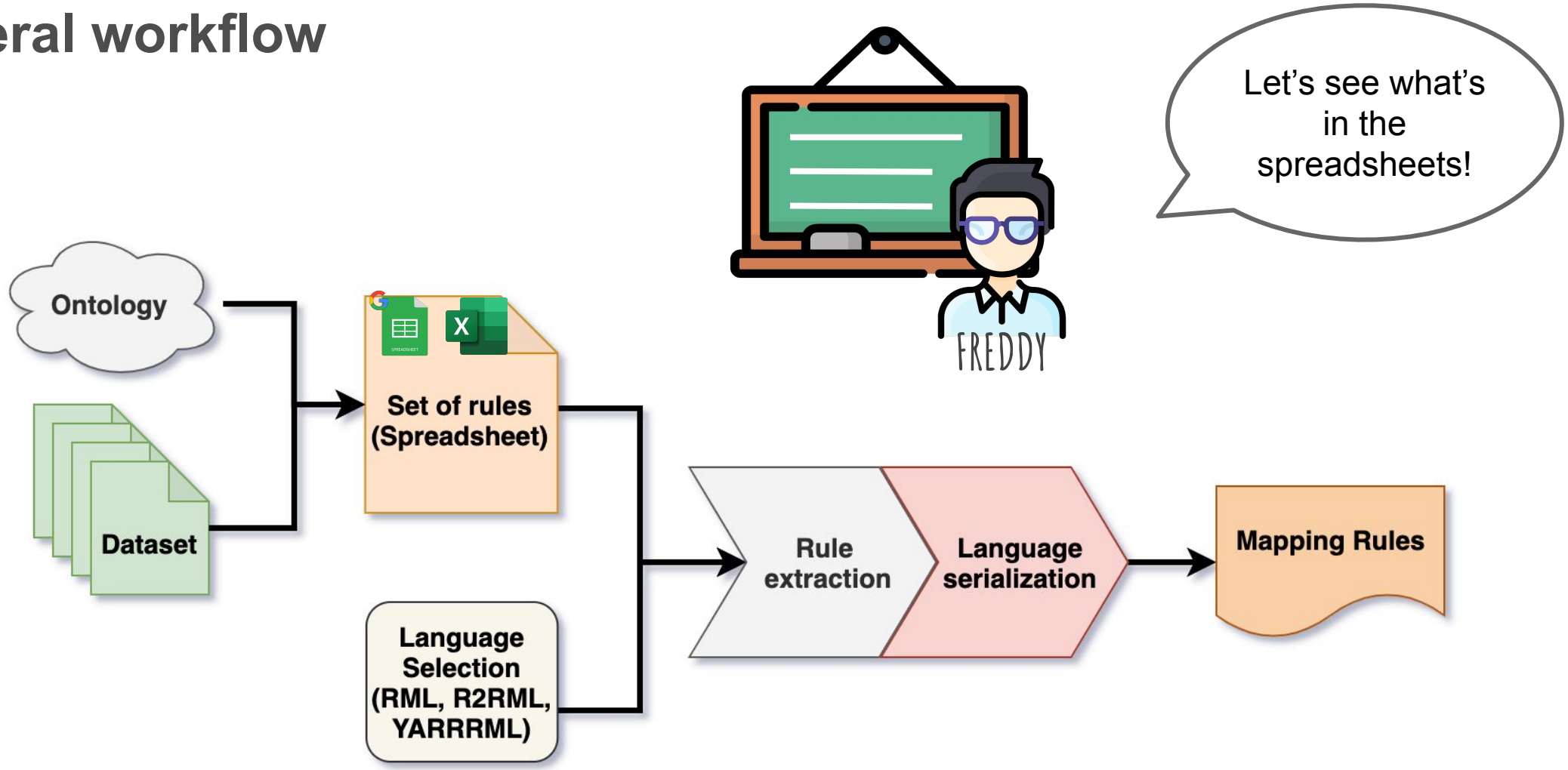
- **Objective**: language-independent, no need to know a mapping language
- **Target user**: Non mapping experts
- **Advantages**: Improves rule visualization, enables using functions of spreadsheets, guides the writing, can be translated into different mapping languages

Implementation: Mapeathor

- Simple spreadsheet parser able to transform Google spreadsheets and Excel into two mapping languages: R2RML and RML (in Turtle and YARRRML representation)
- Available as web service[1,2] and CLI[3,4].

[1] https://morph.oeg.fi.upm.es/tool/mapeathor
[2] https://morph.oeg.fi.upm.es/tool/mapeathor/swagger/
[3] https://pypi.org/project/mapeathor/
[4] https://github.com/oeg-upm/Mapeathor

General workflow



Prefix Subject Source Predicate_Object Function

Prefix	URI
rr	http://www.w3.org/ns/r2rml#
gtfs	http://vocab.gtfs.org/terms#
rml	http://semweb.mmlab.be/ns/rml#
fno	https://w3id.org/function/ontology#

```
@prefix rr: <http://www.w3.org/ns/r2rml#> .
@prefix gtfs: <http://vocab.gtfs.org/terms#> .
@prefix rml: <http://semweb.mmlab.be/ns/rml#> .
@prefix fno: <http://w3id.org/function/ontology#> .

<STOPS>
  rml:logicalSource [
     rml:source "data/STOP.csv" ;
     rml:referenceFormulation ql:CSV ;
  ];
  rr:subjectMap [
     rr:class gtfs:Stop;
     rr:template "http://trans.linkeddata.es/{stop_id}";
  ];
  rr:predicateObjectMap [
     rr:predicateMap    [ rr:constant foaf:name ];
     rr:objectMap    <TITLE_FUNC> ;
  ]; .
```

```
@prefix rr: <http://www.w3.org/ns/r2rml#> .
@prefix gtfs: <http://vocab.gtfs.org/terms#> .
@prefix rml: <http://semweb.mmlab.be/ns/rml#> .
@prefix fno: <http://w3id.org/function/ontology#> .

<STOPTIMES>
  rml:logicalSource [
     rml:source "data/STOP_TIMES.csv" ;
     rml:referenceFormulation ql:CSV ;
  ];
  rr:subjectMap [
     rr:class gtfs:StopTime;
     rr:template "http://trans.linkeddata.es/{trip_id}-{stop_id}";
  ];
  rr:predicateObjectMap [
     rr:predicateMap    [ rr:constant gtfs:arrivalTime ];
     rr:objectMap    [ rml:reference "arrival_time" ];
  ];
  rr:predicateObjectMap [
     rr:predicateMap    [ rr:constant gtfs:departureTime ];
     rr:objectMap    [ rml:reference "departure_time" ];
  ];
  rr:predicateObjectMap [
     rr:predicateMap    [ rr:constant gtfs:stop ];
     rr:objectMap    [ rr:parentTriplesMap <STOPS>;
       rr:joinCondition [ rr:child "stop_id"; rr:parent "stop_id"; ];
     ];
  ]; .
```

Prefix **Subject** Source Predicate_Object Function

ID	Class	URI
STOPTIMES	gtfs:StopTime	http://trans.linkeddata.es/{trip_id}-{stop_id}
STOPS	gtfs:Stop	http://trans.linkeddata.es/{stop_id}

```
@prefix rr: <http://www.w3.org/ns/r2rml#> .
@prefix gtfs: <http://vocab.gtfs.org/terms#> .
@prefix rml: <http://semweb.mmlab.be/ns/rml#> .
@prefix fno: <http://w3id.org/function/ontology#> .

<STOPS>
  rml:logicalSource [
     rml:source "data/STOP.csv" ;
     rml:referenceFormulation ql:CSV ;
  ];
  rr:subjectMap [
     rr:class gtfs:Stop;
     rr:template "http://trans.linkeddata.es/{stop_id}";
  ];
  rr:predicateObjectMap [
     rr:predicateMap    [ rr:constant foaf:name ];
     rr:objectMap    <TITLE_FUNC> ;
  ]; .
```

```
@prefix rr: <http://www.w3.org/ns/r2rml#> .
@prefix gtfs: <http://vocab.gtfs.org/terms#> .
@prefix rml: <http://semweb.mmlab.be/ns/rml#> .
@prefix fno: <http://w3id.org/function/ontology#> .

<STOPTIMES>
  rml:logicalSource [
     rml:source "data/STOP_TIMES.csv" ;
     rml:referenceFormulation ql:CSV ;
  ];
  rr:subjectMap [
     rr:class gtfs:StopTime;
     rr:template "http://trans.linkeddata.es/{trip_id}-{stop_id}";
  ];
  rr:predicateObjectMap [
     rr:predicateMap    [ rr:constant gtfs:arrivalTime ];
     rr:objectMap    [ rml:reference "arrival_time" ];
  ];
  rr:predicateObjectMap [
     rr:predicateMap    [ rr:constant gtfs:departureTime ];
     rr:objectMap    [ rml:reference "departure_time" ];
  ];
  rr:predicateObjectMap [
     rr:predicateMap    [ rr:constant gtfs:stop ];
     rr:objectMap    [ rr:parentTriplesMap <STOPS>;
       rr:joinCondition [ rr:child "stop_id"; rr:parent "stop_id"; ];
     ];
  ]; .
```

Prefix Subject **Source** Predicate_Object Function

ID	Feature	Value
STOPTIMES	source	data/STOP_TIMES.csv
STOPTIMES	format	CSV
STOPS	source	data/STOPS.csv
STOPS	format	CSV

```
@prefix rr: <http://www.w3.org/ns/r2rml#> .
@prefix gtfs: <http://vocab.gtfs.org/terms#> .
@prefix rml: <http://semweb.mmlab.be/ns/rml#> .
@prefix fno: <http://w3id.org/function/ontology#> .

<STOPS>
 rml:logicalSource [
    rml:source "data/STOP.csv" ;
    rml:referenceFormulation ql:CSV ;
 ];
 rr:subjectMap [
    rr:class egtfs:Stop;
    rr:template "http://trans.linkeddata.es/{stop_id}";
 ];
 rr:predicateObjectMap [
    rr:predicateMap    [ rr:constant foaf:name ];
    rr:objectMap    <TITLE_FUNC> ;
 ]; .
```

```
@prefix rr: <http://www.w3.org/ns/r2rml#> .
@prefix gtfs: <http://vocab.gtfs.org/terms#> .
@prefix rml: <http://semweb.mmlab.be/ns/rml#> .
@prefix fno: <http://w3id.org/function/ontology#> .

<STOPTIMES>
 rml:logicalSource [
    rml:source "data/STOP_TIMES.csv" ;
    rml:referenceFormulation ql:CSV ;
 ];
 rr:subjectMap [
    rr:class gtfs:StopTime;
    rr:template "http://trans.linkeddata.es/{trip_id}-{stop_id}";
 ];
 rr:predicateObjectMap [
    rr:predicateMap    [ rr:constant gtfs:arrivalTime ];
    rr:objectMap    [ rml:reference "arrival_time" ];
 ];
 rr:predicateObjectMap [
    rr:predicateMap    [ rr:constant gtfs:departureTime ];
    rr:objectMap    [ rml:reference "departure_time" ];
 ];
 rr:predicateObjectMap [
    rr:predicateMap    [ rr:constant gtfs:stop ];
    rr:objectMap    [ rr:parentTriplesMap <STOPS>;
      rr:joinCondition [ rr:child "stop_id"; rr:parent "stop_id"; ];
    ];
 ]; .
```

Prefix Subject Source **Predicate_Object** Function

ID	Predicate	Object	DataType	ReferenceID	InnerRef	OuterRef
STOPTIMES	gtfs:arrivalTime	{arrival_time}	duration			
STOPTIMES	gtfs:departureTime	{departure_time}	duration			
STOPTIMES	ex:sport			stops	{stop_id}	{stop_id}
STOPS	foaf:name	<TITLE_FUNC>	string			

```
@prefix rr: <http://www.w3.org/ns/r2rml#> .
@prefix gtfs: <http://vocab.gtfs.org/terms#> .
@prefix rml: <http://semweb.mmlab.be/ns/rml#> .
@prefix fno: <http://w3id.org/function/ontology#> .

<STOPS>
 rml:logicalSource [
    rml:source "data/STOP.csv" ;
    rml:referenceFormulation ql:CSV ;
 ];
 rr:subjectMap [
    rr:class gtfs:Stop;
    rr:template "http://trans.linkeddata.es/{stop_id}";
 ];
 rr:predicateObjectMap [
    rr:predicateMap    [ rr:constant foaf:name ];
    rr:objectMap    <TITLE_FUNC> ;
 ]; .
```

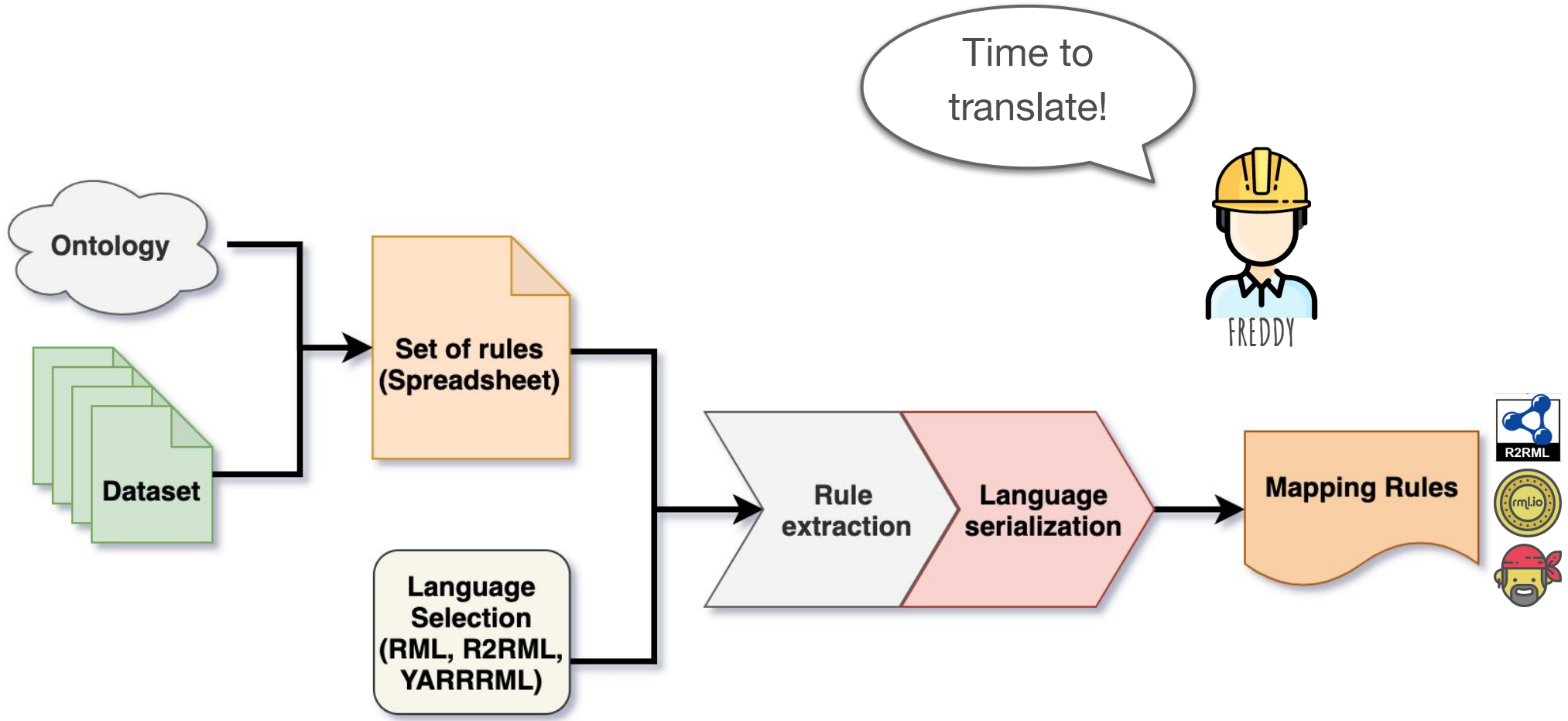
```
@prefix rr: <http://www.w3.org/ns/r2rml#> .
@prefix gtfs: <http://vocab.gtfs.org/terms#> .
rml: <http://semweb.mmlab.be/ns/rml#> .
fno: <http://w3id.org/function/ontology#> .

IMES>
calSource [
ource "data/STOP_TIMES.csv" ;
referenceFormulation ql:CSV ;
];
rr:subjectMap [
   rr:class gtfs:StopTime;
   rr:template "http://trans.linkeddata.es/{trip_id}-{stop_id}";
];
rr:predicateObjectMap [
   rr:predicateMap    [ rr:constant gtfs:arrivalTime ];
   rr:objectMap    [ rml:reference "arrival_time" ];
];
rr:predicateObjectMap [
   rr:predicateMap    [ rr:constant gtfs:departureTime ];
   rr:objectMap    [ rml:reference "departure_time" ];
];
rr:predicateObjectMap [
   rr:predicateMap    [ rr:constant gtfs:stop ];
   rr:objectMap    [ rr:parentTriplesMap <STOPS>;
     rr:joinCondition [ rr:child "stop_id"; rr:parent "stop_id"; ];
   ];
 ] .
```

Prefix Subject Source Predicate_Object **Function**

FunctionID	Feature	Value
<TITLE_FUN>	fno:executes	grel:toTitleCase
<TITLE_FUN>	ex:valueParam1	{stop_name}

```
@prefix rr: <http://www.w3.org/ns/r2rml#> .
@prefix ex: <http://example.com/> .
@prefix rml: <http://semweb.mmlab.be/ns/rml#> .
@prefix fno: <http://w3id.org/function/ontology#> .

<TITLE_FUNC>
  a fnml:FunctionTermMap ;
  fnml:functionValue [
    rml:logicalSource [
      rml:source "data/STOPS.csv" ;
      rml:referenceFormulation ql:CSV ;
    ];
    rr:predicateObjectMap [
      rr:predicateMap    fno:executes;
      rr:objectMap    [ rml:constant grel:toTitleCase ];
    ];
    rr:predicateObjectMap [
      rr:predicateMap    grel:valueParam1;
      rr:objectMap    [ rml:reference “stop_name” ];
    ];
  ];
.
```

Mapeathor: From spreadsheets to mappings



Resources:

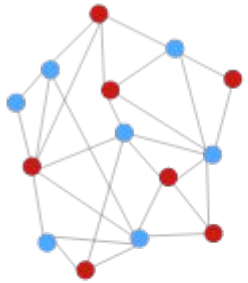
- Data: **Madrid GTFS-Benchmark**, about transport data in Madrid
  - 12 CSV files
  - 4.9 MB
- Ontology: **General Transit Feed Specification (GTFS)**
- Output: 1 mapping describing the data with the ontology as target squema

https://github.com/oeg-upm/kgc-tutorial-iswc2020/tree/master/mapeathor
https://morph.oeg.fi.upm.es/demo/mapeathor

morph

Morph-CSV: Constructing (Virtual) Knowledge Graphs from Real Tabular Data

Special thanks to Luis Pozo who helped in the development of the tool



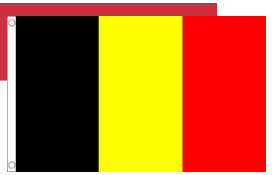
David Chaves-Fraga
dchaves@fi.upm.es
@dchavesf

17:10-17:50

Open Data Portals as data graveyards

Formatos	
ZIP	[illegible]
TSV	7013
Provisional data	2517
HTML	1754
CSV	1388
XML	1365
PDF	1273
Excel XLS	739
MDB	551
Octet Stream	446

Format
CSV (13699)
JSON (8935)
XLS (8576)
HTML (6178)
XLSX (4328)
PDF (4251)
XML (2753)
XML-APP (2662)
ASCII (2028)
PC-Axis (1908)
JPG (1780)



File type

- CSV (3527)
- WMS (2889)
- JSON (2839)
- KML (2316)
- XLS (1688)
- ZIP (1447)
- SHP (1343)
- WFS (1221)
- GEOJSON (1064)
- HTML (1059)
- WCS (878)
- XLSX (796)
- RDF/XML (569)
- JSONLD (556)
- N3 (556)
- TTL (556)
- XML (555)
- PDF (490)
- OV2 (247)
- GML (204)

Social aspects:

- Data is manually created by **humans without control**
- Data is intended to **be consumed by humans**

Technical (data quality) issues:

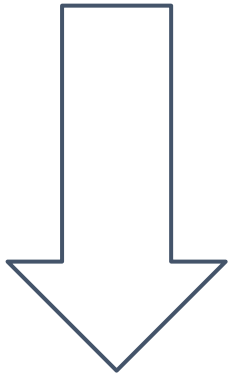
- No schema
- Not normalized
- Not uniform
- Missing data
- Missing meta-data
- No explicit joins

Actually, first time Freddy is sad

TRANSPORTE
Metro: Pinar de Chamartín (líneas 1, 4 y línea 1 metro ligero)Bus: 29, 125, 129, 150
Metro: Quintana (línea 5), Barrio de la Concepción (línea 7). Bus: 21, 146, 48.
Metro: Aluche (línea 5).Bus: 138, 139, 155, 17.Cercanías Renfe: Aluche (línea C5).
Metro: EmbajadoresBUS: 27 , 34 , 36 , 41 , 47 , 60 , 78 , 116 , 118 , 119 , 148Renfe: Embajadores
Metro: Barajas (línea 8)Bus: 101, 105, 115
Bus: 49 , 64
Metro: Alsacia (línea 2)Bus: 4, 38, 48, 165
Metro: Simancas (línea 7).Bus: 109, 38.
Metro: Diego de León (líneas 4 , 5 , 6)Bus: 1, 74, 48, 12, 43, 56, C1, C2
Metro: Torres Arias y Canillejas (línea 5)Bus: 77, 140
Metro: Arturo Soria (línea 4)Bus: 9, 72, 73

Transport data column from information about culture centers of Madrid in CSV

Let us find a solution!



But do not reinvent the wheel!

1) Follow a **declarative approach** for annotations (metadata and rules)

2) **Horizontal approach**

3) Do not delegate the problem to external solutions

4) **Analyze (implicit) information** from annotations to provide **optimizations** for virtual KG but also for ETL (materialize) processes

5) **Pushing down** the application of the proposed steps directly over the sources

- RML: RDF Mapping Language
  - Formats like XML, JSON, RDB, CSV
  - Purpose: Transformation to RDF
- CSVW: Metadata for CSV (Editor)
  - W3C recommendation
  - Purpose: Describe tabular data content
- FnO: The Function Ontology
  - Integrated with RML
  - Purpose: Defining transformation functions

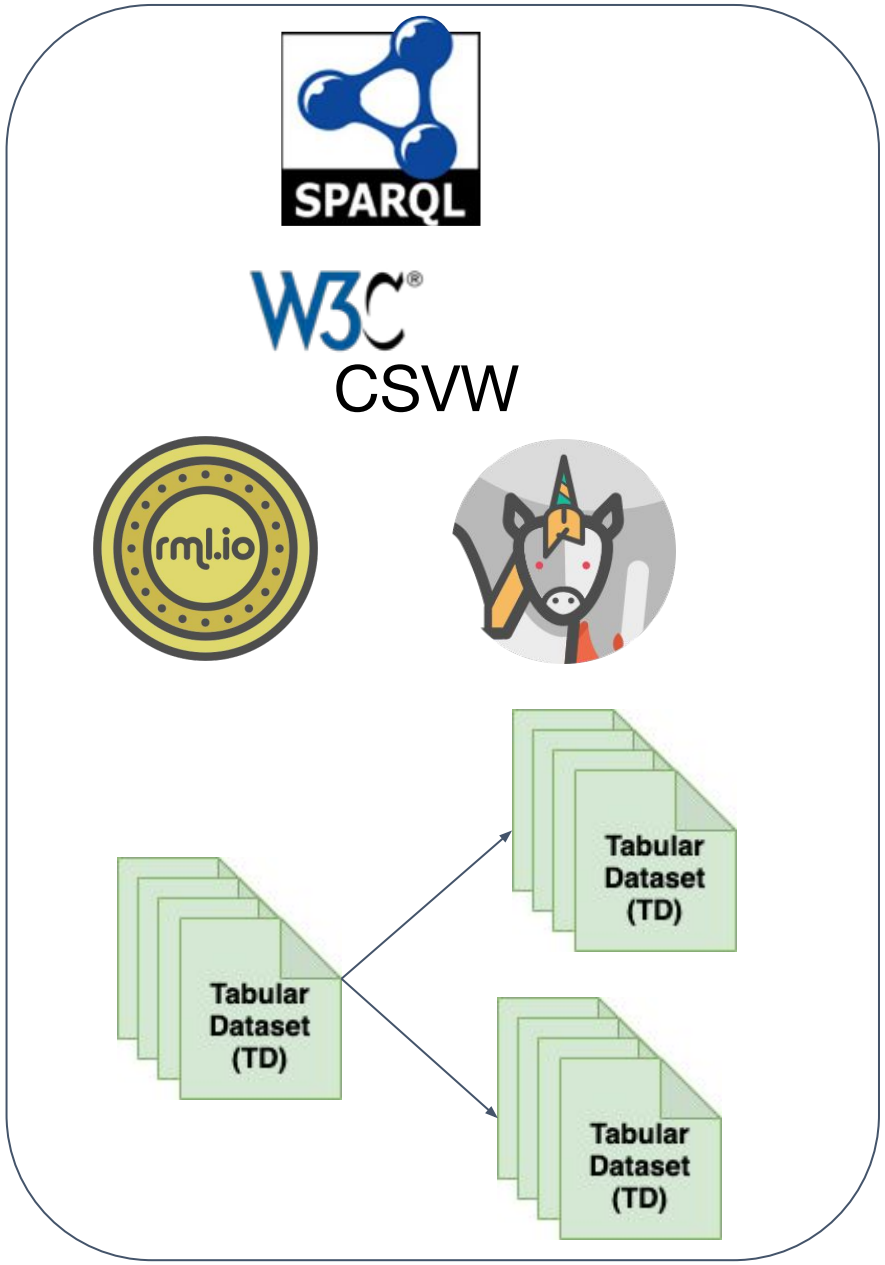
Can we reuse existing optimizations proposed
in SPARQL-to-SQL VKG construction?

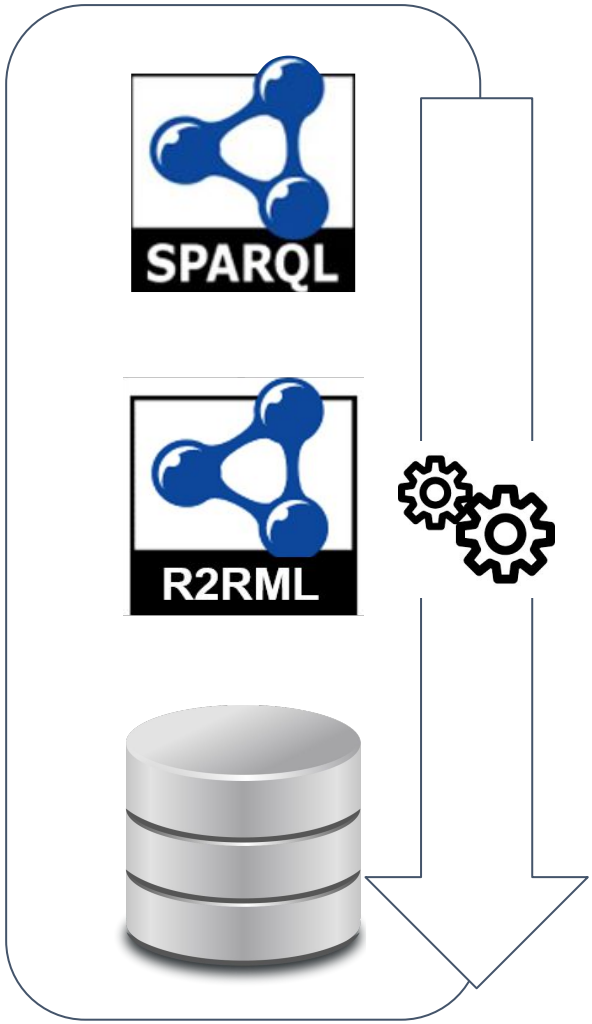
SPARQL-to-SQL optimizations assumptions:

- There is a **native query language** for the input data source.
- There is an schema and typically includes **integrity and domain constraints** (cleaned and normalized).
- The data source is an **RDB instance or is a NoSQL database** instance with an RDB wrapper.

Challenges VS Declarative Annotations in Tabular Data

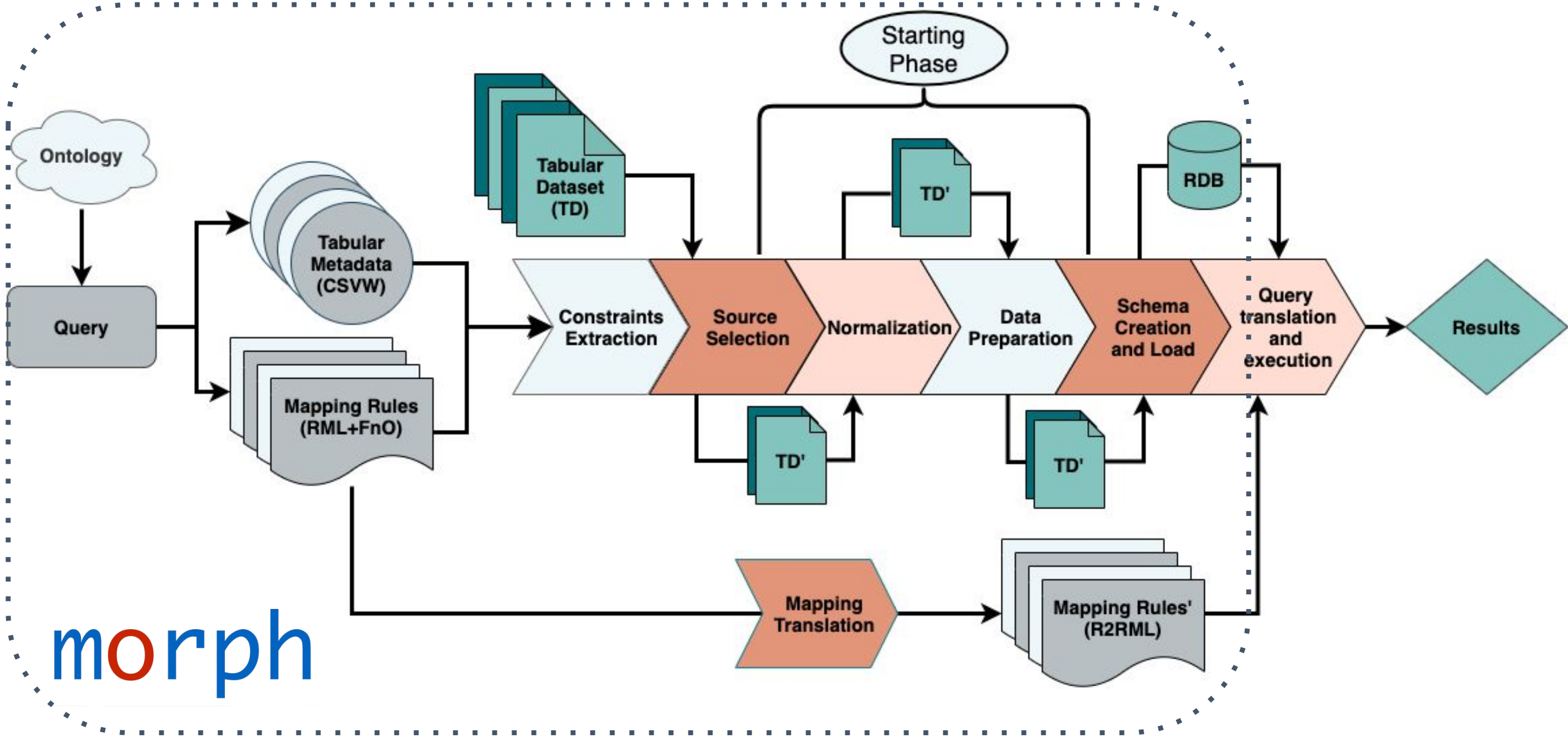
Challenge	Detailed Challenge	Relevant properties
Updated results	Select relevant sources and columns	SPARQL + Mapping
Lightweight Schema	Describing concepts and properties	rr: class / rr:predicateMap
	Add header to a CSV file	csvw:rowTitles
	Column datatype	csvw:datatype
Heterogeneity	Domain values	csvw:minimum, csvw:maximum
	Specify the format of a column	csvw:format
	Transform value	fnml:functionValue
	Default for missing values	csvw:default
	Specify NULL values	csvw:null
	Specify NOT NULL constraint	csvw:required
Not Normalized Sources	Integrity Constraints (PK & FK)	csvw:primaryKey / csvw:foreignKey
	Relationships between columns	rr:parentTriplesMap + rr:joinCondition
	Multiple entities in one source	rr:TriplesMap with same LogicalSource
	Support for multiple values in one cell	csvw:separator





total execution time ↓

query completeness ↑


Starting Phase
Ontology
Query
Tabular Metadata (CSVW)
Mapping Rules (RML+FnO)
Tabular Dataset (TD)
TD'
RDB
Constraints Extraction
Source Selection
Normalization
Data Preparation
Schema Creation and Load
Query translation and execution
Results
TD'
TD'
Mapping Translation
Mapping Rules' (R2RML)
morph

```
SELECT ?trip ?routeName ?routeType ?startTime ?endTime ?code WHERE {
  ?trip a gtfs:Trip .
  ?trip gtfs:route ?route .
  ?frequency a gtfs:Frequency .
  ?frequency gtfs:startTime ?startTime .
  ?frequency gtfs:endTime ?endTime .
  ?frequency gtfs:trip ?trip .
  ?route a gtfs:Route .
  ?route gtfs:shortName ?routeName .
  ?route gtfs:routeType ?routeType .
  ?routeType gtfs:routeTypeCode ?code  }
```

Select the relevant rules for the star shape groups of the input query

```
frequencies:
  sources:
   - [frequencies.csv~csv]
  s: mbench:freq/$(trip_id)-$(start_time)
  po:
   - [a, gtfs:Frequency]
   - [gtfs:startTime,$(start_time)]
   - [gtfs:endTime,$(end_time)]
   - [gtfs:headSecs,$(headway_secs)]
   - [gtfs:exactTimes,$(exact_times)]
   - p: gtfs:trip
     o:
      - mapping: trips
        condition:
          function: equal
          parameters:
            - [str1, $(trip_id)]
            - [str2, $(trip_id)]

trips:
  sources:
   - [trips.csv~csv]
  s: mbench:trips/$(trip_id)
  po:
   - [a, gtfs:Trip]
   - [gtfs:headsign, $(trip_headsign)]
   - [gtfs:shortName, $(trip_short_name)]
   - [gtfs:direction, $(direction_id)]
   - [gtfs:block, $(block_id)]
   - p: gtfs:route
     o:
      - mapping: routes
        condition:
          function: equal
          parameters:
            - [str1, $(route_id)]
            - [str2, $(route_id)]

routes:
  sources:
   - [routes.csv~csv]
  s: mbench:routes/$(route_id)
  po:
   - [a, gtfs:Route]
   - [gtfs:shortName, $(route_short_name)]
   - [gtfs:longName, $(route_long_name)]
   - [dct:description, $(route_desc)]
   - [gtfs:routeUrl, $(route_url)~iri]
   - [gtfs:color, $(route_color)]
   - [gtfs:textColor, $(route_text_color)]
   - p: gtfs:agency
     o:
      - mapping: agency
        condition:
          function: equal
          parameters:
            - [str1, $(agency_id)]
            - [str2, $(agency_id)]
   - p: gtfs:RouteType
     o:
      - mapping: route-type
        condition:
          function: equal
          parameters:
            - [str1, $(route_type)]
            - [str2, $(route_type)]

route-type:
  sources:
   - [routes.csv~csv]
  s: CONCAT(gtfs:,TRANS($(route_type)))
  po:
   - [a, gtfs:RouteType]
   - [gtfs:routeTypeCode,$(route_code)]
```

Step 1: Source selection

route_id	agency_id	route_short_name	route_long_name	route_type	route_code	route_url	route_color
4_1	CRTM	1	Chamartín-Valdecarros	1	401	http://crtm/metro/4_1	2DBEF0
4_2	CRTM	2	Las Rosas - C. Caminos	1	401	http://crtm/metro/4_2	ED1C24
4_3	CRTM	3	Villaverde Alto-Moncloa	1	401	http://crtm/metro/4_3	FFD000
4_4	CRTM	4	Chamartín-Argüelles	1	401	http://crtm/metro/4_4	B65518
5_C1	CRTM	C1	P.Pío-AeropuertoT4	2	109	http://crtm/train/5_1	4FB0E5
5_C2	CRTM	C2	Guadalajara-Chamartín	2	109	http://crtm/train/5_2	008B45
5_C3	CRTM	C3	Aranjuez-Escorial	2	109	http://crtm/train/5_3	9F2E86
5_C4	CRTM	C4	Parla-Colmenar Viejo	2	109	http://crtm/train/5_4	005AA3

Step 1: Source selection

```
frequencies:
  sources:
    - [frequencies.csv~csv]
  s: mbench:freq/$(trip_id)-$(start_time)
  po:
    - [a, gtfs:Frequency]
    - [gtfs:startTime,$(start_time)]
    - [gtfs:endTime,$(end_time)]
    - p: gtfs:trip
      o:
        - mapping: trips
          condition:
            function: equal
            parameters:
              - [str1, $(trip_id)]
              - [str2, $(trip_id)]
trips:
  sources:
    - [trips.csv~csv]
  s: mbench:trips/$(trip_id)
  po:
    - [a, gtfs:Trip]
    - p: gtfs:route
      o:
        - mapping: routes
          condition:
            function: equal
            parameters:
              - [str1, $(route_id)]
              - [str2, $(route_id)]
```

```
routes:
  sources:
    - [routes.csv~csv]
  s: mbench:routes/$(route_id)
  po:
    - [a, gtfs:Route]
    - [gtfs:longName, $(route_long_name)]
    - p: gtfs:RouteType
      o:
        - mapping: route-type
          condition:
            function: equal
            parameters:
              - [str1, $(route_type)]
              - [str2, $(route_type)]
route-type:
  sources:
    - [routes.csv~csv]
  s: CONCAT(gtfs:,TRANS($(route_type)))
  po:
    - [a, gtfs:RouteType]
    - [gtfs:routeTypeCode,$(route_code)]
```

Routes and route-types has the same LogicalSource

⇩

Source contains information from independent entities

route_id	route_long_name	route_type	route_code
4_1	Chamartín-Valdecarros	1	401
4_2	Las Rosas - C. Caminos	1	401
4_3	Villaverde Alto-Moncloa	1	401
4_4	Chamartín-Argüelles	1	401
5_C1	P.Pío-AeropuertoT4	2	109
5_C2	Guadalajara-Chamartín	2	109
5_C3	Aranjuez-Escorial	2	109
5_C4	Parla-Colmenar Viejo	2	109

Step 2: Normalization

route_id	route_long_name	route_type
4_1	Chamartín-Valdecarros	1
4_2	Las Rosas - C. Caminos	1
4_3	Villaverde Alto-Moncloa	1
4_4	Chamartín-Argüelles	1
5_C1	P.Pío-AeropuertoT4	2
5_C2	Guadalajara-Chamartín	2
5_C3	Aranjuez-Escorial	2
5_C4	Parla-Colmenar Viejo	2

route_type	route_code
1	401
1	401
1	401
1	401
2	109
2	109
2	109
2	109

1) **route TriplesMap references:** route_id, route_long_name, route_type
2) **route_type TriplesMap references:** route_type, route_code

route_type	route_code	route_type_fn
1	401	Subway
2	109	Train

```
route-type:
    sources:
      - [routes_types.csv~csv]
    s: gtfs:$(route_type_fn)
     po:
      - [gtfs:routeTypeCode,$(route_code)]
```

1. Remove duplicates from the input sources (from 8 to 2 rows)
2. Substitute values from CSVW annotations (csvw:default, csvw:null, csvw:format) directly over tabular data
3. Generate the SQL actions for the ad-hoc transformation functions (fnml:functionValue)

```
UPDATE route_type_table SET route_type_fn = REPLACE(route_type, '1', 'Subway')
```

Step 4/5: Schema creation and load and Mapping Translation

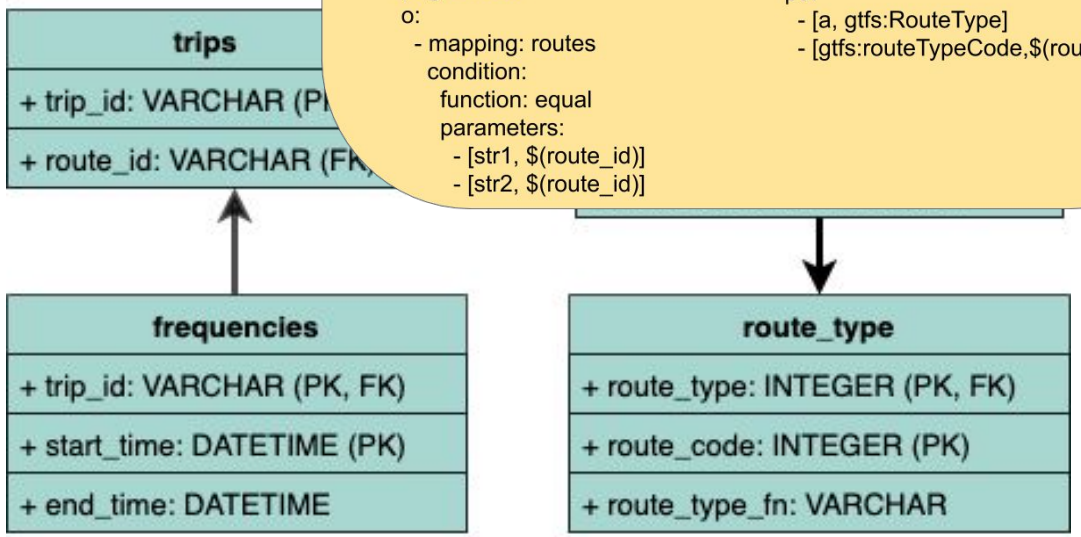
```
frequencies:
  sources:
   - [frequencies.csv~csv]
  s: mbench:freq/$(trip_id)-$(start_time)
  po:
   - [a, gtfs:Frequency]
   - [gtfs:startTime,$(start_time)]
   - [gtfs:endTime,$(end_time)]
   - [gtfs:headSecs,$(headway_secs)]
   - [gtfs:exactTimes,$(exact_times)]
   - p: gtfs:trip
     o:
      - mapping: trips
        condition:
          function: equal
          parameters:
            - [str1, $(trip_id)]
            - [str2, $(trip_id)]

trips:
  sources:
   - [trips.csv~csv]
  s: mbench:trips/$(trip_id)
  po:
   - [a, gtfs:Trip]
   - [gtfs:headsign, $(trip_headsign)]
   - [gtfs:shortName, $(trip_short_name)]
   - [gtfs:direction, $(direction_id)]
   - [gtfs:block, $(block_id)]
   - p: gtfs:route
     o:
      - mapping: routes
        condition:
          function: equal
          parameters:
            - [str1, $(route_id)]
            - [str2, $(route_id)]
```

```
routes:
  sources:
   - [routes.csv~csv]
  s: mbench:routes/$(route_id)
  po:
   - [a, gtfs:Route]
   - [gtfs:shortName, $(route_short_name)]
   - [gtfs:longName, $(route_long_name)]
   - [dct:description, $(route_desc)]
   - [gtfs:routeUrl, $(route_url)~iri]
   - [gtfs:color, $(route_color)]
   - [gtfs:textColor, $(route_text_color)]
   - p: gtfs:agency
     o:
      - mapping: agency
        condition:
          function: equal
          parameters:
            - [str1, $(agency_id)]
            - [str2, $(agency_id)]
   - p: gtfs:RouteType
     o:
      - mapping: route-type
        condition:
          function: equal
          parameters:
            - [str1, $(route_type)]
            - [str2, $(route_type)]

route-type:
  sources:
   - [routes.csv~csv]
  s: CONCAT(gtfs:,TRANS($(route_type)))
  po:
   - [a, gtfs:RouteType]
   - [gtfs:routeTypeCode,$(route_code)]
```

R2RML

```
frequencies:
  tables:
   - [frequencies]
  s: mbench:freq/$(trip_id)-$(start_time)
  po:
   - [a, gtfs:Frequency]
   - [gtfs:startTime,$(start_time)]
   - [gtfs:endTime,$(end_time)]
   - p: gtfs:trip
     o:
      - mapping: trips
        condition:
          function: equal
          parameters:
            - [str1, $(trip_id)]
            - [str2, $(trip_id)]
trips:
  tables:
   - [trips]
  s: mbench:trips/$(trip_id)
  po:
   - [a, gtfs:Trip]
   - p: gtfs:route
     o:
      - mapping: routes
        condition:
          function: equal
          parameters:
            - [str1, $(route_id)]
            - [str2, $(route_id)]
```

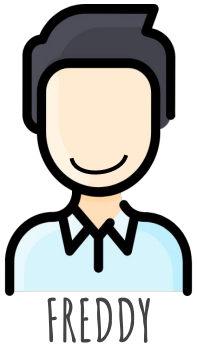
```
routes:
  tables:
   - [route]
  s: mbench:routes/$(route_id)
  po:
   - [a, gtfs:Route]
   - [gtfs:longName, $(route_long_name)]
   - p: gtfs:RouteType
     o:
      - mapping: route-type
        condition:
          function: equal
          parameters:
            - [str1, $(route_type)]
            - [str2, $(route_type)]

route-type:
  tables:
   - [route_type]
  s: gtfs:$(route_type_fn)
  po:
   - [a, gtfs:RouteType]
   - [gtfs:routeTypeCode,$(route_code)]
```



morph

Resources:

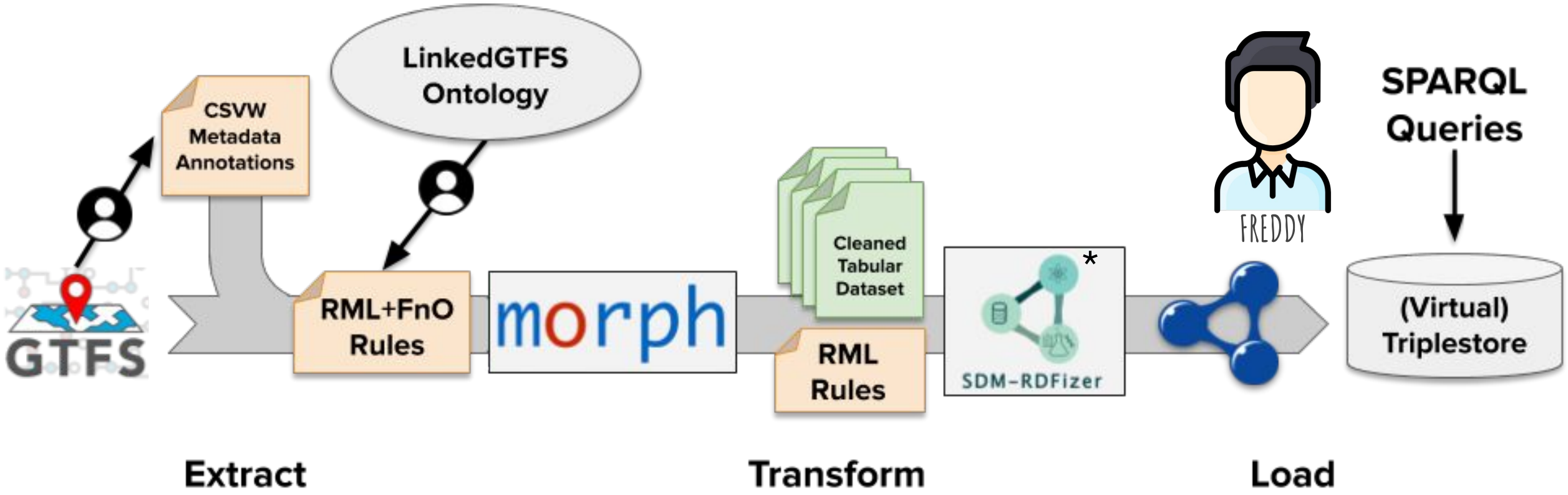
- Morph-CSV website: https://morph.oeg.fi.upm.es/tool/morph-csv
- Morph-CSV has been evaluated over BSBM, GTFS-Madrid-Bench and Bio2RDF: https://morph.oeg.fi.upm.es/demo/morph-csv
- DOI: https://doi.org/10.5281/zenodo.3731941
- Local UI-webapp to run Morph-CSV
- Demo Video



Requirements for run it:

1. docker-compose
2. YARRRML+FnO mapping and CSVW annotations
3. [R2]RML-compliant engine

Materialization process with Morph-CSV and SDM-RDFizer



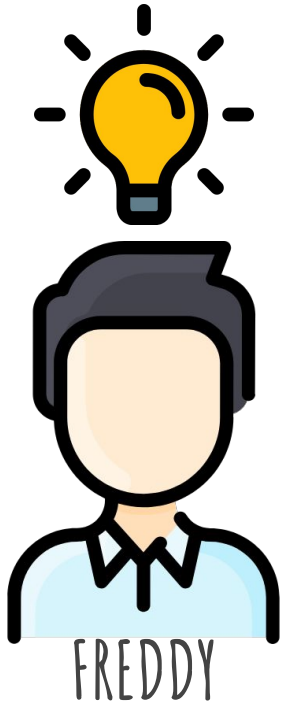
*Iglesias, E., Jozashoori, S., Chaves-Fraga, D., Collarana, D., & Vidal, M. E. (2020, October). *SDM-RDFizer: An RML interpreter for the efficient creation of rdf knowledge graphs*. In Proceedings of the 29th ACM International Conference on Information & Knowledge Management

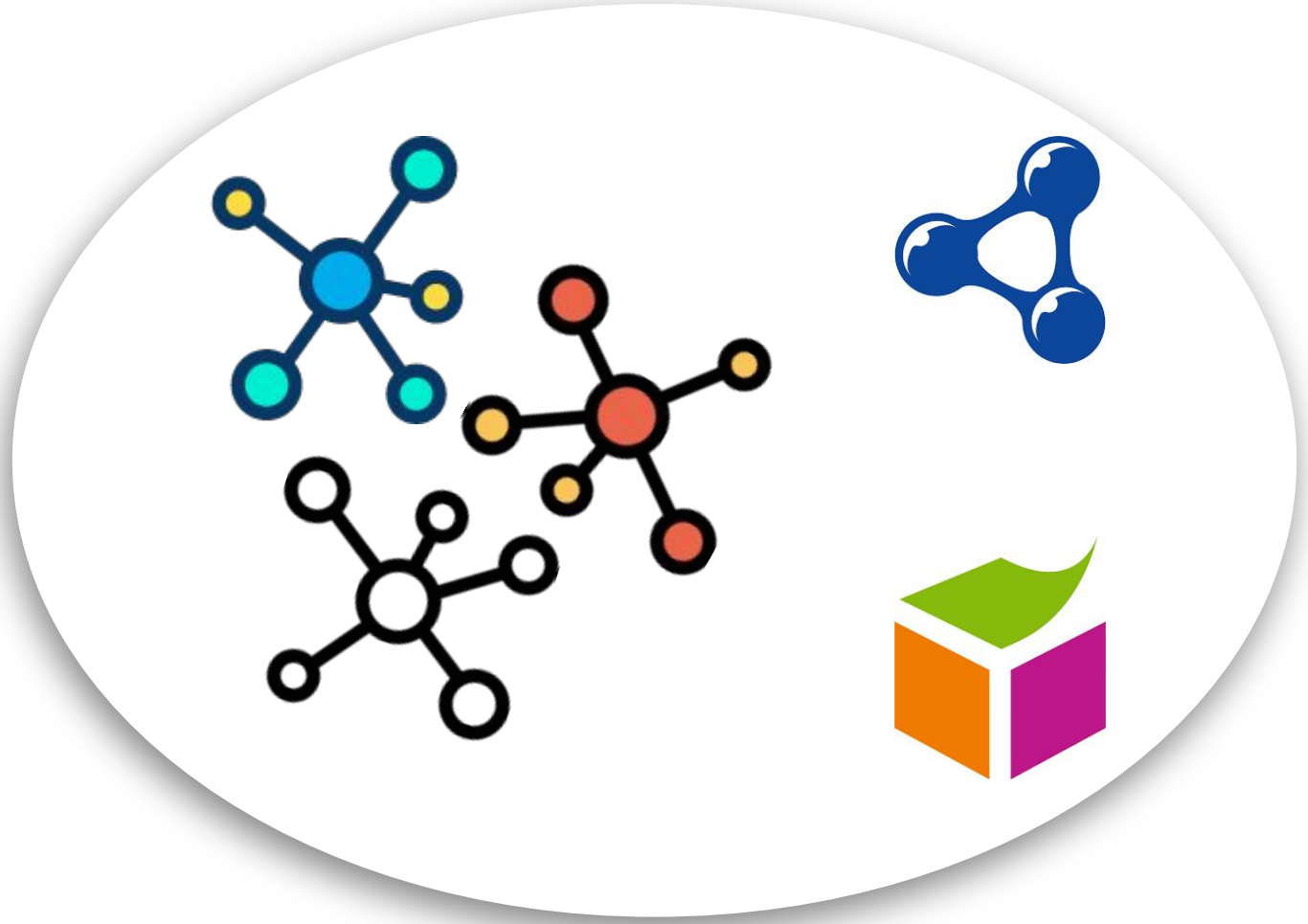
Andrea Cimmino

Post-Doctoral researcher
cimmino@fi.upm.es
@acimmino

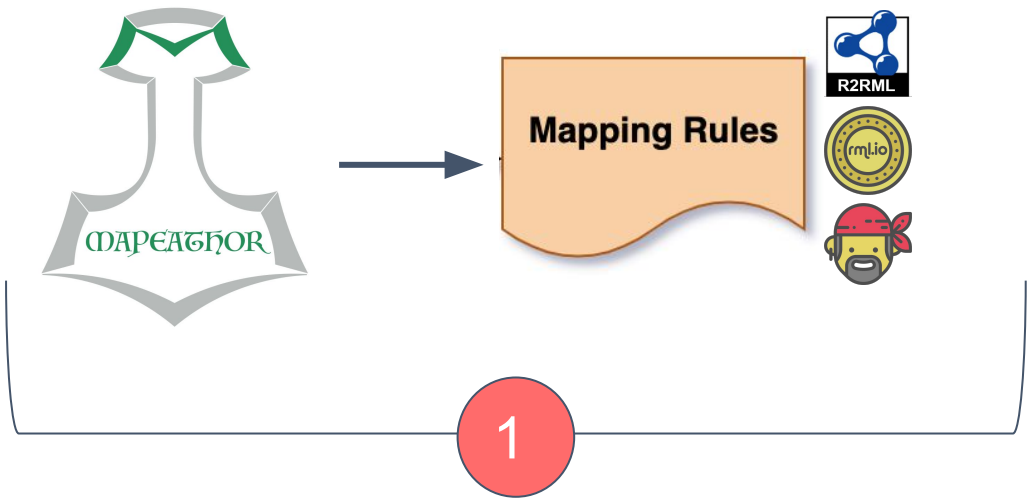
18:05-18:45

So he decided to build a **Knowledge Graph**, the solution to represent clearly the heterogeneous data and make it **clear**, **accessible** and **queriable**

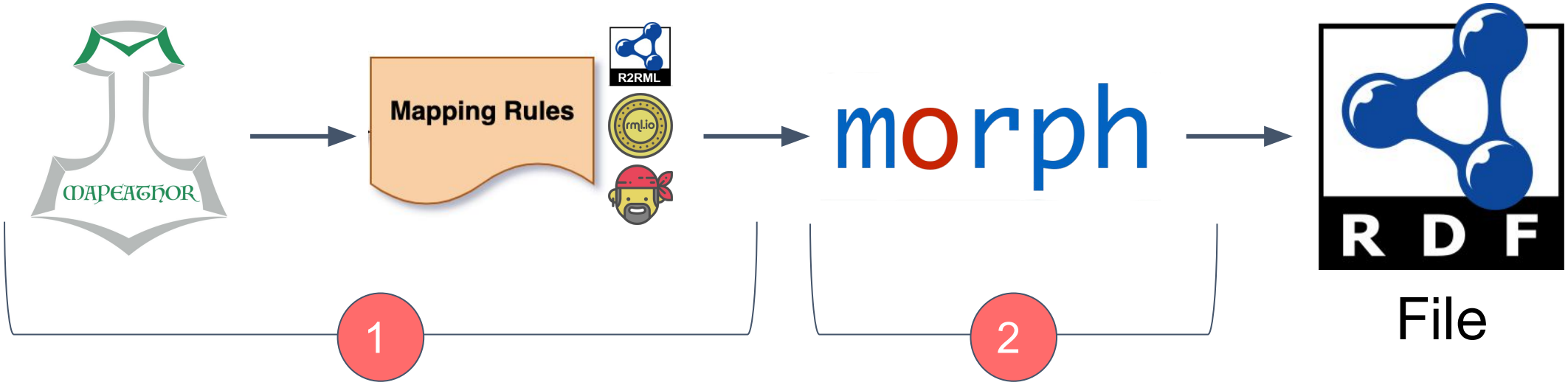


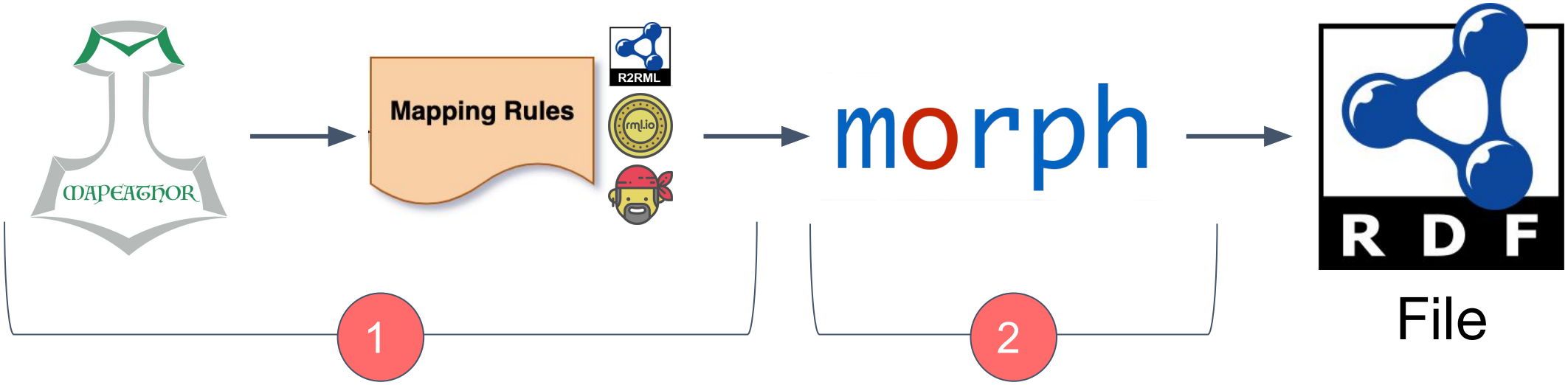


A bit of recap.



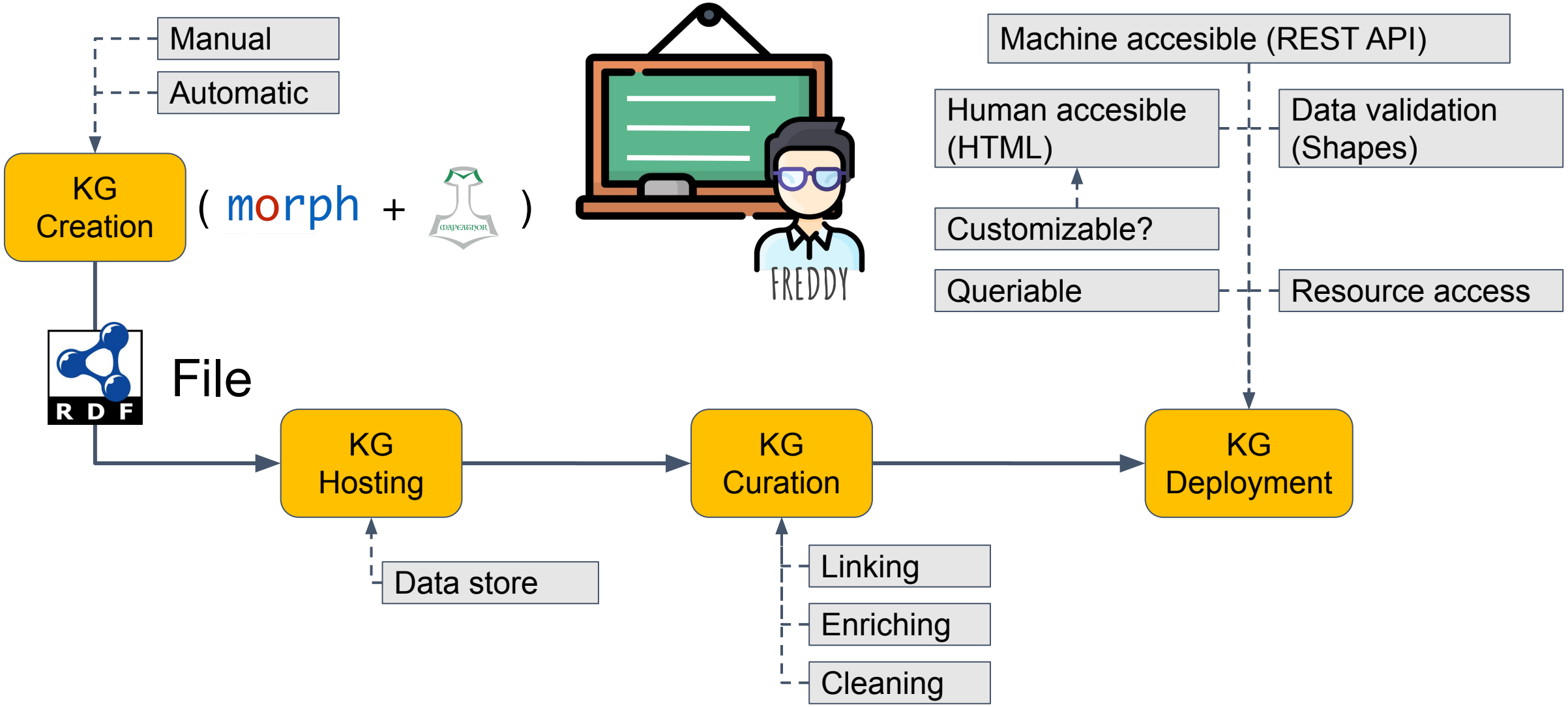
A bit of recap.





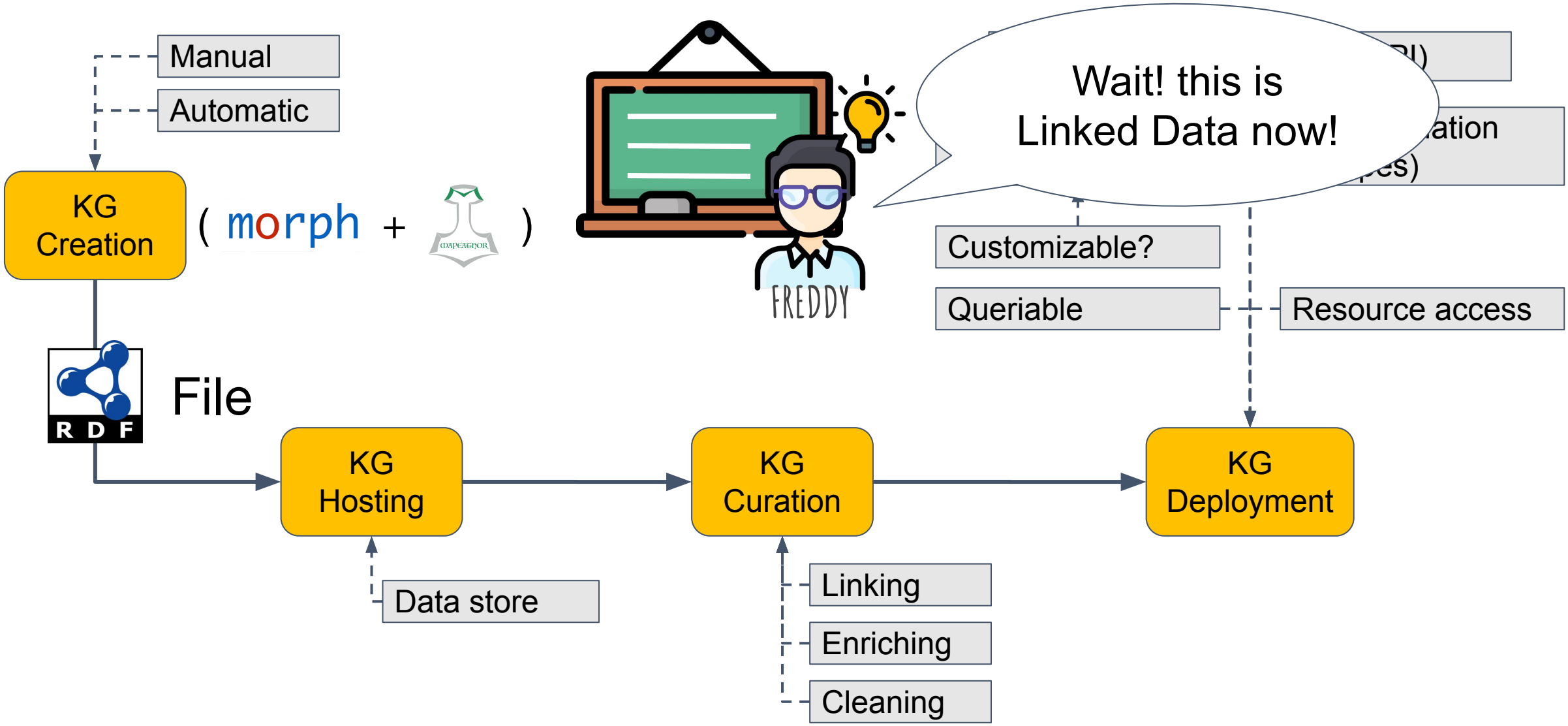
- **RDF File data is not consumable or queriable automatically...**
  - Data must be **curated** and/or **linked** (if needed)
  - Data must be **hosted** in a suitable environment (triple store?)
  - Data must be **published for humans** to be consumable (HTML interfaces)
  - Data must be **published for machines** to be consumable (REST interfaces)
  - Data should be **queriable** for both humans and machines (SPARQL)
  - ....

The Knowledge Graph (KG) lifecycle*



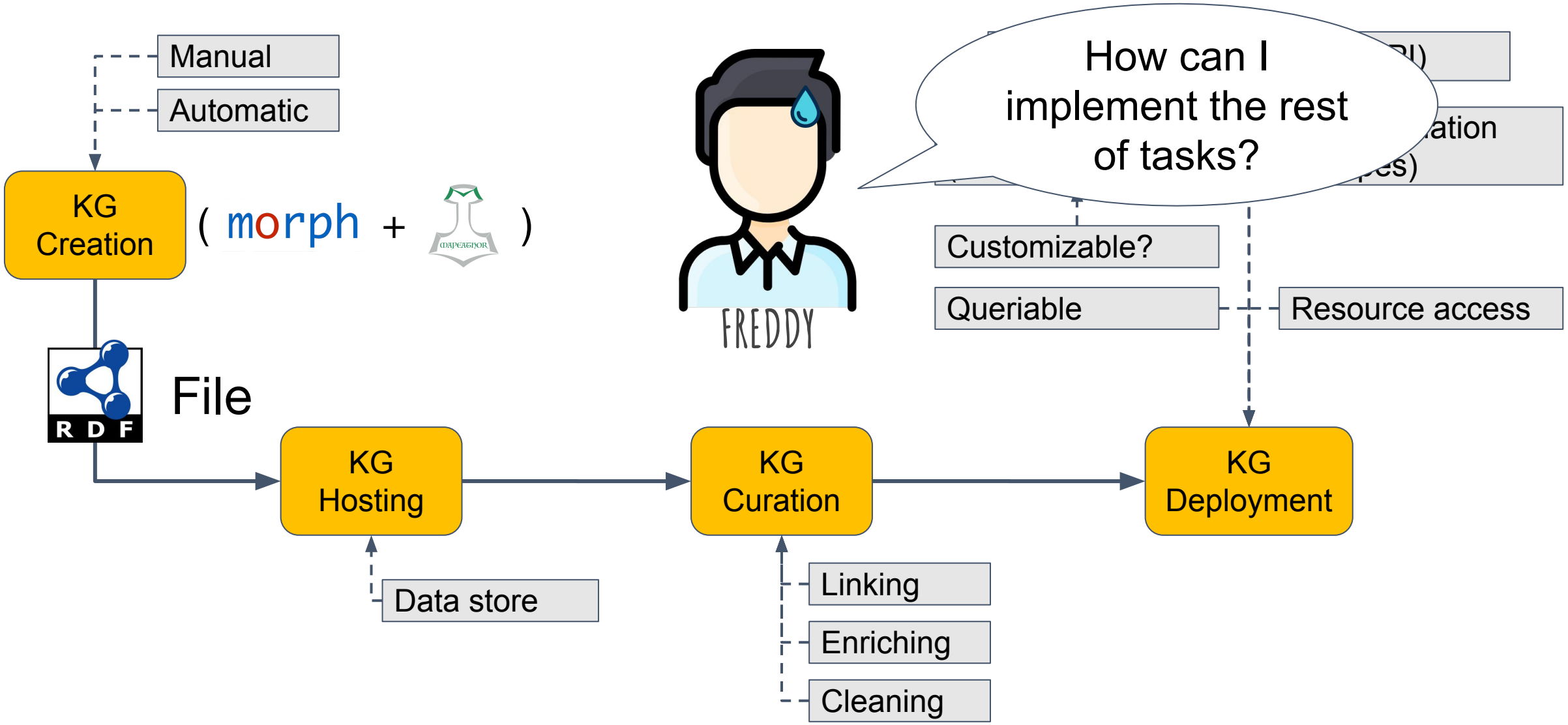
*Fensel et al. Knowledge Graphs - Methodology, Tools and Selected Use Cases. 2020

The Knowledge Graph (KG) lifecycle*



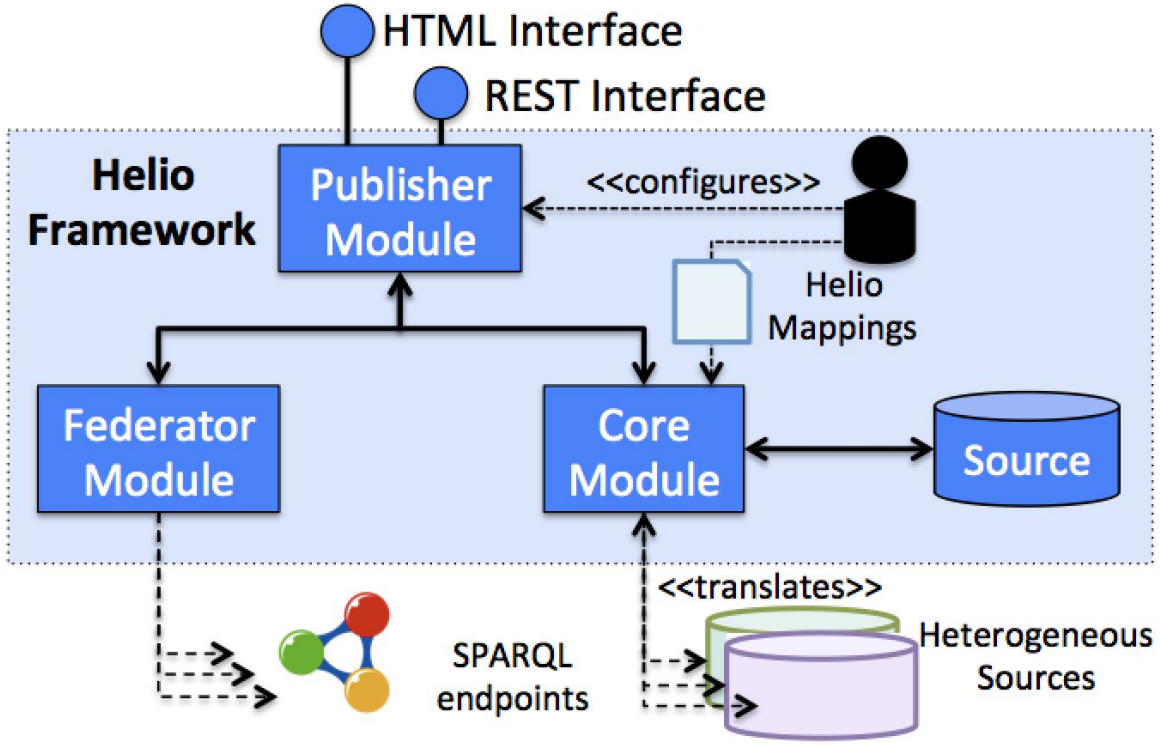
*Fensel et al. Knowledge Graphs - Methodology, Tools and Selected Use Cases. 2020

The Knowledge Graph (KG) lifecycle*

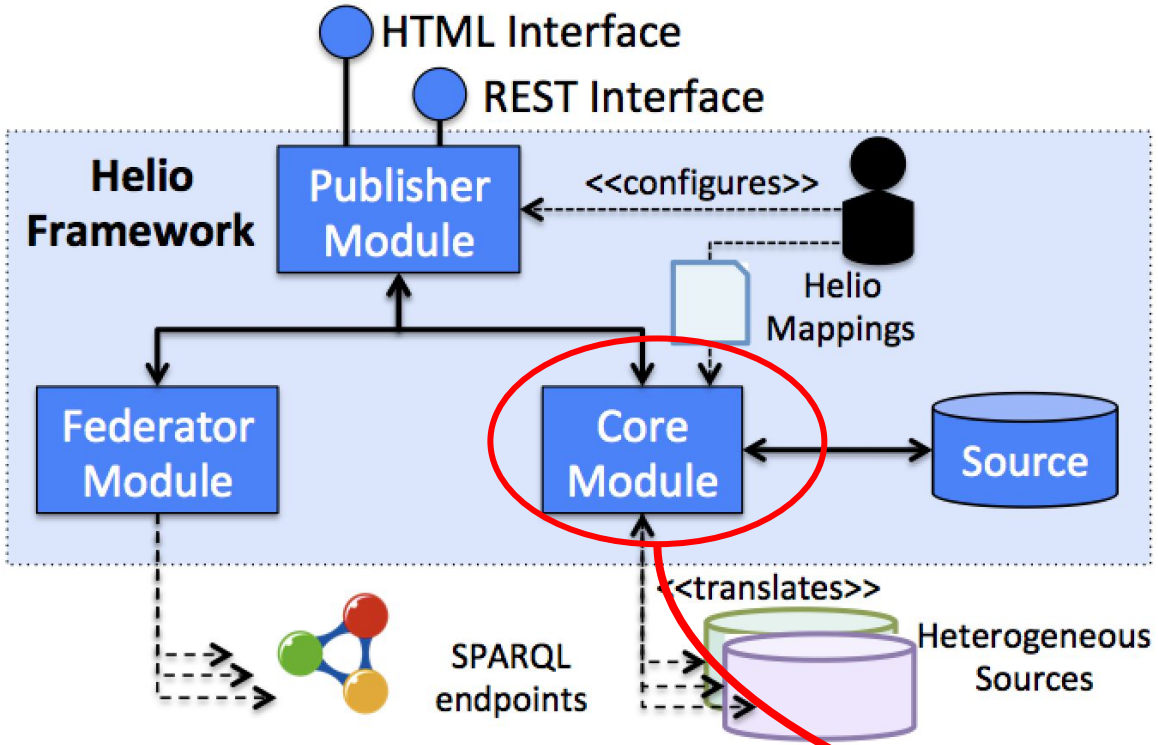


*Fensel et al. Knowledge Graphs - Methodology, Tools and Selected Use Cases. 2020

- Helio aims at assisting users during the whole KG lifecycle

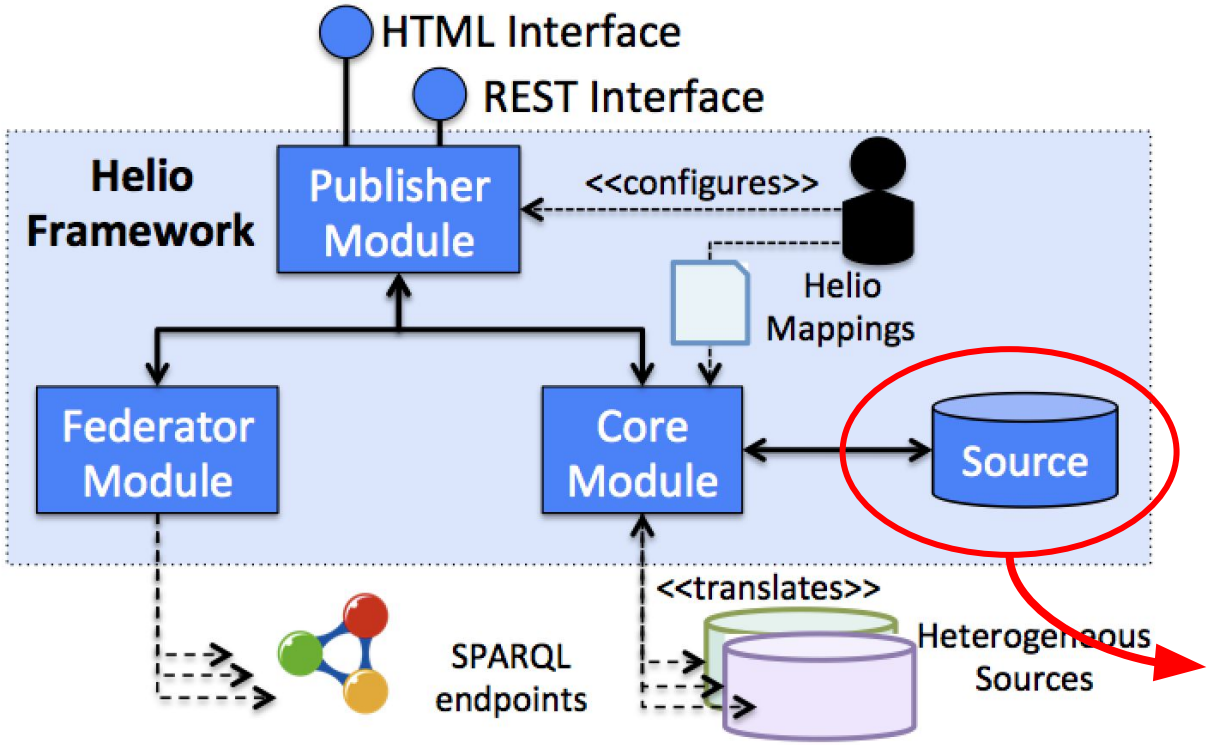


- Helio aims at assisting users during the whole KG lifecycle



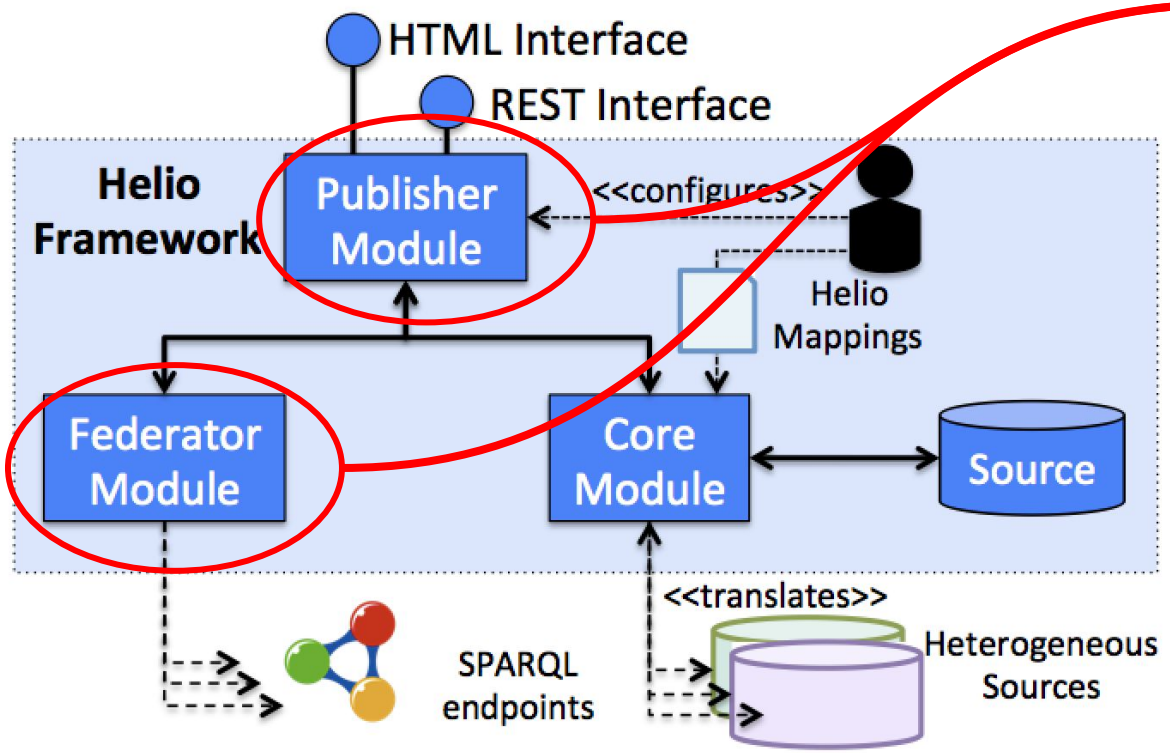
1. Generates RDF from heterogeneous sources
   a. allows existing RDF in the pipeline
   b. async/sync
2. Re-uses (if necessary) other technologies
   a. e.g., morph-csv
3. Cross-mapping language
4. Allows: cleaning, enriching and linking
5. Has a plugin mechanisms for extending:
   a. New Data sources or formats
   b. New mappings or functions
   c. New connectors to store the RDF

- Helio aims at assisting users during the whole KG lifecycle



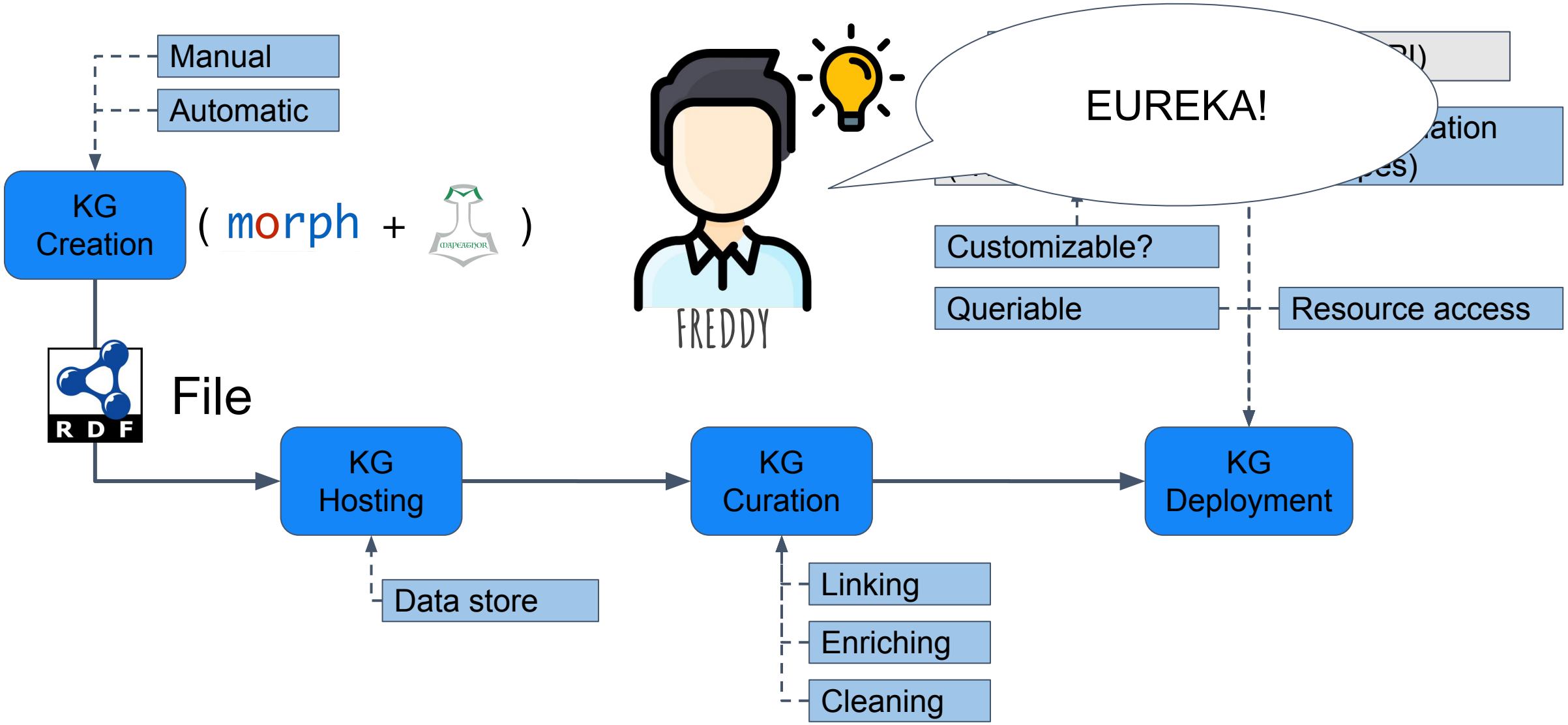
1. Generated RDF is stored in an internal triple store
2. The triple store can be configured
   a. i.e., existing stores like GraphDB can be used changing the Helio configuration
3. The plugin system allows to include new tailored-designed stores
4. Allows validating data with shapes

- Helio aims at assisting users during the whole KG lifecycle



1. Provides Human interfaces (HTML)
   a. Customizables
   b. Dynamic creation of views
2. Provides Machine interface (REST API)
3. Enables fine-grained data access:
   a. SPARQL
   b. Resource access
   c. Dataset access
4. The Federator Module allows providing a KG built from distributed sources

The Helio coverage



*Fensel et al. Knowledge Graphs - Methodology, Tools and Selected Use Cases. 2020

Different approaches to address this scenario

1. Publish the RDF file generated by Morph-csv
   a. How to
2. Generate + Publish the RDF data using the mappings of Mapeathor (RML)
   a. How to
3. Customise the views (HTML) of the published data
   a. Create a dynamic view of the data
4. Integrate Morph-csv in the Helio pipeline using a plugin

Different approaches to address this scenario

1. **Publish the RDF file generated by Morph-csv**
   a. **How to**
2. Generate + Publish the RDF data using the mappings of Mapeathor (RML)
   a. How to
3. Customise the views (HTML) of the published data
   a. Create a dynamic view of the data
4. Integrate Morph-csv in the Helio pipeline using a plugin

Different approaches to address this scenario

1. Publish the RDF file generated by Morph-csv
   a. How to
2. Generate + Publish the RDF data using the mappings of Mapeathor (RML)
   a. How to
3. **Customise the views (HTML) of the published data**
   a. **Create a dynamic view of the data**
4. Integrate Morph-csv in the Helio pipeline using a plugin

Different approaches to address this scenario

1. Publish the RDF file generated by Morph-csv
   a. How to
2. Generate + Publish the RDF data using the mappings of Mapeathor (RML)
   a. How to
3. Customise the views (HTML) of the published data
   a. Create a dynamic view of the data
4. **Integrate Morph-csv in the Helio pipeline using a plugin**

- Helio can help users for generating and publishing KG
- Website: https://oeg-upm.github.io/helio/
- Streamlined use cases
  - https://github.com/oeg-upm/helio/wiki/Streamlined-use-cases

- Helio users:
  - 3 European Projects
  - 2 Data portals
  - 5 papers
  - 3 bachelor proposals
  - 2 conference tutorials

CONCLUSIONS

18:45-19:00

Conclusions

Pros:

- Full semantic data integration pipeline for heterogeneous data
  - Helping in the creation of mapping rules
  - Efficiently dealing with heterogeneity issues in real-world tabular data
  - Assisting KG lifecycle for enhancing accessibility and quality
- Benefits of using declarative mapping rules
  - Reproducibility
  - Maintainability
  - …
- Minimization of manual and unreproducible tasks.
- Freddy is finally happy ;-)

Cons:

- Rules and annotations are still created manually

Discussion

- What are the limitations of this pipeline over your own use cases?
- Are you convinced to use mapping rules after the tutorial?
  - Would it be necessary to organize more tutorials/webinars about this topic?
- Engineering VS Research
  - Are all the problems solved from a research perspective? (i.e., we need more easy and friendly tools)

- More comments and discussions?
  - Help us and join the W3C KGC - CG: https://www.w3.org/community/kg-construct/

Knowledge Graph Construction using Declarative Mapping Rules

David Chaves-Fraga, Ana Iglesias-Molina, Andrea Cimmino, Oscar Corcho

Ontology Engineering Group
Universidad Politécnica de Madrid, Spain

dchaves@fi.upm.es oeg-upm 01/11/20 ISWC2020