Towards the Definition of a Language-Independent Mapping Template for Knowledge Graph Creation

Ana Iglesias-Molina
Ontology Engineering Group
Universidad Politécnica de Madrid, Spain
ana.iglesiasm@upm.es

David Chaves-Fraga
Ontology Engineering Group
Universidad Politécnica de Madrid, Spain
dchaves@fi.upm.es

Freddy Priyatna
Ontology Engineering Group
Universidad Politécnica de Madrid, Spain
fpriyatna@fi.upm.es

Oscar Corcho
Ontology Engineering Group
Universidad Politécnica de Madrid, Spain
ocorcho@fi.upm.es

ABSTRACT

The use of knowledge graphs is spreading in the scientific community across different domains, from social sciences to biomedicine. The creation of knowledge graphs usually needs the integration of multiple heterogeneous data sources in different formats and schemas. One common way to achieve this process is using declarative mappings, which establish the relationships between the source data and the ontology, improving relevant aspects such as maintainability, readability and understandability. Learning how to use and create mappings is not an easy task, hindering the use of this technology to anyone outside the area. As a result, this task is usually carried out by experts. To ease the mapping creation, several mapping editors have been developed, but their success is limited. In this paper, we devise the use of a well-known tool commonly used in the scientific community, the spreadsheets, to specify the mapping rules in a language-independent way. Our aim is to ease the mapping creation and make it more accessible for the community. We also show a real use case, in which using spreadsheets helps in the mapping creation process and enables a handy way for editing and visualizing mapping rules.

CCS CONCEPTS

• **Computing methodologies** → **Artificial intelligence**; Knowledge representation and reasoning.

KEYWORDS

Knowledge graph, spreadsheet, declarative mapping

1 INTRODUCTION

The expansion of the Semantic Web technologies has reached users across several domains, such as legal and biomedical. An increasing number of knowledge graphs from these areas are being created, restructuring knowledge in a machine-readable way [4]. For their construction it is necessary to integrate different data sources; then they allow search optimization and the possibility of applying machine learning techniques to obtain new knowledge, among other possibilities. Some examples are DBpedia [1] and Wikidata [18].

There are multiple approaches to create knowledge graphs, from using ad-hoc tools to declarative mappings. The later defines rules to establish relationships between the global schema and the data sources. Examples of mappings languages are the W3C recommendation R2RML [7] and its extension RML [9].

The use of declarative mappings for semantic web non-experts is often complicated. That is one of the reasons why the mapping creation is usually carried out by knowledge engineers. This poses a barrier for potential users from other domains. To face this issue, several mapping editors have been proposed. They aim at making the mapping creation and editing easier and more intuitive [11, 16]. Despite these efforts, users prefer to use tools like OpenRefine[1], which is non-declarative, thus hindering the reproducibility and maintainability of the transformations performed.

Mapping languages consist of common elements to be created (e.g. the source data, subjects, predicates and objects). In this paper we propose the use of spreadsheets to specify these elements, the mapping rules, in a language-independent way, so it can be translated into the most convenient specification [6]. Spreadsheets are a well-known tool commonly used in the scientific community, versatile and easy to understand, what makes them a suitable target to specify mapping rules. With this proposal, our aim is to lower the barrier of mapping creation and motivate the scientific community to use this technology.

This paper is organized as follows: Section 2 presents the related work done on mapping creation. Section 3 shows the common mapping structure. Section 4 describes the spreadsheet template we propose for the creation of mapping rules. Section 5 shows a real case in which we use spreadsheets to create mappings. Finally, section 6 presents the conclusions and areas for future work.

2 RELATED WORK

A wide variety of mapping languages has been proposed over the last decades [8]. The W3C Recommendation is R2RML [7], a declarative mapping language that allows the generation of adapters to transform relational databases into RDF. There are other declarative languages that enable dealing with more data formats, such as RML [9] (extension of R2RML for CSV, JSON and XML), YARRRML [10] (a user-friendly serialization of RML), xR2RML [15] (for non-SQL databases) and RMLC-Iterator [5] (for statistical data).

There are not as many mapping editors as languages; in fact, the majority of them support R2RML or RML. Some of the most used

[1] http://openrefine.org/

```
<PERSON>
 rml:logicalSource [
    rml:source "/home/user/data/people.csv" ;
    rml:referenceFormulation ql:CSV ;
 ];
 rr:subjectMap [
    rr:class ex:Person;
    rr:template "http://ex.com/Person/{name}";
 ];
 rr:predicateObjectMap [
    rr:predicateMap    [ rr:constant ex:name ];
    rr:objectMap    [ rml:reference "name" ];
 ];
 rr:predicateObjectMap [
    rr:predicateMap    [ rr:constant ex:sport ];
    rr:objectMap    [ rr:parentTriplesMap <SPORT>;
      rr:joinCondition [ rr:child "sport_id"; rr:parent "id"; ];
    ];
 ];
```

(a) Triples Map for PERSON

```
<SPORT>
 rml:logicalSource [
    rml:source "/home/user/data/sports.csv" ;
    rml:referenceFormulation ql:CSV ;
 ];
 rr:subjectMap [
    rr:class ex:Sport;
    rr:template "http://ex.com/Sport/{sport}";
 ];
 rr:predicateObjectMap [
    rr:predicateMap    [ rr:constant ex:name ];
    rr:objectMap    [ rml:reference "sport" ];
 ];
 rr:predicateObjectMap [
    rr:predicateMap    [ rr:constant ex:code ];
    rr:objectMap    [ rml:reference "id"; ];
 ];
```

(b) Triples Map for SPORT

Figure 1: RML mapping. Fig. 2a shows the triples map that generates instances of the class `ex:Person` and two predicate-object maps, the latest a join to the Triples Map shown in Fig. 2b, that creates the instances for the class `ex:Sport` and two predicate-object maps.

```
"name","birthdate","sport_id"
"Serena Williams",19810926,1
"Alexander Ovechkin",19850917,4
"Emily Scarratt",19900208,3
"Javier Fernández",19910415,2
```

(a) people.csv

```
"id","sport"
1,"Tennis"
2,"Ice skating"
3,"Rugby"
4,"Hockey"
```

(b) sports.csv

Figure 2: CSV data example. Example of the source data in CSV format for the RML mapping example form Figure 1.

tools implement graphical visualization and editing of the mappings as graphs, such as Karma [13] and Map-On [17] for R2RML, and RMLEditor [11] for RML. Others provide an environment to write them, like OntopPro[2], an extension of Protégé that allows mapping creation in their custom language and import/export R2RML.

The current mapping editors are language-oriented or create the mapping rules through graphical visualization. Thus, the user either knows the language, or creates the mapping building a visual graph. Using spreadsheets enables a language-independent declarative approach to write concisely the mapping rules taking advantage of the functionalities of a spreadsheet. In other words, the rules can be created specifying only the essential elements without knowing any mapping language, and the repetitive elements can be autocompleted. Moreover, its compact structure allows a quick visualization of all the rules.

There are other approaches that use spreadsheets to capture knowledge of domain experts [12, 19]. This kind of tools enable the specification of ontologies in tables and generate the corresponding RDF. Similarly, the mapping rules for data conversion are declared in spreadsheets with our proposal, to be later translated into different mapping languages.

3 STRUCTURE OF DECLARATIVE MAPPINGS

The mapping languages have usually a similar structure, as many of them are based on the standard. The earliest (e.g. R2O [2]) or the non-declarative languages (e.g. SPARQL-Generate [14]) differ in structure, but they all share the same elements: identifier of data sources (URL, path, table name) and the rules for generating the corresponding RDF triples. An RML mapping example is shown in Figure 1. It organizes the transformation rules in two triple maps, one for each data source (Figure 2) used to generate RDF triples.

We define more in detail the essential elements that declarative mapping rules contain, providing examples based on the RML mappings showed in Figure 1:

- An element that specifies where the data sources are stored. In the case of RML, these elements are defined using the property `rml:logicalSource`.
- A set of rules that defines the subjects and classes of the triples. In RML, the `rr:subjectMap` property is used to specify these characteristics.
- Pairs (`rr:predicateObjectMap` property in RML) that specify rules for generating predicate (`rr:predicateMap`) and object (`rr:objectMap`) of the triples.
- Join condition to another triple map, where the subject of the referenced triples map is to be the object in the new triple. This is defined in RML using `rr:joinCondition` property.

As we show in the example mapping, these rules usually contain multiple and repetitive elements to describe the rules. This characteristic makes it easy to commit mistakes when writing them manually. Using a spreadsheet template can ease this process to non-experts in mapping creation. It enables manual writing, while helping with the repetitive parts with autocompleting functions. Moreover, all the language's syntax and formatting is later automatically written by the tool, not the user.

[2]https://github.com/ontop/ontop/wiki/ontopProUserManual

4 SPREADSHEET DESIGN

In this section we show the designed spreadsheet template[3] that contains the essential elements to create a mapping. It consists of at least four sheets: prefixes, source data, subject and predicate-object maps; and optionally, a sheet with transformation functions.

Prefixes sheet. In this sheet the namespace prefixes for URLs are specified. They can be found at the beginning in most of mapping languages, as they make it easier and shorter to write the mappings. This sheet is composed of two columns, in the column `Prefix` the prefix is defined, and in the column `URI` the whole link is written (Table 1).

Table 1: Prefix sheet. The whole link is written in the column `URI`, and its abbreviation in the column `Prefix`.

Prefix	URI
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#
ex	http://ex.com/
sql	http://w3.org/ns/sql#

Source sheet. Here we specify where the data is taken from (Table 2). It consists of three columns, `ID`, `Feature`, `Value`. The column `Value` contains path to the source data, the format, and optionally the iterator (the loop used to map the data of JSON and XML files). In `Feature` we declare the type of information provided in `Value`. Finally, `ID` refers to its correspondent subject in the Subject Sheet.

Table 2: Source sheet. The information about the source data it's specified, such as where the data is stored and its format. The kind of information is defined in `Feature`, the information itself in `Value`, and to which subject it refers in `ID`.

ID	Feature	Value
PERSON	source	/home/user/data/people.csv
PERSON	format	CSV
SPORT	source	/home/user/data/sports.csv
SPORT	format	CSV

Subject sheet. The subjects of the triples to generate and their correspondent classes are defined in three columns (Table 3). In `ID` is specified an identifier for each subject so it can be referred from other sheets; in `Class`, the class which the subject belongs to; and in `URI`, the template for the URI of the subjects that are to be created. In the latest field, there is a variable part between curly braces that refers to a field in the data (in the first line, `name`, and in the second, `sport`).

Predicate-Object Maps sheet. In this sheet, the triples are defined through the predicates and its correspondent objects (Table 4). The columns `Predicate` and `Object` are responsible for their specification. The kind of data declared in `Object` is defined in `Data type` (e.g. string, float, etc.). When there is a referencing object map, the triple is defined otherwise. There are three fields that are able to specify the join between the object of the new triple and the referenced subject. They specify which is the ID correspondent to the subject to join (`ReferenceID`), and the fields of the source data they share (`InnerRef` for the field of the current triple, and `OuterRef` for the field of the referred subject). These fields are left blank until this case happens. When it does, the aforementioned fields referring to the object are not necessary (`Object` and `Data type`). The last item to specify is which subject each triple belongs to. For that purpose the column `ID` exists. It links each predicate-object to its correspondent subject.

[3]https://doi.org/10.5281/zenodo.3526141

Table 3: Subject sheet. The class of the subject is specified in `Class`, along with the URI that is to be created in `URI` and a unique identifier in `ID`. In the latest, the words between brackets refer to fields in the data.

ID	Class	URI
PERSON	ex:Person	http://ex.com/Person/{name}
SPORT	ex:Sport	http://ex.com/Sport/{sport}

Function sheet. Some languages support the use of transformation functions over the data (e.g. FnO+RML), so the template allow to include an additional sheet to detail these functions (Table 5). The most used are the SQL and GREL functions, but any can be used. The functions are referred from the Predicate Object map sheet or other function row with the identifier specified in `FunctionID`. The function to use is defined in `Function`, and the parameters in `Params` (if there are several, they are written separated by commas).

5 USE CASE: THE BIO2RDF PROJECT

Bio2RDF [3] is an open source project, started in 2008, that integrates heterogeneous sources of biomedical data into Linked Data. For each biological database in its catalogue, Bio2RDF provides an ontology and a PHP script to transform data into RDF. With the aim of enhancing the maintainability and understandability of the transformation, we show the first steps to change the RDF transformation methodology from using ad-hoc PHP scripts to declarative mappings using spreadsheets.

In this use case, we create mappings for the datasets of the project that have their data published as CSVs and relational databases. With the information provided by the PHP sripts and the source data, the mapping rules are specified in the spreadsheets. Then, they are translated into the most suitable mapping language depending on the format of the data source, and which engine is used to build the knowledge graph. In this specific case, we translate them into R2RML for relational databases and RML for CSVs.

For most of the data sources more than one subject is created, or the database is distributed in several files, or there is a high number of triples (predicate-object maps) to generate. Moreover, there are joins between the subjects within the same and in others datasets. The need to represent so many mapping rules arises the necessity to visualize them quickly, and write the repetitive parts of the mappings easily, which can be done thanks to the structure and functions of the spreadsheets. Moreover, the fact that the spreadsheets are an intermediate step in the mapping creation process makes it possible to write the transformation rules only once, and translate it into one or more languages. The tool developed to perform the translation, Mapeathor, is still under development, and

Table 4: Predicate-Object Map sheet. Here there are specified the Predicates (`Predicate`), Objects (`Object`), kind of data of the object (`DataType`), the references to other subjects (`ReferenceID`, `InnerRef`, `OuterRef`) and the subject that forms the triple (`ID`).

Predicate	Object	DataType	ReferenceID	InnerRef	OuterRef	ID
ex:name	{name}	string				PERSON
ex:birthdate	{birthdate}	date				PERSON
ex:sport			SPORT	sport_id	id	PERSON
ex:name	{sport}	string				SPORT
ex:code	{id}	integer				SPORT
ex:comment	<Fun1>					SPORT

Table 5: Function sheet. The function `sql:upper` is specified. It only takes one parameter, the field `sport` from the source data.

FunctionID	Function	Params
<Fun1>	sql:upper	{sport}

it is available in GitHub[4], along with the spreadsheets mappings created for this use case.

6 CONCLUSIONS AND FUTURE WORK

This paper shows a first approach to design a template spreadsheet able to specify the mapping rules used to create knowledge graphs. The full design is described in detail to show all the essential elements contained in a mapping file that can be specified in a spreadsheet in a language-independent manner. Moreover, we present a real use case in which the use of spreadsheets has facilitated the mapping construction and editing.

Both the template spreadsheet and tool developed to translate the spreadsheets to different mapping languages are still under development. Our objective is to keep on improving the template's structure in order to erase the existing influence of the current mapping languages, and make it language-independent. For that purpose, it's necessary to make a design able to contain the essential information to express the mapping rules, and take for each language the necessary elements in the translation.

Moreover, an evaluation has to be carried out to test that using spreadsheets really helps in the mapping creation process, and give some guidelines on how the template can be improved. The tool has to be developed as well, as the template changes, with the aim of being able to translate the spreadsheets to any mapping language.

REFERENCES

[1] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. DBpedia: A nucleus for a web of open data. In *The semantic web*. Springer, 722–735.

[2] Jesús Barrasa Rodríguez, Óscar Corcho, and Asunción Gómez-Pérez. 2004. R2O, an extensible and semantically based database-to-ontology mapping language. (2004).

[3] François Belleau, Marc-Alexandre Nolin, Nicole Tourigny, Philippe Rigault, and Jean Morissette. 2008. Bio2RDF: towards a mashup to build bioinformatics knowledge systems. *Journal of biomedical informatics* 41, 5 (2008), 706–716.

[4] Christian Bizer, Tom Heath, and Tim Berners-Lee. 2011. Linked data: The story so far. In *Semantic services, interoperability and web applications: emerging concepts*. IGI Global, 205–227.

[5] David Chaves-Fraga, Freddy Priyatna, Idafen Perez-Santana, and Oscar Corcho. 2018. Virtual Statistics Knowledge Graph Generation from CSV files. In *Emerging Topics in Semantic Technologies: ISWC 2018 Satellite Events (Studies on the Semantic Web)*, Vol. 36. IOS Press, 235–244.

[6] Oscar Corcho, Freddy Priyatna, and David Chaves-Fraga. 2019. Towards a New Generation of Ontology Based Data Access. *Semantic Web Journal* (2019).

[7] Souripriya Das, Seema Sundara, and Richard Cyganiak. [n. d.]. R2RML: RDB to RDF Mapping Language. https://www.w3.org/TR/r2rml/

[8] Ben De Meester, Pieter Heyvaert, Ruben Verborgh, and Anastasia Dimou. 2019. Mapping Languages: Analysis of Comparative Characteristics. In *1st International Workshop on Knowledge Graph Building*.

[9] Anastasia Dimou, Miel Vander Sande, Pieter Colpaert, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. 2014. RML: A Generic Language for Integrated RDF Mappings of Heterogeneous Data. In *LDOW*.

[10] Pieter Heyvaert, Ben De Meester, Anastasia Dimou, and Ruben Verborgh. 2018. Declarative Rules for Linked Data Generation at Your Fingertips!. In *European Semantic Web Conference*. Springer, 213–217.

[11] Pieter Heyvaert, Anastasia Dimou, Aron-Levi Herregodts, Ruben Verborgh, Dimitri Schuurman, Erik Mannens, and Rik Van de Walle. 2016. RMLEditor: a graph-based mapping editor for linked data mappings. In *European Semantic Web Conference*. Springer, 709–723.

[12] Simon Jupp, Matthew Horridge, Luigi Iannone, Julie Klein, Stuart Owen, Joost Schanstra, Katy Wolstencroft, and Robert Stevens. 2012. Populous: a tool for building OWL ontologies from templates. *BMC bioinformatics* 13, 1 (2012), S5.

[13] Craig A Knoblock, Pedro Szekely, José Luis Ambite, Aman Goel, Shubham Gupta, Kristina Lerman, Maria Muslea, Mohsen Taheriyan, and Parag Mallick. 2012. Semi-automatically mapping structured sources into the semantic web. In *Extended Semantic Web Conference*. Springer, 375–390.

[14] Maxime Lefrançois, Antoine Zimmermann, and Noorani Bakerally. 2017. A SPARQL extension for generating RDF from heterogeneous formats. In *European Semantic Web Conference*. Springer, 35–50.

[15] Franck Michel, Loïc Djimenou, Catherine Faron Zucker, and Johan Montagnat. 2015. Translation of relational and non-relational databases into RDF with xR2RML. In *11th International Confenrence on Web Information Systems and Technologies (WEBIST'15)*. 443–454.

[16] Kunal Sengupta, Peter Haase, Michael Schmidt, and Pascal Hitzler. 2013. Editing R2RML mappings made easy. (2013).

[17] Álvaro Sicilia, German Nemirovski, and Andreas Nolle. 2017. Map-On: A web-based editor for visual ontology mapping. *Semantic Web* 8, 6 (2017), 969–980.

[18] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledge base. *Commun. ACM* 57, 10 (2014), 78–85.

[19] Katy Wolstencroft, Stuart Owen, Matthew Horridge, Olga Krebs, Wolfgang Mueller, Jacky L Snoep, Franco du Preez, and Carole Goble. 2011. RightField: embedding ontology annotation in spreadsheets. *Bioinformatics* 27, 14 (2011), 2021–2022.

[4] https://github.com/oeg-upm/Mapeathor