An Editor that Uses a Block Metaphor for Representing Semantic Mappings in Linked Data

Ademar Crotti Junior[1], Christophe Debruyne[1], Declan O'Sullivan[1]

[1]ADAPT Centre, Trinity College Dublin, Dublin 2, Ireland
{ademar.crotti,christophe.debruyne,declan.osullivan}@adaptcentre.ie

Abstract. The Linked Open Data cloud contains several knowledge bases with overlapping concepts. In order to reduce heterogeneity and enable greater interoperability, semantic mappings between resources can be established. These mappings are usually represented using mapping languages, where visual representations are often used to support user involvement. In prior work, we have proposed a visual representation based on the block metaphor, called Juma, and applied it to uplift mappings. In this paper, we extend its applicability and propose the use of this visual representation for semantic mappings that automatically generate executable mappings between knowledge bases. We also demonstrate the viability of our approach, in the representation of real mappings, through a use case.

Keywords. Linked Data; Visual Representation; Semantic Mappings.

1   Introduction

The Linked Open Data cloud[1] contains several knowledge bases with billions of triples, many of which have overlapping concepts. In order to reduce data heterogeneity and improve interoperability between these knowledge bases one can avail of semantic mappings [3]. Semantic mappings are represented using mapping languages, which can be complex and not as intuitive as one would like [1]. In such cases, visual representations can be used to support user engagement with the mapping task [4].

In previous work [5], we have proposed a visual representation for mappings, called Juma, **J**igsaw p**u**zzles for **ma**ppings. We have applied this representation for uplift mappings and showed how it can automatically generate mappings that can be exported to R2RML or SML representations, two syntactically distinct mapping languages [6-7]. Juma is based on the block (or jigsaw) metaphor that has become popular with visual programming languages – where it is called the block paradigm – such as Scratch[2]. This metaphor allows one to focus on the logic instead of the language's syntax, targeting different types of stakeholders, especially non-experts.

In this paper, we employ the Juma method in representing semantic mappings that generate executable mappings. Executable mappings encode an interpretation of a semantic mapping in a given query language [11]. In this implementation, Juma will

---

[1] http://lod-cloud.net/
[2] https://scratch.mit.edu/

generate SPARQL CONSTRUCT queries from the visual representation of the semantic mappings. The main contributions of this paper are: the use of the block metaphor for semantic mappings that automatically generate executable mappings and a demonstration through a use case.

The remainder of this paper is structured as follows: Section 2 reviews the related work. Section 3 presents our visual representation applied to semantic mappings. A demonstration is presented in Section 4. Section 5 concludes the paper.

2   Related Work

In this section, we briefly discuss the state of the art in visual representations for semantic mappings. A survey [4] defined the main types of representations as tree and graph ones. **Tree representations.** These systems show ontologies as trees, usually side by side. Mappings are represented as lines that connect concepts between the trees. SAMBO [9] is an example of such system. AlignmentVis [1] combines a tree representation of mappings with statistical plots. **Graph representations.** The main visualization in these systems is a graph one. Most systems analyzed combine a graph representation with others. The system VOAR [12] combine graph and tree visualizations. RepOSE [8] only shows mappings as graphs. AgreementMakerLight [10] applies a different approach where ontologies and mappings are shown in a subgraph centered on a selected mapping.

The block metaphor combines a tree representation with visual elements, such as colors and shape, which show users how blocks are related to each other. In this paper, we apply the block metaphor in the representation of semantic mappings that automatically generate executable mappings. To the best of our knowledge, no other work combines a visual representation for semantic mappings and automatic query generation as executable mappings.

3   Using a Block Metaphor in the Representation of Semantic Mappings

In this work, we propose a new implementation of the Juma[3] method applied to semantic mappings[4]. As mentioned before, semantic mappings relate source and target elements from different ontologies, in order to reduce data heterogeneity and improve data interoperability.

We use Google's Blockly API[5] in this implementation. The main interface has menu options on the left-hand side and a workspace on the right-hand side. The menu options provide users with all blocks that can be used in the creation or editing of mappings within our visual representation. The workspace represents the current version of the mapping. The visual representation also uses colors to identity the type of structure that is being created.

In the tool, we make the distinction between simple and complex semantic mappings. Simple mappings relate one entity to another (one-to-one); complex

---

[3] https://www.scss.tcd.ie/~crottija/juma/
[4] A video is available at https://youtu.be/23RhrKbeM50
[5] https://developers.google.com/blockly/

mappings describe relationships between multiple entities (one-to-many, many-to-one, and many-to-many) [3]. Simple mappings have a specific block defining source and target entities with a relation block. This relation can be used to define class equivalence, property equivalence, sub-class, amongst others. Complex mappings rely on a different block that defines source and target elements. The source element defines how to select the attributes that are going to be used in the target element. In this implementation, source and target elements are defined in a similar way. One block is used to define the subject. Then, this block can be associated with predicate/object blocks. Which are used to define the triple patterns found in SPARQL.

Each function has a specific block. For example, to transform minutes in seconds, a block that represents such function would be available in the visual representation. This block is responsible to generate the SPARQL query code needed for such transformation. The same is true for any other function that would use filters or the aggregation of values, for example. We note that all functions within the current implementation of the visual representation have a similar design (with a name, parameters and optional values); and that it is possible to extend it to support other functions. There are also specific blocks for defining blank nodes and classes (see an example of a complex mapping in Section 4). It is also possible to add comments to the blocks in the representation, which can be used to explain matching results coming from the many matching algorithms available. The visual representation also supports zooming in and out, enabling and disabling parts of the mappings - disabled blocks are still visible but not used in the generation of the executable mapping.

The visual representation automatically generates executable mappings in the form of SPARQL CONSTRUCT queries. The reason for using SPARQL being that it has been shown to be able to represent a wide variety of mappings [13]. Furthermore, SPARQL is a W3C recommendation, widely used with many implementations.

Fig. 1 shows an example of a simple class mapping. In this mapping the class `dbpedia:MusicalArtist` is mapped to `mo:MusicArtist` from the Music Ontology[6].
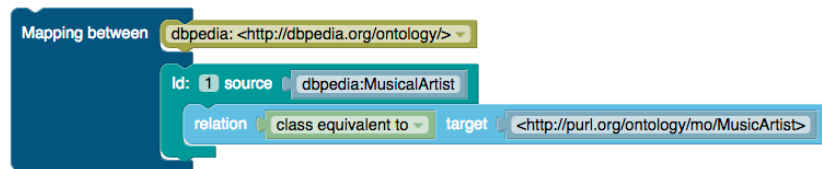

Fig. 1: Visual representation of a simple mapping

4   Demonstration

The dataset used in the demonstration is based on the work published in [2] and consists of mappings between DBpedia and 11 other data sources. These are real use case mappings, designed independently of this research. The mappings were devised by the R2R Framework research [2] (52 simple mappings and 20 complex ones)[7]. We note

---

[6] `http://musicontology.com/specification/`

[7] The mappings are available as SPARQL CONSTRUCT queries at
  `https://www.scss.tcd.ie/~crottija/juma/eswc2018/`

that the current implementation has the built-in functions needed to represent these mappings. In order to evaluate our approach, we represented all mappings using Juma, generated executable mappings, and validated that the executed mappings produced the expected results. Fig. 2 shows an example of a complex mapping from the dataset.
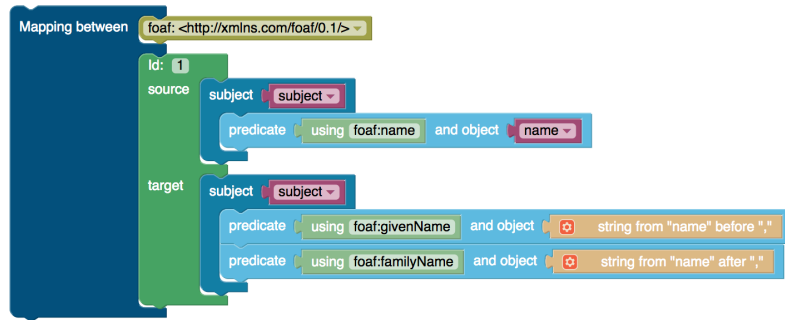

Fig. 2: Visual representation of a complex mapping

In this mapping, the property `foaf:name` is mapped to `foaf:givenName` and `foaf:familyName.` The property `foaf:givenName` uses the function string before. The property `foaf:familyName` uses the function string after. The parameters of the function are defined by clicking on the property icon. The generated executable mapping from in Fig. 2 is presented in Listing 1.

```
prefix foaf: <http://xmlns.com/foaf/0.1/>
#Id1
CONSTRUCT {
      ?subject  foaf:givenName    ?result1 .
      ?subject  foaf:familyName   ?result2 .     }
WHERE {
      ?subject  foaf:name    ?name .
      BIND (STRBEFORE(?name, ",") AS ?result1)
      BIND (STRAFTER(?name, ",") AS ?result2)      }
```
Listing 1: SPARQL CONSTRUCT query generated from a complex mapping

5   Conclusions and Future Work

In this paper, we propose the use of a block metaphor in the representation of semantic mappings. We have demonstrated our approach's expressiveness by applying it to real use case mappings from the R2R framework. By being able to express such mappings, we are encouraged that the approach can cope with simple and complex mappings, but further validation is needed to be conclusive.

In addition, previous work had shown us that this representation was useful, especially for non-experts, in the representation of uplift mappings [6]. Our intuition is that such a visual representation can also be beneficial to different types of users in the representation of semantic mappings. We believe that the combination of our visual representation with others might facilitate the creation, editing and understanding of semantic mappings by experts and non-experts alike.

Future work includes improving the visual representation by making it more intuitive

and extending it to support other executable mapping representations. Moreover, the current implementation requires users to type in the concepts being mapped, with no integration with datasets during mapping creation/editing. Therefore, future work will also focus on minimizing the typing needed, validating it against the ontologies being mapped, and showing the ontologies together with mappings.

References

1. Aurisano, J., Nanavaty, A., Cruz, I.F.: Visual analytics for ontology matching using multi-linked views. In: Proceedings of the International Workshop on Visualizations and User Interfaces for Ontologies and Linked Data (VOILA@ISWC 2015).
2. Bizer, C., Schultz, A.: The R2R framework: Publishing and discovering mappings on the web. In: Proceedings of the First International Workshop on Consuming Linked Data (COLD@ISWC 2010).
3. Euzenat, J., Shvaiko, P..: Ontology matching. Volume 18. Springer (2007).
4. Granitzer, M., Sabol, V., Onn, K.W., Lukose, D., Tochtermann, K.: Ontology alignment - A survey with focus on visually supported semi-automatic techniques. Future Internet 2 (2010).
5. Junior, A.C., Debruyne, C., O'Sullivan, D.: Juma: An editor that uses a block metaphor to facilitate the creation and editing of R2RML mappings. In: The Semantic Web: ESWC 2017 Satellite Events (ESWC 2017).
6. Junior, A.C., Debruyne, C., O'Sullivan, D.: Using a block metaphor for representing R2RML mappings. In: Proceedings of the 3rd International Workshop on Visualization and Interaction for Ontologies and Linked Data (VOILA@ISWC 2017).
7. Junior, A.C., Debruyne, C., O'Sullivan, D.: Juma Uplift: Using a Block Metaphor for Representing Uplift Mappings. In: 12th IEEE International Conference on Semantic Computing (ICSC 2018).
8. Lambrix, P., Ivanova, V.: A unified approach for debugging is-a structure and mappings in networked taxonomies. J. Biomedical Semantics 4 (2013).
9. Lambrix, P., Kaliyaperumal, R.: A session-based approach for aligning large ontologies. In: The Semantic Web (ESWC 2013).
10. Pesquita, C., Faria, D., Santos, E., Neefs, J., Couto, F.M.: Towards visualizing the alignment of large biomedical ontologies. In: Data Integration in the Life Sciences (DILS 2014).
11. Rivero, C.R., Hernandez, I., Ruiz, D., Corchuelo, R.: Generating SPARQL executable mappings to integrate ontologies. In: Conceptual Modelling - ER 2011, 30th International Conference (ER 2011).
12. Severo, B., Trojahn, C., Vieira, R.: VOAR 3.0: a configurable environment for manipulating multiple ontology alignments. In: Proceedings of the ISWC 2017 Posters & Demonstrations and Industry Tracks (ISWC 2017).
13. Thieblin, E., Amarger, F., Haemmerle, O., Hernandez, N., dos Santos, C.T.: Rewriting SELECT SPARQL queries from 1:n complex correspondences. In: Proceedings of the 11th International Workshop on Ontology Matching (OM@ISWC 2016).