Towards Approaches for Generating RDF Mapping Definitions

Pieter Heyvaert, Anastasia Dimou,
Ruben Verborgh, Erik Mannens, and Rik Van de Walle

Ghent University - iMinds - Multimedia Lab,
pheyvaer.heyvaert@ugent.be

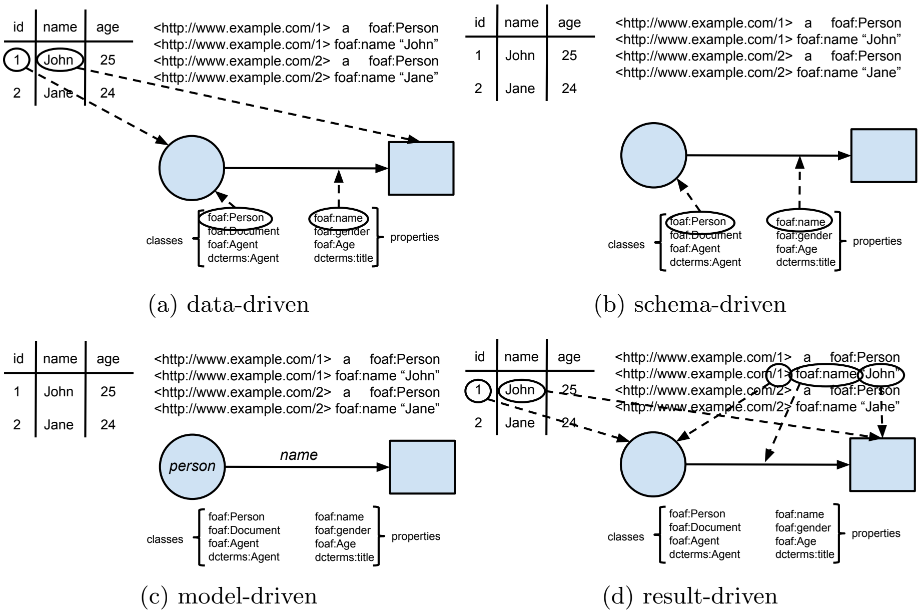
Abstract. Obtaining Linked Data by modeling domain-level knowledge derived from input data is not straightforward for data publishers, especially if they are not Semantic Web experts. Developing user interfaces that support domain experts to semantically annotate their data became feasible, as the mapping rules were abstracted from their execution. However, most existing approaches reflect how mappings are typically executed: they offer a single linear workflow, triggered by a particular data source. Alternative approaches were neither thoroughly investigated yet, nor incorporated in most existing user interfaces for mappings. In this paper, we generalize the two prevalent approaches for generating mappings of data in databases: database-driven and ontology-driven, to be applicable for any other data structure; and introduce two approaches: model-driven and result-driven.

1 Introduction

A substantial amount of Linked Data is generated from data that exists in heterogeneous formats and comes from different sources. This generation process is facilitated by *mapping languages*, such as the W3C recommended R2RML [1] or its extended version RML [2], which separate the definition of mappings from their execution. While data publishers are domain experts—the intended *creators* of mappings—manually creating and editing mapping definitions requires knowledge of the mapping language's syntax, which is unpractical for most publishers [3]. Therefore, user interfaces can facilitate domain experts to specify mappings much more conveniently.

Pinkel et al. [3] introduced two types of approaches for editing mappings for data in relational databases to the Resource Description Framework (RDF), namely (i) the *database-driven* and (ii) the *ontology-driven*, both of which are implemented at fluidOps[1]. The suitability of each approach depends on different factors. However, most existing mapping interfaces, which mainly refer to data in databases, support only one of the two approaches. By doing so, data publishers' editing options are restricted. Alternative approaches beside the two

[1] http://www.fluidops.com/


id
name
age
1
John
25
2
Jane
24
<http://www.example.com/1> a foaf:Person
<http://www.example.com/1> foaf:name "John"
<http://www.example.com/2> a foaf:Person
<http://www.example.com/2> foaf:name "Jane"
classes
foaf:Person
foaf:Document
foaf:Agent
dcterms:Agent
foaf:name
foaf:gender
foaf:Age
dcterms:title
properties
person
name

(a) data-driven (b) schema-driven

(c) model-driven (d) result-driven

Fig. 1: Conceptual comparison of approaches to generate mappings. Input data, vocabularies, mappings and RDF triples are involved in different combinations.

aforementioned ones were not thoroughly investigated so far, even though they might be more adequate under different circumstances.

Moreover, the identified approaches are limited to modeling data in databases. Thus, the implementations completely disregard *heterogeneous* formats. Additionally, these implementations fail to take into account and combine *multiple* data sources [4]. In this paper, we therefore generalize the approaches to *data-driven* and *schema-driven* and introduce two alternatives based on observation: the *model-driven* and the *result-driven* approaches.

2 Approaches

Our goal is to introduce mapping generation approaches that cover more thoroughly the different needs and alternative usage scenarios: ranging from semantically annotating a particular data source to modeling domain-level knowledge. Each of the approaches is described in detail below and visualized in Figure 1.

Data-Driven Mapping Definitions Generation

In the *database-driven* approach [3], data publishers have the data from the database available. The generation of the mappings is based on that data, namely data fractions are iteratively associated to a corresponding mapping rule. For instance, the fluidOps and Karma[2] interfaces follow this approach. However, from a more general point of view, this approach could be applicable to any type of input data, and any combination of them, besides relational databases. Thus, we introduce a more generic approach, so-called *data-driven* (Figure 1a). Instead of only considering data from a database, any number of input data sources in any format, such as CSV, XML, JSON, is equally considered.

Schema-Driven Mapping Definitions Generation

An existing ontology can be used as the basis for generating the mappings. This is the *ontology-driven* approach [3], which is supported by the fluidOps editor. Hence, generating the mappings is driven in the first place by the schema, as it accrues from the ontology. Afterwards, data publishers edit the mappings by associating them to the applicable data from the input source(s). In contrast to the *data-driven* approach where the correct schema(s) is associated to the data, here the appropriate data is associated to the schema(s). While Pinkel et al. [3] consider a single ontology, the approach can be more generic and applied to any schema, namely any combination of ontologies and/or vocabularies. Thus, we consider the more generic notion of *schema-driven* approach (Figure 1b).

Model-Driven Mapping Definitions Generation

Alternatively, data publishers can firstly model the domain, by generating abstract mappings. More precisely, data publishers define the entities, their attributes and their relationships to other entities, without explicitly indicating neither the schema (ontologies and vocabularies) nor the input data fractions to be used. The model is subsequently instantiated by applying adequate schema(s) and it is associated with input data, by specifying which fractions of the input data sources are associated with which parts of the model. To this end, we introduce the *model-driven* approach (Figure 1c). While this approach is practical and useful, for instance applied by De Vocht et al. [5], to the best of our knowledge, no user interface supports it. However, it enables data publishers to formally generate abstract definitions and instantiate them afterwards with the appropriate data and schema(s).

Result-Driven Mapping Definitions Generation

Last, we introduce the *result-driven* approach (Figure 1d), where mappings can be generated based on the desired results. To be more precise, from a desired RDF output, the mappings are generated based on the desired output's model and schema(s). Afterwards, they are associated with data fractions from the input sources. In contrast to the data-driven approach, which is based on the input data and where the appropriate schema is subsequently chosen, this approach is based on the desired result and the model, together with the proper schema(s), is derived from it. A real-world example of this approach is transformy.io[3].

[2] `http://usc-isi-i2.github.io/karma/`

[3] `https://www.transformy.io`

3 Discussion

This paper lists approaches to generate mappings. Besides the ones mentioned, hybrid approaches might emerge when implementing them or as data publishers specify their mappings. Identifying the different approaches, together with their advantages, allows publishers to select the approach best suited for the task at hand. Starting with a particular approach does not necessarily mean that data publishers can/should not switch between approaches over the course of a mappings' editing time. Thus, a user interface should allow and support switching between multiple approaches as suggested by Pinkel et al. [3]. The user interface of our prototype mapping editor, the RMLEditor[4], aims to validate the aforementioned approaches by creating the conditions for data publishers to follow any of them. This is facilitated by simultaneously offering three different panels to data publishers: (i) *Input Panel* (i.e., the input data), (ii) *Modeling Panel* (i.e., the mappings); and (iii) *Results Panel* (i.e., the output RDF dataset).

Acknowledgements. The described research activities were funded by Ghent University, iMinds, the Institute for the Promotion of Innovation by Science and Technology in Flanders (IWT), the Fund for Scientific Research Flanders (FWO Flanders), and the European Union.

References

[1] Souripriya Das, Seema Sundara, and Richard Cyganiak. R2RML: RDB to RDF Mapping Language. Working group recommendation, W3C, September 2012. URL http://www.w3.org/TR/r2rml/.

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

[3] Christoph Pinkel, Carsten Binnig, Peter Haase, Clemens Martin, Kunal Sengupta, and Johannes Trame. How to best find a partner? An evaluation of editing approaches to construct R2RML mappings. In *The Semantic Web: Trends and Challenges*, pages 675–690. Springer, 2014.

[4] Christoph Pinkel, Carsten Binnig, Ernesto Jiménez-Ruiz, Wolfgang May, Dominique Ritze, Martin G Skjæveland, Alessandro Solimando, and Evgeny Kharlamov. RODI: A Benchmark for Automatic Mapping Generation in Relational-to-Ontology Data Integration. In *The Semantic Web. Latest Advances and New Domains*, pages 21–37. Springer, 2015.

[5] Laurens De Vocht, Mathias Van Compernolle, Anastasia Dimou, Pieter Colpaert, Ruben Verborgh, Erik Mannens, Peter Mechant, and Rik Van de Walle. Converging on semantics to ensure local government data reuse. *Proceedings of the 5th workshop on Semantics for Smarter Cities (SSC14), 13th International Semantic Web Conference (ISWC)*, 2014.

[4] http://rml.io/RML_editor.html