Ontology-Based Data Access Mapping Generation using Data, Schema, Query, and Mapping Knowledge[⋆]

Pieter Heyvaert
supervised by Anastasia Dimou, Ruben Verborgh, and Erik Mannens

Ghent University – imec – IDLab
pheyvaer.heyvaert@ugent.be

Abstract. Ontology-Based Data Access systems provide access to non-RDF data using ontologies. These systems require mappings between the non-RDF data and ontologies to facilitate this access. Manually defining such mappings can become a costly process when dealing with large and complex data sources, and/or multiple data sources at the same time. This resulted in different mapping generation tools. While a number of these tools use knowledge from the original data, existing Linked Data, schemas, and/or mappings, they still fall short when dealing with complex challenges and the user effort can be high. In this paper, we propose an approach, together with an evaluation, that discovers and uses extended knowledge from existing (Linked) Data, schemas, query workload, and mappings, and combines it with knowledge provided by the mapping process to generate a new mapping. Our approach aims to improve the mapping quality, while decreasing the task complexity, and subsequently the user effort.

1 Introduction

Nowadays, Linked Data is materialized using RDF, which uses schemas (ontologies and vocabularies) to provide annotations, and is queried using the SPARQL query language [1]. However, Linked Data applications have the need to access data that is available in non-RDF formats [2]. Ontology-Based Data Access (OBDA) systems provides such access where an ontology mediates between the raw data and its consumers [6]. This access requires a mapping between the data schema of the non-RDF data and the ontologies. Subsequently, the aforementioned applications use OBDA systems to access non-RDF data, as if dealing with RDF data. However, manually defining such mappings can become a costly process when dealing with large and complex data sources [3, 4], and/or multiple data sources at the same time [5]. This resulted in the development of (semi-)automatic mapping generation tools. Such tools reduce the user effort during

[⋆] The described research activities were funded by Ghent University, imec, Flanders Innovation & Entrepreneurship (AIO), the Research Foundation – Flanders (FWO), and the European Union.

the mapping process by reducing the required user interaction. This process takes as minimum input the raw data and outputs a mapping that maps this data to RDF triples. The process' tasks include selecting the appropriate classes, predicates, and datatypes, and matching them with the data.

Existing tools for single scenarios only need limited user interaction, but fail on scenarios involving non-trivial data schemas. Automatic tools are developed for use cases where only a mapping for a single scenario is required and where no mappings for subsequent, similar scenarios need to be created (hereinafter referred to as single-scenario use cases). They have a low task complexity [6], as the required user interaction is limited. Subsequently, the mapping process requires a low user effort. In most cases, these tools only use the original data schema and ignore other knowledge available in existing (Linked) **D**ata [7, 8], **S**chemas (data schemas, ontologies and vocabularies), the **Q**ueries that will be executed on the new RDF data (query workload) [2], and **M**appings [9] (DSQM). With existing knowledge, we refer to knowledge that is available before the mapping process. Although these tools are able to generate a promising mapping for simple scenarios, they fail on scenarios with more complex data sources involving non-trivial data schemas (hereinafter referred to as complex challenges) [2].

Semi-automatic tools are developed for both single-scenario uses cases and use cases where multiple mappings need to be created in the same domain (hereinafter referred to as multi-scenario use case). They require more user interaction, such as writing SQL queries [10] or validating a suggested mapping [11], which might improve the generated mapping. However, it increases the required user effort. Despite using more existing knowledge compared to automatic tools, such as mappings and Linked Data, they neglect the query workload.

In this work, we present our semi-automatic approach to improve the quality of single-scenario mappings, while decreasing the user effort, compared to the state of the art. Our approach discovers and uses a more extended set of DSQM knowledge compared to existing tools to deal with complex scenarios, and reduces task complexity. Furthermore, the approach is not limited to a specific data format. The remainder of the paper is structured as follows. In Section 2, we elaborate on the state of the art. In Section 3, we discuss the research questions and the corresponding hypotheses. In Section 4, we explain the methodology and approach. In Section 5, we give preliminary results, followed by the evaluation plan in Section 6. In Section 7, we conclude the paper.

2 State of the Art

In this section, we elaborate on the state of the art for OBDA mapping generation and evaluation.

2.1 Mapping Generation Tools

Existing automated tools require no user interaction and use the data, data schema, and/or target ontology, but neglect the query workload and existing

DSQM knowledge. These tools are based on the direct mappings approach[1], in which tables are mapped to classes, data attributes are mapped to datatype properties, and foreign keys to object properties. They generate a new schema, called a bootstrap ontology, based on the database schema. When an existing ontology, called a target ontology, needs to be used, alignment between the target and bootstrap ontology is required afterwards. D2RQ [12] generates additional rules to tackle more complex challenges not considered by direct mappings. The alignment between the bootstrap and target ontology is done with schema matching tools, such as LogMap [13]. MIRROR [14] and Ontop [15] are similar to D2RQ. However, MIRROR extends the mappings by using information in the databases to determine, e.g., subclass-of relationships and *m:n* relationships. Ontop updates the mappings using T-Mappings [16], which use knowledge embedded in the target ontology. Furthermore, D2RQ, MIRROR, and Ontop neglect the actual data, the query workload, and existing DSQM knowledge. AutoMap4OBDA [17] uses both the data and data schema, together with the target ontology. The ontology alignment is done by the tool itself and does not require an external schema matching tool. However, it neglects the query workload and existing DSQM knowledge. While the aforementioned tools work with relational databases (RDBs), Gloze [18] and JTOWL [19] apply a similar approach for XML and JSON files, respectively.

Existing semi-automated tools use the data, data schema, target ontology, existing mappings, existing Linked Data, and user interaction to create and improve the mapping, but not all information in the mappings is used and the query workload is neglected. BootOX [10] is the only semi-automatic tool for RDBs that applies the direct mappings approach and uses a bootstrap ontology, while IncMap and Karma are not. BootOX deals with more complex mappings than the ones tackled by direct mappings, due to the user interaction. IncMap [11] creates an IncGraph, based on the graph used in the Similarity Flooding Algorithm [20], to represent the data schema and the target ontology. The calculation of the weights of the graph is dynamic and allows to incorporate user feedback to improve results. Karma [21] is different because it uses existing DSQM knowledge to suggest mappings to the user. During a multi-scenario use case information in the previous mappings (called the semantic model) is reused, such as the classes, properties, and how these are related to each other [9]. However, they do not utilize the other information available via the mappings to tackle more complex challenges. If the different scenarios are in different domains previous mappings will have a limited usability. They use graph patterns found in existing Linked Data to determine how the classes and properties are related to each other [8]. This allows support for single-scenario use cases, because for these cases the tool is not able to use previous mappings to get that knowledge. However, the quality of mappings generated by using existing mappings is higher, because they provide a more coherent semantic model than the small graph patterns of the Linked Data. Furthermore, none of these tools validate the correct use of properties and classes.

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

The use of mapping process knowledge in the tools is limited to either the original data, schema and/or ontology, while the query workload is neglected. Furthermore, besides the use of a target ontology for alignment with the bootstrap ontology, only Ontop, AutoMap4OBDA, IncMap, and Karma use knowledge provided by the ontologies to improve the mapping. Karma is also the only tool that uses existing mappings (semantic models and existing data with their corresponding classes and properties) and Linked Data to provide improvements.

Nevertheless, these RDB tools still fall short when tackling more complex challenges, e.g., when subclasses are grouped in a single table and need to be separated, and in real-life scenarios with more complex queries [2, 17].

2.2 Mapping Generation Evaluation

Liu and Li [6] propose a task model to evaluate the task complexity. This complexity is the aggregation of any intrinsic task characteristic that influences the task's performance. The task is in our case the mapping process. The model's components that contribute to the complexity are input (e.g., data, procedures, guidance, and random events), goal/output, process (e.g., steps and actions), time, and presentation (e.g., format and task compatibility). Decreasing the task complexity results in decreasing the required user effort, because less is required from the user (e.g., input, actions, and time).

In most cases, when tools are accompanied with an evaluation, they only assess a limited set of the mapping process' aspects: time required to transform the mapping suggestions to the correct ones [11]; W3C Direct Mapping Test Cases[2] [14]; precision, recall, and CPU time when using the semantic model [9] and Linked Data [8]; or number of user actions [21]. However, they only give a limited insight about task complexity and/or mapping quality, and not every tool is compared with the results of other tools. Furthermore, during these evaluations there is a lack of clear test case descriptions and performance indicators for the mapping quality, and they are different for each tool. Therefore, Pinkel et al. [2] developed a mapping generation quality benchmark for **R**elational-to-**O**ntology **D**ata **I**ntegration scenarios (RODI). Mappings tools are evaluated by assessing the generated mapping's quality, i.e., a comparison between triples generated via the mapping, as a result of given queries, and the expected triples. However, as RODI is focused on automatic tools, it does not provide a formalized way to evaluate the task complexity, which is required when dealing with semi-automatic tools. Furthermore, it works only when the target ontology is used by the tool, while a combination of ontologies might provide better annotations. It is not suited to evaluate tools for other formats, as it only works for RDBs.

3 Problem Statement

In our approach, we aim to improve the mappings of single-scenario use cases, i.e., the precision and recall of the query-answering of the resulting Linked Data

[2] `https://www.w3.org/2001/sw/rdb2rdf/test-cases/`

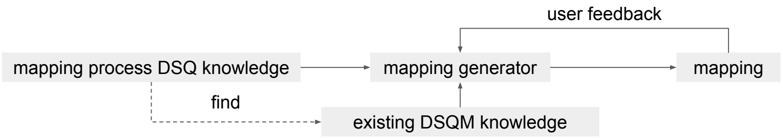

Fig. 1: Overview of the Approach

improves, by using DSQMs, because the DSQMs might contain knowledge that already tackles these challenges and DSQMs have proven benefits [8, 9]. However, we aim to use an extended set of knowledge compared to previous efforts to improve the mappings to tackle these complex challenges. Therefore, we need to discover existing DSQM knowledge, i.e., find the relevant DSQM knowledge that is already available before the mapping process. This leads to the following main research question: *can we improve the (semi-)automatic generation of new single-scenario mappings using existing DSQM knowledge?* To answer this question, we need to answer these subquestions:

- How can we (semi-)automatically discover existing DSQMs that are relevant to the mapping process?
- How can we (semi-)automatically integrate the discovered DSQM knowledge with the DSQ knowledge of the mapping process to generate a new mapping?

These research questions lead to the following hypotheses:

- Using existing DSQM knowledge improves the quality of a new single-scenario mapping compared to the state of the art.
- Using existing DSQM knowledge decreases the task complexity of the mapping process compared to the state of the art.

4 Research Methodology and Approach

Based on the research questions, we need to tackle two aspects: (semi-)automated discovery of relevant existing DSQM knowledge (Section 4.1) and (semi-)automated us of this knowledge to generate mappings (Section 4.2). Both can be addressed separately. The first aspect is not tackled by any of the other tools. For the second aspect, we exploit all options where existing tools are limited to only a subset of the possibilities. The knowledge of the new mapping process is combined with relevant existing knowledge (Figure 1). This results in an initial mapping. Subsequently, user feedback on the mapping, collected via a user interface, is used to improve it. Furthermore, our approach an be used for heterogeneous formats.

4.1 Discover Existing DSQM Knowledge

A high-level overview of the approach to discover the relevant existing DSQM knowledge can be found in Figure 2. The mapping process provides DSQ knowledge (bottom elements). Furthermore, we have existing DSQM knowledge (top

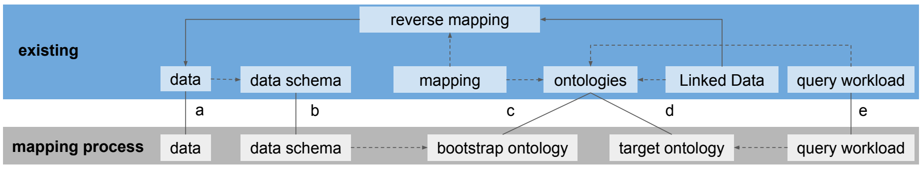

Fig. 2: Discover Existing DSQM Knowledge

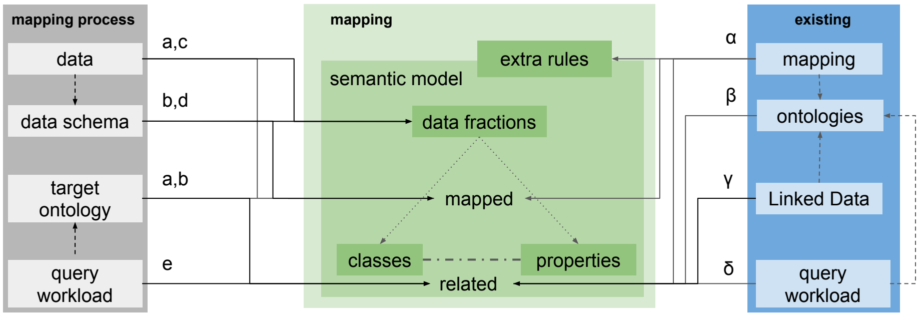

Fig. 3: Use DSQM Knowledge

elements). In the ideal scenario all the DSQM knowledge is at hand. However, this might not always be the case. To address this, we infer knowledge from other knowledge (dashed arrows): based on the original data the data schema can be reconstructed to a certain extent, as done in our previous work [22]; based on the classes and properties used in a mapping [9], Linked Data [8], and/or queries, you can derive the used ontologies. To discover the relevant knowledge we employ algorithms to measure the similarity of other knowledge components (methods a-e in Figure 3). For example, if two data schemas of data sources about persons are similar then the classes (e.g., `foaf:Person`) and properties (e.g., `foaf:name`) of the existing mapping become candiates to be reused for the new mapping (b). Another example is comparing the query workload (e). If the two query workloads contain a query that searches for the graph patterns `?s a foaf:Person. ?s foaf:name 'John Doe'.`, then both mappings will be similar as both will need to annotate entities with the class `foaf:Person` and annotate them with their name using `foaf:name`.

In our approach, we aim to collect as much knowledge as possible to improve the mapping. First, we infer the knowledge that is not at hand. Then, we calculate the similarity measures between the knowledge from the mapping process and the existing knowledge. Finally, based on results of the similarity measures, we select the most relevant knowledge components.

4.2 Use DSQM Knowledge

A high-level overview of the approach to use DSQM knowledge for mapping generation can be found in Figure 3. The mapping process provides DSQ knowledge (left elements). The mapping consists of the semantic model and extra rules (middle elements). The semantic model contains how the used classes and properties are related to each other, and how classes and properties are mapped to the data fractions. The extra rules represent the mapping rules that are needed to tackle mapping challenges that cannot be solved using the direct mappings approach. Furthermore, we have existing DSQM knowledge (right elements). A target ontology can be used together with the data (a) or the data schema (b). However, requiring the user to provide a target ontology is not always straightforward and a combination of classes and properties from different schemas might result in a better annotation of the data. Therefore, we can only use the data (c) and/or data schema (d). However, in this case information from the existing DSQM knowledge is required to complete the mapping. This information can come from mappings (α), ontologies (β), Linked Data (γ), and/or query workloads (δ). When using mappings, we need some information to be adjusted to take into account the specifics of the new mapping process. The ontologies contain classes, properties, and how they are related to each other. However, no information about how they are related to data fractions is provided. This is the same for Linked Datasets and queries.

In our approach, we aim to execute the aforementioned methods separately. Subsequently, we merge each result to improve the mapping. During each merge, we generate adequate mapping rules, while assuring correct use of the ontologies.

5 Preliminary Results

In previous work [23], we developed the RMLEditor. It is a graphical user interface (GUI) that enables non-Semantic Web experts to create their own mappings while limiting the need to understand the underlying mapping language, which is RML [24], or the used (Semantic Web) technologies. In our approach, we aim to use it as a starting point to receive user feedback after each mapping generation iteration. Besides the RMLEditor, we also developed the RMLWorkbench [25]. It is a GUI to support data owners to administrate their Linked Data generation and publication workflow. This includes the data in heterogeneous formats and mappings, which are both used by our approach. Therefore, the RMLWorkbench offers a GUI to administrate the different elements of our approach, while hiding the implementation from the user. Furthermore, we have looked into different modeling approaches to generate mappings [26]. They help describing how different elements of the DSQM knowledge can be used for the generation. We aim to use the approaches separately or combined in our approach. Finally, we developed a tool [22] to effectively perform data analysis on hierarchical data sources to identify RDF terms. This is needed when the schema of the data that needs to be mapped is not available, which might be the case for JSON and XML.

6 Evaluation Plan

Our hypotheses state that the use of DSQM knowledge improves the quality of the mappings, while decreasing the task complexity. Therefore, two aspects need to be evaluated: the quality of the mapping and the task complexity.

6.1 Mapping Quality

To assess the quality of a mapping, we will assess the precision and recall of the query results, because it allows us to compare our approach with existing approaches that do not use certain standards or languages [2]. Our approach needs to be evaluated with different scenarios representing challenges of different complexity, which needs to be reflected in the queries. The benchmark tool for relation-to-ontology mappings RODI [2] contains such a set of scenarios for RDBs with the corresponding queries, designed with real-life challenges in mind. We intend to reuse this benchmark for testing our approach for RDBs. However, to test it against data sources in heterogeneous formats, we need to extend RODI.

6.2 Task Complexity

During our evaluation of the task complexity, we want to apply the model by Liu and Li [6] to our approach, by evaluating each aspect during the mapping process. The input consists mainly of the information and knowledge that needs to be provided by users, e.g., the data, data schema, target ontology, query workload, and the information required during the process. The output is a mapping. The process is defined by the required user actions. The time is the duration to perform these actions. The presentation is defined by the GUI used to complete the actions. To have a mapping process with a low complexity, the input, the actions, and the time to complete these actions needs to be decreased, and the GUI needs to fit the actions. While previous evaluations only analyzed a single component, we want to evaluate all of them to know the complete impact of the mapping process on the task complexity. As RODI is developed for automatic tools, it does not take into account the task complexity. Therefore, we need to extend RODI to also evaluate the different aspects of the task complexity.

7 Conclusion

The main differences of our approach with the state of the art is that we discover relevant DSQM knowledge, and use an extended set of DSQM knowledge, including the found DSQMs, for the generation of OBDA mappings. Challenges for the former include finding the correct similarity metrics and combining these metrics when comparing multiple elements of knowledge. Challenges for the latter include determining how to merge the different knowledge and how to ensure that resulting mapping is valid regarding, e.g., ontology definitions. Furthermore, our approach is not limited to RDBs. It can also be used for JSON and XML data.

If we can validate the hypothesis, then users will have a method that requires less user effort to generate higher quality OBDA mappings, and subsequently, they will have access to higher quality OBDA systems. Even more, Linked Data applications will have access to a larger amount of non-RDF datasets, allowing them to utilize RDF-based techniques on these non-RDF datasets.

References

[1] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data – the story so far. *Semantic Services, Interoperability and Web Applications: Emerging Concepts*, 2009.

[2] Christoph Pinkel, Carsten Binnig, Ernesto Jiménez-Ruiz, Evgeny Kharlamov, Wolfgang May, Andriy Nikolov, Martin G Skjæveland, Alessandro Solimando, Mohsen Taheriyan, Christian Heupel, et al. RODI: Benchmarking Relational-to-Ontology Mapping Generation Quality. *Semantic Web*, 2016.

[3] Evgeny Kharlamov, Dag Hovland, Ernesto Jiménez-Ruiz, Davide Lanti, Hallstein Lie, Christoph Pinkel, Martin Rezk, Martin G. Skjæveland, Evgenij Thorstensen, Guohui Xiao, Dmitriy Zheleznyakov, and Ian Horrocks. Ontology Based Access to Exploration Data at Statoil. In *Proceedings of the 14th International Semantic Web Conference*, pages 93–112. Springer, 2015.

[4] Evgeny Kharlamov, Nina Solomakhina, Özgür Lütfü Özçep, Dmitriy Zheleznyakov, Thomas Hubauer, Steffen Lamparter, Mikhail Roshchin, Ahmet Soylu, and Stuart Watson. How Semantic Technologies Can Enhance Data Access at Siemens Energy. In *Proceedings of the 13th International Semantic Web Conference*, pages 601–619. Springer, 2014.

[5] Bin He, Mitesh Patel, Zhen Zhang, and Kevin Chen-Chuan Chang. Accessing the Deep Web. *Communications of the ACM*, 50(5):94–101, 2007.

[6] Peng Liu and Zhizhong Li. Task complexity: A review and conceptualization framework. *International Journal of Industrial Ergonomics*, 42(6):553–568, 2012.

[7] S.K. Ramnandan, Amol Mittal, Craig A. Knoblock, and Pedro Szekely. Assigning Semantic Labels to Data Sources. In *The Semantic Web – Latest Advances and New Domains (ESWC 2015)*, pages 403–417. Springer, 2015.

[8] Mohsen Taheriyan, Craig A. Knoblock, Pedro Szekely, and José Luis Ambite. Leveraging Linked Data to Discover Semantic Relations Within Data Sources. In *Proceedings of the 15th International Semantic Web Conference*, pages 549–565. Springer, 2016.

[9] Mohsen Taheriyan, Craig A Knoblock, Pedro Szekely, and José Luis Ambite. Learning The Semantics of Structured Data Sources. *Web Semantics: Science, Services and Agents on the World Wide Web*, 37:152–169, 2016.

[10] Ernesto Jiménez-Ruiz, Evgeny Kharlamov, Dmitriy Zheleznyakov, Ian Horrocks, Christoph Pinkel, Martin G. Skjæveland, Evgenij Thorstensen, and Jose Mora. BootOX: Practical Mapping of RDBs to OWL 2. In *Proceedings of the 14th International Semantic Web Conference (Part II)*, pages 113–132. Springer, 2015.

[11] Christoph Pinkel, Carsten Binnig, Evgeny Kharlamov, and Peter Haase. IncMap: Pay As You Go Matching of Relational Schemata to OWL Ontologies. In *Proceedings of the 8th International Conference on Ontology Matching*, pages 37–48. CEUR-WS.org, 2013.

[12] Christian Bizer and Andy Seaborne. D2RQ - Treating Non-RDF Databases as Virtual RDF Graphs. In *Proceedings of the 3rd International Semantic Web Conference*, 2004.

[13] Ernesto Jiménez-Ruiz and Bernardo Cuenca Grau. LogMap: Logic-Based and Scalable Ontology Matching. In *Proceedings of the 10th International Semantic Web Conference (Part I)*, pages 273–288. Springer, 2011.

[14] Luciano Frontino de Medeiros, Freddy Priyatna, and Oscar Corcho. MIRROR: Automatic R2RML Mapping Generation from Relational Databases. In *Engineering the Web in the Big Data Era (ICWE 2015)*, pages 326–343. Springer, 2015.

[15] Diego Calvanese, Benjamin Cogrel, Sarah Komla-Ebri, Roman Kontchakov, Davide Lanti, Martin Rezk, Mariano Rodriguez-Muro, and Guohui Xiao. Ontop: Answering SPARQL Queries over Relational Databases. *Semantic Web*, 8(3): 471–487, 2017.

[16] Mariano Rodrıguez-Muro and Diego Calvanese. Dependencies: Making ontology based data access work in practice. In *Proceedings of the 5th Alberto Mendelzon International Workshop on Foundations of Data Management*, 2011.

[17] Álvaro Sicilia and German Nemirovski. AutoMap4OBDA: Automated Generation of R2RML Mappings for OBDA. In *Proceedings of the 20th Interational Conference on Knowledge Engineering and Knowledge Management, Proceedings*, pages 577–592. Springer, 2016.

[18] Steve Battle. Gloze: XML to RDF and Back Again. In *Jena User Conference*, 2006.

[19] Yuangang Yao, Runpu Wu, and Hui Liu. JTOWL: A JSON to OWL Convertor. In *Proceedings of the 5th International Workshop on Web-scale Knowledge Representation Retrieval & Reasoning*, pages 13–14. ACM, 2014.

[20] Sergey Melnik, Hector Garcia-Molina, and Erhard Rahm. Similarity Flooding: A versatile Graph Matching Algorithm and Its Application to Schema Matching. In *Proceedings of the 18th International Conference on Data Engineering*, pages 117–128. IEEE, 2002.

[21] Craig A. Knoblock, Pedro Szekely, José Luis Ambite, Aman Goel, Shubham Gupta, Kristina Lerman, Maria Muslea, Mohsen Taheriyan, and Parag Mallick. Semi-automatically Mapping Structured Sources into the Semantic Web. In *The Semantic Web: Research and Applications (ESWC 2012)*, pages 375–390. Springer, 2012.

[22] Pieter Heyvaert, Anastasia Dimou, Ruben Verborgh, and Erik Mannens. Data Analysis of Hierarchical Data for RDF Term Identification. In *Proceedings of the Joint International Semantic Technology Conference*, pages 204–212. Springer, 2016.

[23] Pieter Heyvaert, Anastasia Dimou, Aron-Levi Herregodts, Ruben Verborgh, Dimitri Schuurman, Erik Mannens, and Rik Van de Walle. RMLEditor: A Graph-based Mapping Editor for Linked Data Mappings. In *The Semantic Web – Latest Advances and New Domains (ESWC 2016)*, pages 709–723. Springer, 2016.

[24] 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 *Proceedings of the 7th Workshop on Linked Data on the Web*, 2014.

[25] Anastasia Dimou, Pieter Heyvaert, Wouter Maroy, Laurens De Graeve, Ruben Verborgh, and Erik Mannens. Towards an Interface for User-Friendly Linked Data Generation Administration. In *Proceedings of the 15th International Semantic Web Conference: Posters and Demos*, 2016.

[26] Pieter Heyvaert, Anastasia Dimou, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. Towards Approaches for Generating RDF Mapping Definitions. In *Proceedings of the 14th International Semantic Web Conference: Posters and Demos*, 2015.