Modeling, Generating, and Publishing Knowledge as Linked Data

Anastasia Dimou[1], Pieter Heyvaert[2], Ruben Taelman[1], and Ruben Verborgh[1]

Ghent University – imec – IDLab, Belgium
[1]{firstname.lastname}@ugent.be, [2]pheyvaer.heyvaert@ugent.be

Abstract. The process of extracting, structuring, and organizing knowledge from one or multiple data sources and preparing it for the Semantic Web requires a dedicated class of systems. They enable processing large and originally heterogeneous data sources and capturing new knowledge. Offering existing data as Linked Data increases its shareability, extensibility, and reusability. However, using Linking Data as a means to represent knowledge can be easier said than done. In this tutorial, we elaborate on the importance of semantically annotating data and how existing technologies facilitate their mapping to Linked Data. We introduce [R2]RML languages to generate Linked Data derived from different heterogeneous data formats –e.g., DBs, XML, or JSON– and from different interfaces –e.g., files or Web APIs. Those who are not Semantic Web experts can annotate their data with the RMLEditor, whose user interface hides all underlying Semantic Web technologies to data owners. Last, we show how to easily publish Linked Data on the Web as Triple Pattern Fragments. As a result, participants, independently of their knowledge background, can model, annotate and publish data on their own.

Keywords: Linked Data Generation, Linked Data Publishing, [R2]RML, Linked Data Fragments

1 Introduction

Semantic Web technologies offer the possibility to integrate domain-level information from a combination of heterogeneous data sources and published as Linked Data. Thereby, Linked Data gains traction as a prominent solution for machine-interpretable knowledge representation. However, only a limited amount of data is available as Linked Data, as acquiring semantically enriched representations remains complicated, in addition to scalability issues that emerge once Linked Data is published for consumption.

This tutorial aims to show to data owners, who are domain-experts, how to perform the different steps to make their data available as Linked Data, namely modeling, generation, and publication. These steps altogether form a *Linked Data publishing workflow*. By the end of this tutorial, data owners should know how to profit of modeling the knowledge that appears in their data, semantically annotating them to generate corresponding Linked Data and publishing them.

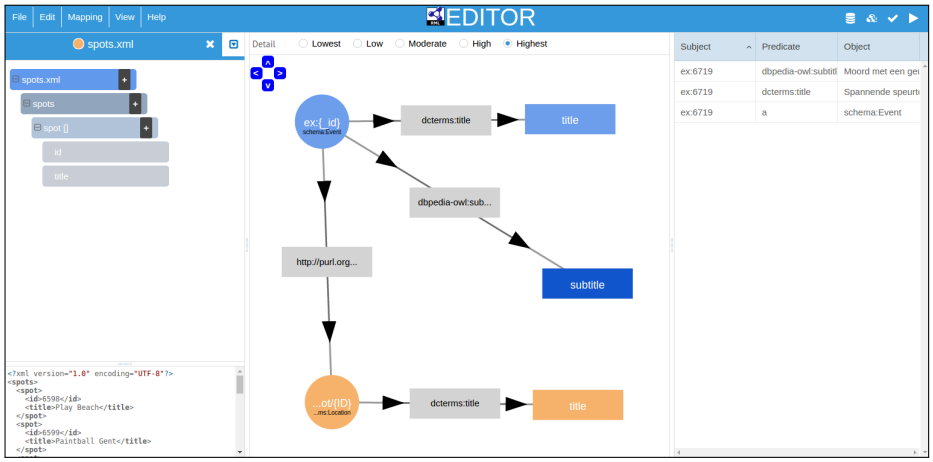

Fig. 1. The graphical user interface of the RMLEditor

The paper is organized as follows: in Section 2, we describe how mapping rules can be defined to specify the generation of Linked Data from raw data. In Section 3, we explain how such mapping rules can be executed to actually generate a Linked Data set. In Section 4 we discuss the publication of Linked Data. In Section 5, the administration of this workflow is described. Last, in Section 6 we explain how the tutorial was given at EKAW2016 conference.

2 Linked Data Modeling

Modeling is the first step of a Linked Data publishing workflow. It involves defining how to make knowledge available as Linked Data. In this step, raw data is modeled and semantically annotated using vocabularies. Data owners indicate:

- the **entities** that appear in the dataset, by assigning *IRIs* to them;
- how the **attributes** are related to the entities, by using *predicates*;
- what the **(data)types** are of these entities, by using *classes*, and of these attributes, by using `xsd`[1] or custom *datatypes*; and
- **relationships** between entities which might originally be in different data sources, by using *predicates*.

Mapping languages allow to declaratively specify the mapping rules which are defined during the modeling step. However, directly editing them using these languages is difficult for data owners who are not Semantic Web experts. Therefore, graphical user interface tools, such as the RMLEditor [14], are developed to ease modeling and support in defining mapping rules to semantically annotate raw data and generate Linked Data by hiding the underlying mapping language.

[1] `https://www.w3.org/TR/swbp-xsch-datatypes/`

The RMLEditor [14] is a tool that enables data owners to define mapping rules that specify how Linked Data is generated using a uniform user interface that presents the data sources, mapping rules, and resulting Linked Data. It is implemented according to the desired features for mapping editors [15]. It is independent of the underlying mapping language, so data owners define mapping rules without knowledge of the language's syntax. Thus, even though the mapping rules execution might be performed directly in the tool for a sample of data, it allows to be exported and, hence, further processed, validated or executed.

The RMLEditor enables users to consider multiple data sources at the same time, as data might be spread across multiple sources. Moreover, it supports data in different formats, as the raw data might reside in, e.g., CSV, JSON, and XML files. As multiple vocabularies can be used, the RMLEditor supports the use of both existing and customs vocabularies. It enables multiple alternative modeling approaches [16], as certain use cases might benefit from using a specific approach and non-linear workflows, thus users are able to keep an overview of the different components which are involved.

The graphical user interface consists of three panels: Input, Modeling, and Results which are aligned next to each other from left to right as seen in Fig. 1. The **Input Panel** shows the data sources, both its structure and a sample of its raw data, and each data source is assigned with a unique color. The **Modeling Panel** presents the mapping rules using a graph-based visualization. The color of each node and edge depends on the data source that is used in that specific rule, if any. The panel offers the means to manipulate the nodes and edges of the graphs to update the rules. The **Results Panel** shows the resulting Linked Data when the mapping rules are executed on the data sources. For each RDF triple of the dataset, it shows the subject, predicate, and object.

The functionality of and the interaction between the panels supports different Linked Data generation approaches that depend on the use case and the data owners' preferences [15]. The *data-driven* approach uses the input data sources as the basis to construct the mapping rules. The classes, properties and datatypes of the schemas are then assigned to the mapping rules. The *schema-driven* approach occurs when data owners start with the vocabularies to define the mapping rules and data values from the data sources fill in the schema. Next, data fractions from the data sources can be associated with the mapping rules.

Semantic annotations can be applied relying on multiple vocabularies. The Linked Open Vocabularies[2] is integrated and can be consulted to get suggestions on which classes, properties and datatypes to use. As the graphs offer a generic representation of the mapping rules, they do not depend on the underlying mapping language. Additionally, the graph visualization and the mapping rules can be exported, allowing the execution of the mapping rules outside the RMLEditor. Additionally, by not restricting when users can interact with which panels, the RMLEditor supports *non-linear workflows*.

[2] http://lov.okfn.org/dataset/lov/

3 Linked Data Generation

The next step in a typical Linked Data life cycle is the *generation* step. Although more and more data is semantically annotated and published as Linked Data, efficiently extracting and integrating information from diverse, distributed and heterogeneous sources to enable Semantic Web based applications remains complicated, let alone keeping track of their provenance. Moreover, the quality and consistency of the resulting Linked Data significantly varies, ranging from expensively curated to relatively low quality Linked Data sets.

Mapping languages enable detaching the mapping rules from the implementation that executes them and specifying in a declarative way how Linked Data is generated from raw data. User interfaces, as the aforementioned RMLEditor, allow defining mapping rules. However, dedicated tools *validate* and *execute* these to generate the desired Linked Data and assure their high quality.

3.1 Generation

Despite the significant number of existing tools, generating Linked Data from data in multiple heterogeneous sources, formats and access interfaces remains complicated. There is still no recommended formalization to define mapping rules regarding how to generate Linked Data derived from such raw data in an integrated and interoperable fashion, except for databases.

More precisely, the R2RML mapping language [4] was recommended by W3C in 2012 to define mapping rules for generating Linked Data but only from data which is derived from relational databases. In 2014, the RDF Mapping language (RML) [8] was proposed as a superset of the W3C-recommended mapping language R2RML, extending its applicability and broadening its scope.

RML is a generic mapping language defined to specify customized mapping rules that generate Linked Data from heterogeneous data in a uniform and integrated way. It is easily extended to cover references to different data structures, combined with case-specific extensions, but remains backward compatible with and follows the same syntax as R2RML. The RML vocabulary namespace is `http://semweb.mmlab.be/ns/rml#` with a suggested prefix of `rml`. More details about RML can be found at `http://rml.io/`.

RML, in contrast to most mapping languages, also considers diverse dataset and services descriptions to access the raw data which is required to generate the desired Linked Data [9]. Corresponding vocabularies which are used to describe how to access the underlying raw data is aligned with the mapping rules definition. Such vocabularies may refer to: (i) the dataset's metadata (e.g., DCAT[3]); (ii) hypermedia-driven Web APIs (e.g., Hydra[4]); (iii) SPARQL services, (e.g., SPARQL-SD[5]); and (iv) database connectivity frameworks.

[3] `https://www.w3.org/TR/vocab-dcat/`
[4] `http://www.hydra-cg.com/spec/latest/core/`
[5] `https://www.w3.org/TR/sparql11-service-description/`

Provenance and other metadata are essential to determine Linked Data set's ownership and trust. Thus, capturing them on every step of the Linked Data publishing workflow is systematically and incrementally required. Thus, declarative and machine-interpretable data descriptions, such as the aforementioned, and mapping rules, such as RML statements, are considered to automatically assert provenance and metadata information related to Linked Data generation [5].

3.2 Validation

Linked Data validation aims at aiding the data owners to acquire high quality Linked Data. The most frequent violations are related to the dataset's schema, namely the vocabularies used to annotate the original data [18]. The dataset's schema, on its own turn, derives from the classes and properties specified in mapping rules, while combining vocabularies increases the likelihood of violations to appear. Hence applying such mapping rules to raw data derived from one or more data sources, results in same violations being repeatedly observed even within the same Linked Data set.

Only recently efforts focus on formalizing Linked Data quality tracking and assessment [27]. Nevertheless, most approaches remain independent of the Linked Data workflow. In this tutorial, we follow a methodology [7] that incorporates systematically the Linked Data validation in the Linked Data workflow. A set of test cases ensures that the same violations are prevented to appear repeatedly within a Linked Data set and over distinct entities, and data publishers can discover violations even before the Linked Data is generated.

This methodology allows to uniformly validate both the mapping rules and the resulting Linked Data. To achieve that, we take advantage of mapping rules, such as RML statements, which are also expressed as Linked Data. Even though mapping rules specify how Linked Data will be formed and can cover many violations related to vocabularies which are used to annotate the data, some schema-related violations depend on how the mapping rules are instantiated on the original data. Thus, validating both the mapping rules and the resulting Linked Data ensures higher quality Linked Data.

We consider two tools in the frame of this tutorial that enable high quality Linked Data generation: (i) the **RMLProcessor** that executes mapping rules expressed in RML to generate Linked Data, and (i) the **RMLValidator** that validates sets of mapping rules expressed in RML or Linked Data.

3.3 RMLProcessor

The RMLProcessor[6] is a tool that enables data owners to execute mapping rules which are defined as RML statements and generate the desired Linked Data. It requires a mapping document as input that summarizes the mapping rules to be executed. Those mapping rules contain also the references to the raw data which are considered to generate the desired Linked Data. The data owners may

[6] `https://github.com/RMLio/RML-Mapper`

choose the preferred RDF serialization when the RMLProcessor is triggered and decide on executing all the mapping rules or a few of them.

Moreover, data owners can automatically generate the corresponding metadata and provenance. The desired metadata can be configured by a data owner who can define the vocabulary to be used and the desired details level. There are different details levels, the data owners may choose among: (i) dataset level, (ii) named graph level, (iii) partitioned dataset (iii) RDF triple level, or (iv) RDF term level. The RMLProcessor considers implicit graphs for metadata which are related with the whole dataset, as well as for named graphs and reification triples if the metadata level is set on RDF triple or term level.

3.4 RMLValidator

The RMLValidator[7] [7] is a tool that enables data owners to validate both their mapping rules that are defined as RML statements and the Linked Data which are generated. It builds on RDFUnit [18] a validation framework for Linked Data. RDFUnit relies its function on pattern-based SPARQL templates that form the different test cases. Those test cases are instantiated for each vocabulary that is used to semantically annotate the data.

As [R2]RML mapping rules can be validated as Linked Data, because they are written as the generated triples and they are defined as Linked Data too. Thus, the same set of schema validation patterns normally applied on the Linked Data is also applicable on the mapping rules that state how the corresponding Linked Data is generated. The data owners need to provide to the RMLValidator (i) a mapping document with RML statements, or (ii) a Linked Data set and they receive a report with the validation results.

4 Linked Data Publishing

The next step in a typical Linked Data publishing workflow is *publication*. In this section, we discuss how Linked Data can be published. We presented the different Linked Data interfaces (Section 4.1); and non-technical tasks related to publishing (Section 4.2).

4.1 Linked Data Interfaces

One of the Linked Data publication main goals is for data to be *retrieved* and *discovered* by machines through HTTP interfaces. Not only the data, but also the interface(s) through which it is published, should be machine-understandable. *Linked Data Fragments* (LDF) [26] was introduced as a framework for comparing different Linked Data publication interfaces on dimensions such as the required server and client effort for querying them. It achieves this by conceptualizing the response of *any* HTTP interface for reading RDF as a *Linked Data Fragment*. Furthermore, this axis has been extended with a data dynamicity axis for

[7] `https://github.com/RMLio/RML-Validator`

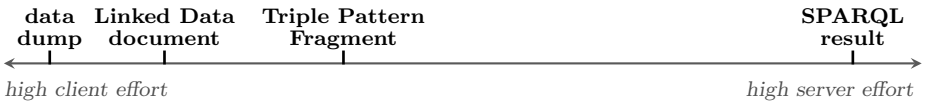

Fig. 2. Conceptual axis showing the server effort needed to query different types of Linked Data Fragments interfaces.

different types of temporal publication interfaces [21], such as TPF-QS [22] and CSPARQL [2]. Next, we discuss 4 types of interfaces using the LDF framework, as shown in Fig. 2.

Data Dump A data dump is a file which contains a Linked Data set that is created once, and typically does not change anymore afterwards. These files can be available in any RDF serialization format, such as Turtle, RDF/XML, or TriG. Such data dumps may easily have sizes in the order of gigabytes. In these cases they may be available in a compressed archive such as gzip or HDT [11]. Data dumps do not provide any querying functionality themselves, it requires clients to download the data dump and load it locally if desired to query it. Hence, significant effort is required from the client to query such Linked Data sets, although little effort from the data owner is required.

Linked Data Document The third Linked Data principle says "When someone looks up a URI, provide useful information, using the open Web standards such as RDF, SPARQL" [3]. When a URI is *dereferenced*, the provided information can be returned as a Linked Data document, containing information about the provided URI. This means that data becomes available in smaller fragments, which can be access on a per-subject basis. Such documents require a bit more effort from the publisher when compared to data dumps, because data needs to be available in fragments. Publication involves low server cost, browsing is easy, and querying such documents is possible by traversing links. However, query result completeness depends on the number of links within the data [13, 25].

SPARQL Query Result SPARQL endpoints are a popular method for exposing Linked Data through an interface [10] that supports queries in the SPARQL query language [12]. Even though they are widely used across the Web, they suffer from significant availability issues [20]. The potentially high complexity of SPARQL queries, together with the public nature of query engines on the Web cause a very high load on servers that expose SPARQL access, which can lead to downtime. This makes SPARQL query engines a costly approach for publishing Linked Data. However, the required effort for clients to query these endpoints is very low, because the server performs the entire query evaluation process.

Triple Pattern Fragments The Triple Pattern Fragments interface (TPF) [26] was introduced as a trade-off between server and client effort for querying. The approach consists of a low-cost server interface that accepts triple pattern queries, while clients then evaluate more complex SPARQL queries. Clients

Linked Data interface	Storage solution
Data dump	RDF file, HDT, ...
Linked Data documents	static or dynamically generated RDF files
TPF	RDF file, HDT, SPARQL engine, ...
SPARQL endpoint	SPARQL engine

Table 1. A mapping of Linked Data interfaces to appropriate storage solutions.

achieve this by splitting up their queries into one or more triple pattern queries, sending them to a TPF interface, and joining their results locally. Experiments show that the TPF requires less effort from the server when compared to SPARQL endpoints [26], at the cost of slower query execution times and increase bandwidth. This approach makes it possible to publish Linked Data at a low cost while still enabling efficient querying, as illustrated by the high availability of DBpedia's TPF endpoint [24].

Depending on the desired interface, different storage solutions might be chosen. Table 1 shows what storage solutions can be used per Linked Data interface.

4.2 Linked Data Licensing, Announcement and Maintenance

In addition to the technical tasks of setting up an interface with an appropriate storage solution, there are also several non-technical tasks related to Linked Data publishing process, such as licensing and announcement which are performed following the best practises for publishing Linked Data [17].

Licensing. In most cases, the goal of Linked Data publishing is to reach Linked *Open* Data. By default, non-licensed data is not open, because regular copyright rules apply. The Open Knowledge Foundation[8] argues that openness is defined by (i) the availability and access of data; (ii) the possibility for anyone to reuse and redistribute the data; and (iii) universal participation, namely that no one should be excluded from these rights. A popular open license is the CC0 license[9], which is in line with the aforementioned definition of openness. To make the license known, it should be mentioned in the dataset listings and as metadata in the dataset using for example the Creative Commons vocabulary[10]. Some cases may require a non-open data license, such as datasets that include confidential data. The publication strategy should then support this license by applying security to the interface. This can be done by adding an authentication and authorization layer to the data-access interface for confidential and private information.

[8] `https://okfn.org/opendata/`
[9] `https://creativecommons.org/publicdomain/zero/1.0/`
[10] `https://creativecommons.org/ns`

Announcement. After the infrastructure for publishing has been set up, the data is ready and properly licensed, its existence must be announced to the public. Depending on the target audience, different communication channels may be of interest, which may include mailing lists, blogs, newsletters. During the announcement, a feedback channel must be in place. This allows data consumers to report issues or ask questions about the Linked Data set, when, for instance, the server would go down. Furthermore, the dataset can also be published on registries, such as `https://datahub.io/`. If the dataset has been properly annotated using dataset vocabularies such as DCAT [19] and VoID [1], crawlers may automatically discover and index a Linked Data set.

5 Linked Data Publishing Workflow Administration

A typical Linked Data workflow consists of the following steps: (i) *modeling* domain knowledge by defining mapping rules that specify how Linked Data may be generated; (ii) *generating* Linked Data by executing the mapping rules; and (iii) *publishing* Linked Data by turning it available for consumption.

Nevertheless, after Linked Data publication, the workflow is not over yet. In fact, it can be seen as a continuous process. Once a data publisher makes new data available, a *social contract* with data consumers is initiated. Data consumers depend on the availability of the published Linked Data, which needs to be guaranteed. This means that dataset and interface removal should be avoided at all costs. In practice, Linked Data sets will rarely remain static over time [23]. Detected violations or inconsistencies, as well as changes to the original raw data needs to be reflected in the Linked Data set.

Ideally, this is accomplished by publishing a new version without having to remove the old one. A good URI strategy should allow multiple dataset versions to exist next to each other. Linked Data set removal should be properly announced and the Linked Data set should then be moved to a different location. Redirects should be put into place, so as existing consumer applications will still work. Finally, an agent should be assigned as responsible for the Linked Data set maintenance and the feedback channel that also needs to be kept open.

We consider the **RMLWorkbench** in the frame of this tutorial that enables administrating and maintaining the complete Linked Data publication workflow, offering a user friendly interface to data owners.

5.1 RMLWorkbench

The **RMLWorkbench** [6] enables user friendly administration of the complete Linked Data publishing workflow in a single place. It offers a graphical user interface consisting of five panels: Access, Retrieve, Generate, Publish, and Schedule.

The **Access Panel** allows data owners to manage the different data sources which are considered to generate Linked Data. It supports data sources that are accessible through (i)local files, (ii) databases or (iii) the Web.

The **Retrieve Panel** enable data owners to specify which data from each data source from the Access Panel is considered for the Linked Data generation.

The **Generate Panel** gives data owners access to different mapping rules. The data owners have several methods to provide mapping rules: (i) upload a mapping document, (ii) specify a Web source with mapping rules, or (iii) create and edit rules via the RMLEditor. These rules can be executed, and the resulting Linked Dataset is available in the Publish Panel.

The **Publish Panel** panel enables data owners to publish the generated Linked Data via an LDF server. Nevertheless, the administrator can easily configure other interfaces, for instance a SPARQL endpoint. The data owners can then choose one or more of them to publish their Linked Data.

Last, the **Schedule Panel** enables users to specify the recurrence of the generation and publication steps. This allows to re-generate the desired Linked Data and keep it up-to-date with the original data.

6 EKAW2016 Tutorial Report

This tutorial was given at the 20th International Conference on Knowledge Engineering and Knowledge Management (EKAW2016). It lasted a whole day and consisted of three parts: (i) a broader introduction on Linked Data generation and publishing; (ii) a more specific introduction on [R2]RML for Linked Data generation and TPF for Linked Data publishing; and (iii) a practical session where participants used the RML tools to model and generate Linked Data and the TPF server and client to publish and query respectively the Linked Data which they generated. The former two were in the morning session, while the practical part was during the afternoon session. The detailed scheduled and the accompanying material are available at `http://rml.io/EKAW2016tutorial.html`. There were approximately twenty participants who are knowledge management or data experts. They were interested in using and profiting of Semantic Web technologies but still faced barriers to publish their own Linked Data.

Acknowledgements

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

Bibliography

[1] Alexander, K., Cyganiak, R., Hausenblas, M., Zhao, J.: Describing Linked Datasets with the VOIDVocabulary. Interest Group Note, W3C (Mar 2011), `https://www.w3.org/TR/void/`

[2] Barbieri, D.F., Braga, D., Ceri, S., Valle, E.D., Grossniklaus, M.: Querying RDF streams with CSPARQL. SIGMOD Rec. 39(1) (Sep 2010)

[3] Berners-Lee, T.: Linked Data (Jul 2006), http://www.w3.org/DesignIssues/LinkedData.html

[4] Das, S., Sundara, S., Cyganiak, R.: R2RML: RDB to RDF Mapping Language. Working group recommendation, W3C (Sep 2012), http://www.w3.org/TR/r2rml/

[5] Dimou, A., De Nies, T., Verborgh, R., Mannens, E., Van de Walle, R.: Automated Metadata Generation for Linked Data Generation and Publishing Workflows. In: Auer, S., Berners-Lee, T., Bizer, C., Heath, T. (eds.) Proceedings of the 9th Workshop on Linked Data on the Web. CEUR Workshop Proceedings, vol. 1593 (Apr 2016)

[6] Dimou, A., Heyvaert, P., Maroy, W., De Graeve, L., Verborgh, R., Mannens, E.: Towards an Interface for User-Friendly Linked Data Generation Administration. In: Kawamura, T., Paulheim, H. (eds.) Proceedings of the 15th International Semantic Web Conference: Posters and Demos. CEUR Workshop Proceedings, vol. 1690 (Oct 2016)

[7] Dimou, A., Kontokostas, D., Freudenberg, M., Verborgh, R., Lehmann, J., Mannens, E., Hellmann, S., Van de Walle, R.: Assessing and Refining Mappings to RDF to Improve Dataset Quality. In: Arenas, M., Corcho, O., Simperl, E., Strohmaier, M., d'Aquin, M., Srinivas, K., Groth, P., Dumontier, M., Heflin, J., Thirunarayan, K., Staab, S. (eds.) The Semantic Web – ISWC 2015. Lecture Notes in Computer Science, vol. 9367, pp. 133–149. Springer (Oct 2015)

[8] Dimou, A., Vander Sande, M., Colpaert, P., Verborgh, R., Mannens, E., Van de Walle, R.: RML: A Generic Language for Integrated RDF Mappings of Heterogeneous Data. In: Proceedings of the 7th Workshop on Linked Data on the Web. CEUR Workshop Proceedings, vol. 1184 (Apr 2014)

[9] Dimou, A., Verborgh, R., Vander Sande, M., Mannens, E., Van de Walle, R.: Machine-Interpretable Dataset and Service Descriptions for Heterogeneous Data Access and Retrieval. In: Proceedings of the 11th International Conference on Semantic Systems. pp. 145–152 (Sep 2015)

[10] Feigenbaum, L., Todd Williams, G., Grant Clark, K., Torres, E.: SPARQL 1.1 Protocol. Recommendation, W3C (Mar 2013), http://www.w3.org/TR/sparql11-protocol/

[11] Fernández, J.D., Martínez-Prieto, M.A., Gutiérrez, C., Polleres, A., Arias, M.: Binary RDF representation for publication and exchange (HDT). Web Semantics: Science, Services and Agents on the World Wide Web 19, 22 – 41 (2013)

[12] Harris, S., Seaborne, A., Prud'hommeaux, E.: SPARQL 1.1 Query Language. Recommendation, W3C (Mar 2013), http://www.w3.org/TR/sparql11-query/

[13] Hartig, O., Bizer, C., Freytag, J.C.: Executing SPARQL Queries over the Web of Linked Data. In: Bernstein, A., Karger, D.R., Heath, T., Feigenbaum, L., Maynard, D., Motta, E., Thirunarayan, K. (eds.) Proceedings of the 8th International Semantic Web Conference. pp. 293–309. Springer Berlin Heidelberg (2009)

[14] Heyvaert, P., Dimou, A., Herregodts, A.L., Verborgh, R., Schuurman, D., Mannens, E., Van de Walle, R.: RMLEditor: a Graph-based Mapping Editor for Linked Data Mappings. In: Sack, H., Blomqvist, E., d'Aquin, M., Ghidini, C., Ponzetto, P.S., Lange, C. (eds.) The Semantic Web – Latest Advances and New Domains (ESWC 2016), Lecture Notes in Computer Science, vol. 9678, pp. 709–723. Springer (May 2016)

[15] Heyvaert, P., Dimou, A., Verborgh, R., Mannens, E., Van de Walle, R.: Towards a Uniform User Interface for Editing Mapping Definitions. In: Proceedings of the 4th Workshop on Intelligent Exploration of Semantic Data. CEUR Workshop Proceedings, vol. 1472 (Oct 2015)

[16] Heyvaert, P., Dimou, A., Verborgh, R., Mannens, E., Van de Walle, R.: Towards Approaches for Generating RDF Mapping Definitions. In: Proceedings of the 14th International Semantic Web Conference: Posters and Demos. CEUR Workshop Proceedings, vol. 1486 (Oct 2015)

[17] Hyland, B., Atemezing, G., Villazón-Terrazas, B.: Best Practices for Publishing Linked Data. Working Group Note, W3C (Jan 2014), http://www.w3.org/TR/ld-bp/

[18] Kontokostas, D., Westphal, P., Auer, S., Hellmann, S., Lehmann, J., Cornelissen, R., Zaveri, A.: Test-driven Evaluation of Linked Data Quality. In: Proceedings of the 23rd International Conference on World Wide Web. pp. 747–758 (2014)

[19] Maali, F., Erickson, J.: Data Catalog Vocabulary (DCAT). W3CRecommendation, W3C (Jan 2014), http://www.w3.org/TR/vocab-dcat/

[20] Pérez, J., Arenas, M., Gutierrez, C.: Semantics and Complexity of SPARQL, pp. 30–43. Springer Berlin Heidelberg, Berlin, Heidelberg (2006)

[21] Taelman, R.: Continuously self-updating query results over dynamic heterogeneous linked data. In: The Semantic Web. Latest Advances and New Domains: 13th International Conference, ESWC 2016, Heraklion, Crete, Greece, May 29 – June 2, 2016, Proceedings. pp. 863–872. Springer International Publishing (May 2016)

[22] Taelman, R., Verborgh, R., Colpaert, P., Mannens, E.: Continuous client-side query evaluation over dynamic linked data. In: The Semantic Web: ESWC 2016 Satellite Events, Heraklion, Crete, Greece, May 29 – June 2, 2016, Revised Selected Papers. pp. 273–289. Springer International Publishing (May 2016)

[23] Umbrich, J., Decker, S., Hausenblas, M., Polleres, A., Hogan, A.: Towards Dataset Dynamics: Change Frequency of Linked Open Data Sources. In: Proceedings of the WWW2010 Workshop on Linked Data on the Web. CEUR Workshop Proceedings, vol. 628 (Apr 2010)

[24] Verborgh, R.: DBpedia's Triple Pattern Fragments: Usage Patterns and Insights. In: Gandon, F., Guéret, C., Villata, S., Breslin, J., Faron-Zucker, C., Zimmermann, A. (eds.) The Semantic Web: ESWC 2015 Satellite Events. Lecture Notes in Computer Science, vol. 9341, pp. 431–442. Springer (Jun 2015)

[25] Verborgh, R.: Piecing the puzzle – Self-publishing queryable research data on the Web. In: Proceedings of the 10th Workshop on Linked Data on the Web (Apr 2017)
[26] Verborgh, R., Vander Sande, M., Hartig, O., Van Herwegen, J., De Vocht, L., De Meester, B., Haesendonck, G., Colpaert, P.: Triple Pattern Fragments: a Low-cost Knowledge Graph Interface for the Web. Journal of Web Semantics 37–38, 184–206 (2016)
[27] Zaveri, A., Rula, A., Maurino, A., Pietrobon, R., Lehmann, J., Auer, S.: Quality assessment for linked data: A survey. Semantic Web Journal (2015)