High Quality Linked Data Generation from Heterogeneous Data

De generatie van kwaliteitsvolle gelinkte data uit heterogene gegevens

Anastasia Dimou

31 oktober 2017

Table of Contents

List of Figures

List of Tables

List of Acronyms

API	Application Programming Interface
ADMS	Asset Description Metadata Schema
B2B	Business-to-business
BiBO	BiBliographic Ontology
CSV	Comma Separated Values
CSS	Cascading Style Sheets
CDFLGD	Contact Details of Flemish Local Governments Dataset
CERIF	Common European Research Information Format Ontology
COMBUST	Combining Multi-trUst BusinesS daTa
CSV	Comma Separated Values
CSVW	CSV on the Web
CWA	Closed World Assumption
DBLP	Digital Bibliography & Library Project
DBO	DBpedia ontology
DCAT	Dta CATalog vocabulary
DCMI	Dublin Core Metadata Initiative
DQA	Data Quality Assessment
DQTP	Data Quality Test Pattern
DWH	Data WareHouse
EBNF	Extended Backus-Naur Form
ESWC	European/Extended Semantic Web Conference
ETL	Extract, Transform, Load
EWI	Economie, Wetenschap en Innovatie
FaBiO	FRBR-aligned Bibliographic Ontology
FOAF	The Friend Of A Friend ontology
FnO	Function Ontology
GAV	Global-As-View
GIM	Geographic Information Management
GML	Geography Markup Language
GRDL	Gleaning Resource Descriptions from Dialects of Languages
GREL	General Refine Expression Language
GR	Good Relations ontology
GSoC	Google Summer of Code
GUI	Graphical User Interface
HDT	Header-Dictionary-Triples
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol

IRI	Internationalized Resource Identifier
ISO	International Organization for Standardization
ISWC	International Semantic Web Conference
JAPE	Java Annotation Patterns Engine
JDBC	Java DataBase Connectivity
JSON	JavaScript Object Notation
JSON-LD	JSON for Linked Data
KML	Keyhole Markup Language
LAV	Local-As-View
LDOW	Linked Data On the Web
LD	Linked Data
LOD	Linked Open Data
LIMES	LInk discovery framework for MEtric Spaces
LOV	Linked Open Vocabularies
LOCN	LOCatioN Core Vocabulary
MQA	Mapping Quality Assessment
N3	Notation3
NER	Named Entity Recognition
ODBC	Open DataBase Connectivity
ORG	ORGanization ontology
OSLO	Open Standards for Local Administrations
OWL	Web Ontology Language
PDF	Portable Document Format
PROV-O	PROV Ontology
RDF	Resource Description Framework
RDFa	RDF Annotations
RDFS	RDF Schema
RSS	Really Simple Syndication
SHACL	Shapes Constraint Language
SKOS	Simple Knowledge Organization System
SML	Sparqlification Mapping Language
SPAR	Semantic Publishing And Referencing ontologies
SPARQL	SPARQL Protocol and RDF Query Language
SPARQL-SD	SPARQL Service Description
SPC	Semantic Publishing Challenge
SPIN	SPARQL Inferencing Notation
SQL	Structured Query Language
SSD	Solid State Disk
SWC	Semantic Web Conference
SWDF	Semantic Web Dog Food
SWRC	Semantic Web for Research Communities
SVM	Support Vector Machine
TDT	The DataTank
TPF	Triple Pattern Fragments
Turtle	Terse RDF Triple Language
URI	Universal Resource Identifier
URL	Universal Resource Locator

URN	Universal Resource Name
VoID	Vocabulary of Interlinked Datasets
W3C	World Wide Web Consortium
WSDL	Web Service Definition Language
WSMO	Web Service Modeling Ontology
WWW	World Wide Web
X3ML	EXtensible Mapping Language
XHTML	Extensible HyperText Markup Language
XML	Extensible Markup Language
XSD	XML Schema Definition
XSL	EXtensible Stylesheet Language
XSLT	EXtensible Stylesheet Language Transformations

Summary

Nowadays, the Web acts as a principal driver of innovation, shifting us to a Web-dominated world in which various data sources together give a fully integrated view of distributed information. The emergence of the Web revolutionized the access to data, but this would be disputable if relevant information could not be easily and quickly found and integrated. Web applications need to handle the distributed aspects of the Web, the issues that arise from mutual information sharing, and the heterogeneity and uncertainty in a flexible and efficient fashion. To achieve it, efficiently extracting and integrating information from diverse, distributed, and heterogeneous data sources is needed to generate such rich knowledge.

Semantic Web enabled technologies become more mature lately and Linked Data is gaining traction as a prominent solution for knowledge representation and integration on the Web. The Semantic Web is an extension of the current Web. It adds logic to it by bringing structure to the content of Web pages, turning it meaningful for software agents as it already is for humans. Linked Data refers to a set of best practices for publishing and connecting structured data on the Web. Its meaning is explicitly defined, is machine-readable, linked to other external Linked Data sets, and can in turn be linked.

Nevertheless, the real power of the Semantic Web will be realized once a significant number of software agents that require information derived from diverse sources, become available. However, software agents still have limited ability to interact with heterogeneous data. Intelligent software agents which function with Semantic Web technologies do not have enough data to work with, and human agents do not want to put in effort to provide Linked Data until there are software agents that use it. This PhD thesis proposes a set of complementary techniques, each one addressing a part of the semantically-enhanced, interrelated and integrated information acquisition from (semi-)structured heterogeneous data. The uppermost goal is to facilitate high quality Linked Data generation independently of the available original data. How each part contributes is attested by evaluating the execution's performance and validation's results, and applying it in different use cases.

Firstly, we identify the barriers that turn Linked Data generation into a complicated task. Then we outline current solution's limitations and outline the requirements to easily generate Linked Data and integrate knowledge from multiple and heterogeneous data sources. We provide a solution, in the form of a mapping language, RML, that allows agents to declaratively define mapping rules that specify how Linked Data is generated and a tool with a user friendly interface, the RMLEditor, which facilitates the editing for human agents.

Decoupling the mapping rules representation from the implementation that executes them, requires designing algorithms and implementing corresponding systems which efficiently

execute those mapping rules. To achieve that, we firstly investigate factors which determine alternative approaches for executing rule-based Linked Data generation. Then, we describe in details the RMLMapper, a tool for Linked Data generation from heterogeneous data, and present results of its evaluation over heterogeneous data.

Besides mapping rules declaration and execution, the quality assessment aspects of Linked Data generation is addressed in this PhD thesis. To achieve that, we introduce a methodology that applies the quality assessment to mapping rules that generate Linked Data, and allows to refine them, instead of assessing the quality of the generated Linked Data. We describe how useful this methodology is to generate Linked Data of higher quality and we provide a corresponding implementation, the RMLValidator.

Moreover, we look into workflows which altogether smoothen the generation of high quality Linked Data. We firstly present a complete workflow for Linked Data generation, then we shed more light with respect to the workflow for refining mapping rules based on the mapping rules and Linked Data quality assessment results, as well as the workflow for tracking metadata and provenance information when Linked Data generation is executed. Last, we provide a tool with a user friendly interface, the RMLWorkbench, which facilitates the administration of the Linked Data generation workflow.

We evaluated the execution's performance with respect to accessing heterogeneous data sources and executing mapping rules to generate Linked Data. The results showed that our methodology, namely the declarative mapping rules in combination with their decoupled execution, is applicable for heterogeneous data with diverse sizes and hierarchical data structures of different depths, as well as when it is required to generate large Linked Data set. Its execution time is in the same order of magnitude as for a custom implementation deliberately designed and optimized for a certain data source.

More, we have proven that assessing the mapping rules indicates the root causes of the violations that degrade a Linked Data set's quality, is more efficient in terms of computational complexity, and requires significantly less time to be executed compared to assessing the entire Linked Data set. The results of our evaluation shows that our methodology is applicable to Linked Data sets without native [R2]RML mapping rules, large Linked Data sets, and (iii) Linked Data sets in the whole process of defining their mapping rules.

To attest the results of this PhD thesis, we describe five use cases where the proposed mapping language for mapping rules declaration, the corresponding implementations, i.e., the RMLEditor, RMLMapper, RMLValidator and RMLWorkbench, and our workflows for Linked Data generation are progressively introduced and used. Two use cases are related to semantic publishing, one to open data, one to business-to-business data transactions, based on the Data-as-a-Service (DaaS) principle, and one is related to DBpedia.

to my grandmother,
γιαγιά Ελένη
1931 - 2016

Preface

Ever since I can remember, my father has been intrigued by technology. Despite their limited resources, he and my mother always made sure that our family had the latest technologies available. He paved the way for me towards computer science, and by accomplishing this PhD I think that I surpassed even their uppermost dreams.

The unconventional nature of research and my passion for creativity and adventure were the actual reasons I leaned towards research and chose to pursue a PhD. I always found that research is an art, the expression of a creative act within science. Research results are like a work of art, artistic endeavors that offer contentment and deserve admiration. Researchers are artists who look for challenges and live out of our inspirations. Conducting research requires creativity and passion, but also faith, perseverance and often sacrifice.

As more people read the preface than the rest book, I tried to be honest with my reasons, apologies and gratitude that accompany the way paved toward this PhD dissertation.

There are different aspects that make a PhD happen; for each one of these, there were different persons who influenced this work, each one in their own way. I would like to start by thanking all members of the jury for reviewing the work presented in this dissertation: prof. Chris Blondia, prof. Steven Latré, prof. Erik Mannens, prof. Ruben Verborgh, prof. Jens Lehmann, prof. Philip De Turck, and prof. Bart Goethals.

I would like to thank Erik not only for making sure that there is funding for this research to happen, but also for teaching me how to work on projects, deal with partners and be independent, but, mostly, for protecting my back when needed. I would also like to thank Ruben who taught me the art of research, shared his passion for creative ideas and stood by my side during good and bad moments. Jens, in his own way, taught me how to turn research into successful papers and set the bar high; I would like to thank him for that. Last, I would like to thank Steven for helping me to turn my research into a PhD.

When I joined IDLab, MMLab as it was called back in the days, there were only a few Semantic Web researchers. I would like to thank Sam, Tom and Laurens for guiding me during my first steps, I was glad I had the chance to work with you, as well as Femke O. who I met on the way. Nothing would have happened without the administrative support I enjoyed from Ellen and Laura; many thanks for making my life easier! I would also like to thank Miel and Pieter C. with whom I was a companion on our parallel ways towards our PhDs. Many thanks to Ben, Pieter H. and Wouter with whom I closely collaborated the last years and who actively shaped this and future research on this topic. I feel honored that you believed in the ideas of this PhD and chose to continue this work. I would like to thank all other colleagues with whom I had fruitful brainstorms that shaped this research.

Special thanks to Joachim for keeping us tight and reminding us that we are a team!

But it is not only the colleagues and experts who influenced me in shaping this research. Other people had their own contribution during this PhD but also over the years before.

True friendship as the one I have with my best friends Io, Nantia, Peny and Vivian rarely exists—certainly given the distance that separates us. I would like to thank you girls for being loyally next to me through happy and sad moments during my PhD time and beyond. You always reflected that the strong bond of friendship is not a balanced equation and its most beautiful quality is to understand and being understood. There is also Anna Maria, who I met during my PhD years. I would like to thank you Anna Maria too for being supportive and available, especially for those moments that required a glass of wine or two. It was nice to have a friend physically present with me in Belgium!

Unexpected incidents often happen—sometimes with surprising effects. Writing this book might not have happened without Koen's honest encouragement. Thank you Koen for suddenly invading in my life and supporting me at the disproportionately difficult "last mile" of my PhD. Together I would like to thank you Frédérique for the carefree moments that you intuitively offered, distracting my mind from the daily routine and stressful PhD life.

"I'm your fan" my godfather used to say to me all these years. His infinite admiration inspired me to eagerly try more and more new adventures in my life, perhaps with a sub-conscious desire to impress him once again. I would like to particularly thank him and my godmother, who are always present in my life, aware of their role's symbolic meaning. Not only to bring the presents, as they consistently did, but also to effectively communicate with me and show their interest in every aspect of my life, including my research and PhD.

Of course, I could not leave out my family. There is no friend like a sister; luckily I have Eleni, who became my strongest ally as we grew older. I cannot even imagine where I would be today without my mother, Stella, and father, Christos, their unconditional love, never-failing sympathy and endless patience. They taught me how to strive for what I want to achieve and armed me with the skills which are needed in life. There are no words to express my gratitude, but I would like to thank you for being there for me, full of compassion, care and forgiveness. My deepest apologies for all those moments that I was not physically present, and even more for those that I was but my thoughts were not.

I dedicate this book to my beloved (grand)mother, Eleni, who brought me up with lot of love and affection. Even though she lacked any education herself and never understood what exactly I am doing, she was always open-minded and unconditionally supportive. She will remain in my heart for her kindness, grateful heart and tremendous love for me. Thank you grandma for your unique touch that, in your own way, shaped who I am today.

Anastasia
21 October 2017

Ἓν οἶδα ὅτι ουδὲν οἶδα
Ένα ξέρω, ότι δεν ξέρω

Σωκράτης

The only true wisdom is in knowing you know nothing

Socrates

1

Introduction

In recent years, the Web acts as a principal driver of innovation, shifting us from a world in which a data owner had a central data repository, for instance a database, to a Web-dominated world in which various data sources need to interact and interoperate, in a way that gives a fully integrated view of distributed information [19]. The emergence of the Web revolutionized the access to data, which is achieved nowadays by just specifying a query in a browser, but this explosion of public information would be disputable if relevant information could not be easily and quickly found and integrated.

Nevertheless, enabling rich knowledge that can accurately answer complicated queries and lead to effective decision making, remains one of the most significant challenges. To achieve it, *efficiently extracting and integrating information from diverse, distributed, and heterogeneous data sources* is required to generate such rich knowledge. Therefore, unlike traditional desktop applications, Web applications require the ability to handle the distributed aspects of the Web, the issues that arise from mutual information sharing, and especially to deal with heterogeneity and uncertainty in a flexible and efficient fashion [19].

Data integration is the problem of combining data residing at different sources, and providing a unified view of this data [34]. Nevertheless, data integration on the Web is hard due to different reasons [19]: (i) *systems*, enabling different systems to interoperate is required; (ii) *logical*, even though data sources are organized according to a schema, those schemas, and the data representations in them, are different, thus bridging this *semantic heterogeneity* is required; (iii) *social*, different data owners do not cooperate, as they want to avoid prohibitively heavy load on their system, or the access to certain data is only allowed to certain people; and (iv) *administrative*, finding a set of data sources is required.

Nowadays, Semantic Web enabled technologies become more mature and Linked Data is gaining traction as a prominent solution for knowledge representation and integration on the Web. Therefore, this PhD thesis proposes a set of complementary techniques, each one addressing a part of the semantically-enhanced, interrelated and integrated information acquisition in the form of Linked Data from semi-structured heterogeneous data. The uppermost goal is to facilitate high quality Linked Data generation independently of the available original data. How each part contributes is attested by evaluating the execution's performance and validation's results, and applying it in different use cases, among others open government, scientific research, industry, and generic domain knowledge.

In this chapter, we provide an overview of the Semantic Web (Section 1.1), the Resource Description Framework (Section 1.3), knowledge representation (Section 1.4), Linked Data (Section 1.2) and its life cycles (Section 1.5). More, we state the problem that this PhD thesis deals with and the accompanying hypotheses and research questions (Section 1.6).

1.1 Semantic Web

The Semantic Web is an extension of the current Web. It adds logic to it by bringing structure to the content of Web pages, turning it meaningful for software agents as it already is for humans [8]. To achieve that, information is given well-defined meaning and thus an environment is created where software agents can comprehend and process the information and carry out sophisticated tasks for humans, instead of merely displaying data [8]. The increasing complexities, caused by the accelerating and virtually uncontrolled growth of the World Wide Web, render the need for, not only software agents, but also *intelligent agents* that are even more crucial [25]. The situation can only get more complicated, as new Web technologies permit the further integration of more complex data sources.

To anticipate this extension of the Web, we need to move from the *Web of Documents*, which is the Web as we experience it today, to the *Web of Data*, as the Semantic Web is also called. The fundamental difference is that, currently, Web applications consider the entire document as their primary object, and thus are not able to provide context to data. Software agents do not understand what the data is about, and thus which part of the document is relevant and which one is not, as human agents do. To the contrary, in the case of the Semantic Web, the primary object turns to be the resources (or their description). A resource is identified by a unique identifier, namely an IRI which might identify both a resource and a Web document describing the resource [40]. In more detail:

URI A *Uniform Resource Identifier* (URI) provides a simple and extensible means for identifying a resource [7]. The term *Uniform Resource Locator* (URL) [6] refers to the subset of URIs that, not only identify a resource, but also locate it by describing its primary access mechanism, rather than identifying the resource by name or some other attribute(s). Thus, associating a resource with a URI means that anyone can link, refer, or retrieve a representation of it. The term *Uniform Resource Name* (URN) is used to refer to both URIs under the "urn" scheme [37], which are required to remain globally unique and persistent, even when the resource ceases to exist or becomes unavailable, and to any other URI.

IRI The *Internationalized Resource Identifier* (IRI) is a complement of URI, which extends the syntax of URIs to a much wider repertoire of characters, namely the Universal Character Set (Unicode[1] /ISO 10646[2]) [20]. Any URI can be mapped to an IRI, namely IRIs can be used instead of URIs, where appropriate, to identify resources. The characters in IRIs are frequently used for representing words of natural languages. By design, IRIs have global scope, and, by social convention, its owner denotes what the intended referent of an IRI is [32]. This can be achieved by means of a specification or other document that explains what is denoted, ideally by setting up the IRI, so it dereferences to such a document.

A cool URI does not change, and if it changes, it happens because there is some information in it which changes[3]. To achieve that, it is critical how a URI is designed. Therefore, W3C released an Interest Group Note [40] with informative guidelines for its effective design and use, inspired by Tim Berners-Lee's article "Cool URIs don't change"[4]. According

[1] Unicode, `http://www.unicode.org/`
[2] ISO 10646, `https://www.iso.org/standard/63182.html`
[3] `https://www.w3.org/Provider/Style/URI`
[4] `https://www.w3.org/Provider/Style/URI`

to these guidelines, a URI should (i) *be on the Web*, and agents should be able to retrieve from the Web a description about the resource identified by this URI. Thus, the standard Web transfer protocol, HTTP [23], should then be used, because Web clients and servers use it to uniquely identify and refer to resources; (ii) *be unambiguous*, there should be no confusion among identifiers for different resources, a URI is meant to identify one of them.

There are two solutions for designing URIs that identify resources [40]: 303 URIs and hash URIs. Hash URIs contain a fragment that is separated from the rest of the URI by a hash symbol ("#") and are used for non-document resources, as a resource then cannot be retrieved directly. To the contrary, 303 URIs give an indication that the requested resource is not a regular Web document, relying on the 303 HTTP code (redirect status code), as a 200 code is inappropriate to be returned for a URI. This way, we avoid ambiguity between the original, real-world resource and the Web document that represents it. The hash URIs have the advantage of reducing the number of necessary HTTP round-trips, which, in turn, reduces access latency, whereas 303 URIs are very flexible because the redirection target can be configured separately for each resource. Hash URIs are preferred for rather small and stable sets of resources that evolve together, whereas 303 URIs for making neater-looking URIs, but with an impact on run-time performance and server load.

Nevertheless, to realize the Semantic Web, besides identifying a resource (relying on its URI), retrieving and processing its description is required too. As the Semantic Web aims to turn the content meaningful both for human and software agents, the obtained resources' descriptions should also be machine-processable. To achieve this, a fundamental change is required which would have as crucial impact as, for instance blogs had, leading to the transition from the "*Read-Only Web*" (Web 1.0), where human agents could only read the information which was presented to them, to the "*Read-Write Web*" (Web 2.0), where they can interact with the Web. The fundamental difference between the "*Web of Documents*" and the "*Web of Data*" lies in *providing context to the data*, but how could that be achieved?

1.2 Linked Data

The term *Linked Data* refers to a set of practices for publishing and connecting structured data on the Web to be accessed by both human and software agents. Linked Data is machine-readable, its meaning is explicitly defined, it is linked to other external Linked Data sets, and can in turn be linked to from external Linked Data sets [10].

In 2006, Tim Berners-Lee introduced the Linked Data Principles[5]:

- ✓ Use URIs as names for things
- ✓ Use HTTP URIs so that people can look up those names
- ✓ Provide useful information, using standards (RDF, SPARQL), when looking up a URI
- ✓ Include links to other URIs, so that they can discover more things

[5]Linked Data design issues, `https://www.w3.org/DesignIssues/LinkedData.html`

The Semantic Web is not only about putting data on the Web, in Tim Berners-Lee stated at the same note. It is about making links, so (human or not) agents can explore the Web of data. Breaking the Linked Data principles does not destroy anything, but misses an opportunity to make data interconnected and limits the ways Linked Data can later be reused in unexpected ways. However, the added-value of the Semantic Web is the unexpected re-use of Linked Data. Therefore, in 2010, Tim Berners-Lee added the *five stars of Linked Open Data*[6] to his earlier article on Linked Data design principles to encourage and motivate (human) agents to pave the way towards Linked Data:

- ⋆ make your data available on the Web (whatever format) under an open license
- ⋆⋆ make them available as structured data
- ⋆⋆⋆ make them available in a non-proprietary open format
- ⋆⋆⋆⋆ use URIs to denote things
- ⋆⋆⋆⋆⋆ link your data to other data to provide context

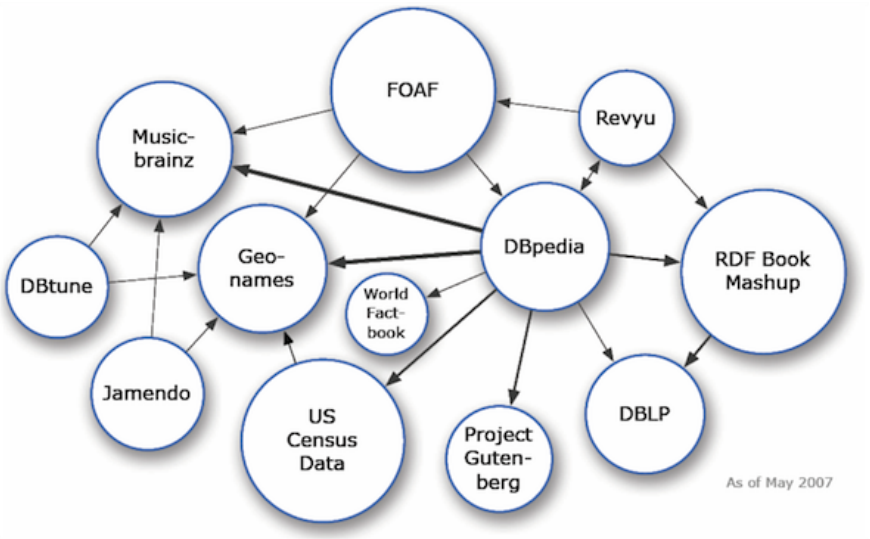


Figure 1.1: *Linked Open Data cloud in 2007*

The Open Data movement aims at making data freely available to everyone, the so-called *Open Data*. Linked Data which is made available to everyone and where different data sets are linked to each other is called *Linked Open Data*. Efforts around Linked Open Data are channeled to enrich the *Linked Open Data cloud*. The Linked Open Data (LOD) cloud[7] consisted of 12 Linked Data sets in 2007, grew to almost 300 in 2011[8], and by the end of 2014 counted up to 1,100[9]. The diagram of the 2007 Linked Open Data cloud is presented in Figure 1.1, to contrast with the one of 2017 which is presented in Figure 1.2.

[6] Five star Linked Data, `http://5stardata.info/en/`
[7] Linked Data cloud, `http://lod-cloud.net/`
[8] Linked Data cloud state (2011), `http://lod-cloud.net/state`
[9] Linked Data state (2014), `http://linkeddatacatalog.dws.informatik.uni-mannheim.de/state/`

The difference in size of the two diagrams shows the evolution of the Linked Open Data cloud. More detailed statistics with respect to the Linked Open Data cloud is available at LODStats[10] and LODLaundromat[11], two well-known Linked Open Data indexes.

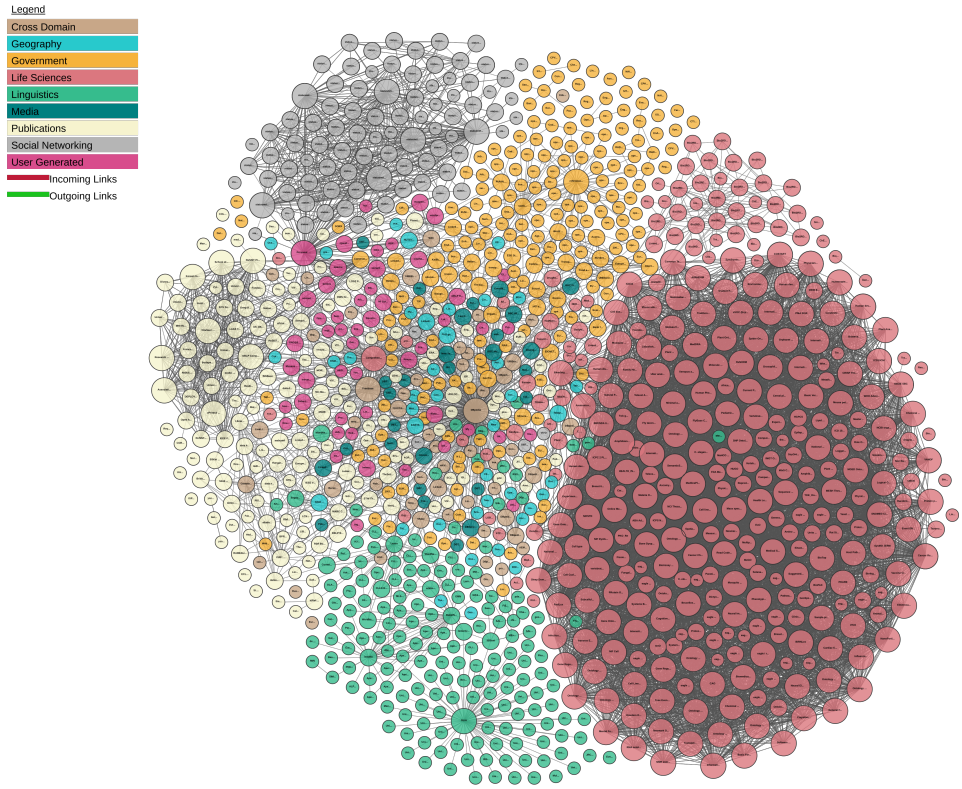


Figure 1.2: *Linked Open Data cloud in 2017*

A series of best practices designed to facilitate development and delivery of open government data as Linked Open Data, but is applicable for other data too, was released by the W3C Government Linked Data Working Group [30]. The best practices are summarized as follows: (i) prepare the stakeholders, (ii) select a data source, (iii) model the data, (iv) specify an appropriate license, (v) define "good URIs" for Linked Data, (vi) use standard vocabularies, (vii) convert data to Linked Data, (viii) provide machine access to data, (ix) announce to the public, and (x) establish social contract of a Linked Data publisher.

While the Linked Data principles remain a bit abstract, barely suggesting *how Linked Data should be*, the aforementioned best practices clarify the methodological steps on *how Linked Data should be obtained*. However, it still remains unclear how the data should be represented to provide the desired *context*. Which data structure should be used?

[10] LODStats, `http://stats.lod2.eu/`
[11] LODLaundromat, `http://lodlaundromat.org/`

1.3 Resource Description Framework

The Resource Description Framework (RDF) [32] is a framework for representing static snapshots of information (*atemporal information*) for resources on the Web. Currently, it is the most well-known and W3C recommended framework for representing Linked Data. Below, we outline some basic concepts and terminology used by RDF:

dataset An RDF dataset is a collection of RDF graphs [32]. An RDF graph, on its own turn, is the conjunction of its triples. It consists of exactly one *default graph* and zero or more other graphs, the so-called *named graphs*. Each RDF graph has an associated IRI or blank node, the *graph name*. The default graph does not have an associated IRI.

document An RDF *document* encodes an RDF graph or dataset in a concrete RDF syntax, and enable the exchange of RDF graphs and datasets between systems.

triple The core structure of RDF's abstract syntax is a set of triples. Each triple consists of a subject, a predicate, and an object. An RDF triple says that some relationship, indicated by the *predicate*, holds between two resources, denoted by the *subject* and *object* (Figure 1.3). The statement that corresponds to an RDF triple, which is a simple logical expression, or claim about the world, is denoted as an RDF *statement*. An RDF triple consists of the subject, which can be an IRI or blank node, the predicate, which can be an IRI, and the object, which can be an IRI, literal or blank node. An RDF graph can be visualized as a node and directed-arc diagram. Each triple is represented as a node-arc-node link.

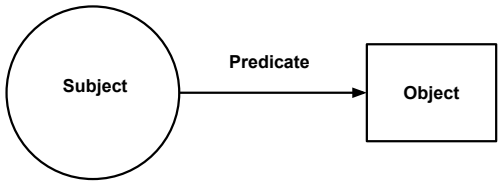


Figure 1.3: *An RDF triple consists of a subject, predicate, and object*

term The set of nodes of an RDF graph is the set of subjects and objects of its RDF triples. An RDF graph's node, called an RDF *term*, can be an IRI, literal, or blank node. An IRI or literal denotes a *resource* (in this context, a resource is synonymous with the term *entity*).

A resource can be anything, for instance, a physical thing, document, abstract concept, etc. A resource denoted by an IRI is called its *referent*, and a resource denoted by a literal is called its *literal value*. Literals have datatypes which define the range of possible values, such as strings, numbers, and dates. IRIs of the `xsd` namespace denote the built-in datatypes. For instance, `"18"^^xsd:integer` denotes a Literal whose value is 18 and its datatype is integer. Certain literals, the *language-tagged strings*, denote plain-text strings in a natural language, and may appear if and only if the literal's datatype is

`rdfs:langString`. For instance, `"Anastasia"@en` denotes my name in Engish, whereas `"Αναστασία"@el` denotes my name in Greek. A *Blank node* [32] does not identify a specific resource, but it is mainly used to denote that a given relationship exists, without explicitly naming the involved resource. Blank nodes are always locally scoped, and do not have persistent or portable identifiers. In situations where stronger identification is needed, a blank node may be replaced with a new, globally unique IRI, the so-called *skolem* IRI.

serializations A concrete RDF syntax may offer many different ways to encode the same RDF graph or dataset, but they are not significant for its meaning. This occurs through the use of namespace prefixes, relative IRIs, blank node identifiers, and different ordering of statements. Bellow, we outline the different RDF serializations which are recommended by W3C. We distinguish based on whether they support *named graphs* or not.

The following representations focus on RDF triples, for each one of the different RDF serializations we provide an example:

RDF/XML RDF 1.1 XML syntax (RDF/XML) [24] defines an XML syntax specification for RDF. To encode the graph in XML, the nodes and predicates have to be represented in XML terms. RDF/XML uses XML QNames, to represent IRIs.

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
             xmlns:dc="http://purl.org/dc/elements/1.1/"
             xmlns:ex="http://example.org/"
             xmlns:foaf="http://xmlns.com/foaf/0.1/">

  <rdf:Description rdf:about="http://example.org/HQ-LDgeneration/">
    <dc:title>High Quality Linked Data Generation</dc:title>
    <dc:creator>
      <rdf:Description foaf:name="Anastasia Dimou">
        <foaf:homepage rdf:resource="http://example.org/andimou/" />
      </rdf:Description>
    </dc:creator>
  </rdf:Description>
</rdf:RDF>
```

Listing 1.1: RDF/XML

```
@base <http://example.org/> .

@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix   dc: <http://purl.org/dc/elements/1.1> .
@prefix   ex: <http://example.org/> .

<#HQ-LDgeneration> dc:creator <#andimou> ;
    dc:title "High Quality Linked Data Generation" .

<#andimou> a foaf:Person ;
    foaf:name "Anastasia Dimou" .
```

Listing 1.2: *Turtle*

Turtle The Terse RDF Triple Language (RDF 1.1 Turtle) [5] is a textual representation of an RDF graph. The simplest RDF triple statement is a sequence of RDF terms, separated by whitespace and terminated by '.' after each RDF triple. The ';' is used to repeat the subject of triples that vary only in predicate and object RDF terms. The ',' is used to repeat the subject and predicate of triples that only differ in the object RDF term.

N-Triples RDF 1.1 N-Triples [14] is an easy-to-parse line-based (subset of Turtle) syntax for an RDF graph. N-Triples are a sequence of RDF terms, representing the subject, predicate, and object of an RDF triple, separated by white space and terminated by a '.' (same as with Turtle). N-Triples are also Turtle triples. Thus, Turtle parsers parse N-Triples and produce the same triples as N-triples parsers. However, Turtle includes other representations of RDF terms and abbreviations of RDF triples. IRIs are written only as absolute IRIs, are enclosed in '<' and '>' and may contain numeric escape sequences. Literals consist of an initial delimiter ", a sequence of permitted characters or numeric/string escape sequence, and a final delimiter. Blank nodes are expressed as _: followed by a series of name characters.

```
1  # comment
2  <http://example.org/HQ-LDgeneration>
3      <http://purl.org/dc/elements/1.1/title> <http://example.org/andimou> .
4  <http://example.org/HQ-LDgeneration>
5      <http://purl.org/dc/elements/1.1/title> "High Quality Linked Data Generation" .
6
7  <http://example.org/andimou> <http://xmlns.com/foaf/0.1/name> "Anastasia Dimou" .
```

Listing 1.3: *N-Triples*

The following representations support *named graphs* too:

```
1  { "@context": {
2            http://json-ld.org/contexts/person.jsonld"
3            "@vocab": "http://purl.org/dc/elements/1.1"},
4    "@id": "http://example.org/HQ-LDgeneration",
5    "title": "High Quality Linked Data Generation",
6    "creator": "http://example.org/andimou" }
```

Listing 1.4: JSON-LD

JSON-LD JSON-LD 1.0 [16] is a lightweight JSON-based serialization for Linked Data, aiming to be easily integrated into deployed systems that already use JSON. This way, a smooth upgrade from JSON to JSON-LD is achieved, as existing JSON parsers and libraries can be reused. In contrast to JSON, in JSON-LD the keys in an object must be unique. Similarly, in JSON, an array is an ordered sequence of zero or more values, while, in JSON-LD, the collection is unordered by default. In JSON-LD, an entity is identified using the `@id` keyword, and its type using the `@type` keyword. A common prefix to be used for all properties is set using the `@vocab` keyword. *Context* is used to map terms, i.e., simple strings, to IRIs. It can be directly embedded (inline) into the document or referenced. A JSON object which is associated with a term, is called an *expanded term definition*. JSON documents can be interpreted as JSON-LD without having to be modified by referencing a context via an HTTP Link Header or applying a custom context using the JSON-LD API.

TriG RDF 1.1 TriG [12] is a textual syntax for RDF that allows RDF to be written in a compact and natural text form, with abbreviations for common usage patterns and datatypes. TriG is an extension of Turtle to meet the requirement for multiple graphs, thus it allows any constructs from the Turtle language. RDF graph statements are a pair of an IRI or blank node label and a group of triple statements surrounded by '{}'. A graph statement that is not labeled with an IRI belongs to the default graph of an RDF dataset.

```
1  # This document contains a default graph and a named graph.
2
3  @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
4  @prefix dc: <http://purl.org/dc/terms/> .
5  @prefix foaf: <http://xmlns.com/foaf/0.1/> .
6
7  # default graph
8      {
9        <http://example.org/HQ-LDgeneration> dc:creator <http://example.org/andimou> .
10       <http://example.org/HQ-LDgeneration> dc:title "High Quality Linked Data Generation" .
11     }
12
13 <http://example.org/andimou>
14     { _:a foaf:name "Anastasia Dimou" . }
```

Listing 1.5: T*ri*G

N-Quads RDF 1.1 N-Quads [13] is a line-based plain-text syntax for RDF. N-Quads is similar to N-Triples, but N-Quads allows encoding multiple graphs too. N-Quads are a sequence of a subject, predicate, object, and an optional blank node label or IRI defining the graph in an RDF dataset where the triple belongs to, all are separated by whitespace and terminated by '.' after each statement.

```
1  <http://example.org/HQ-LDgeneration> <http://purl.org/dc/terms/title>
2      "High Quality Linked Data Generation" <http://ex.org/thesis> .
3  <http://example.org/HQ-LDgeneration> <http://purl.org/dc/terms/creator>
4      <http://example.org/andimou> <http://ex.org/thesis> .
5
6  <http://example.org/andimou> <http://xmlns.com/foaf/0.1/name>
7      "Anastasia Dimou" <http://ex.org/students> .
```

Listing 1.6: *NQuads*

1.4 Knowledge Representation

So far, we have discussed that, in order for the Semantic Web to be materialized, Linked Data is required which, on their own turn, can be described, relying on the RDF framework. However, we have not discussed yet *how the "context" is added to the data?*

Semantic annotation is the process of attaching additional information, i.e., context, to an entity in certain content. A semantic annotation is described relying on vocabularies or ontologies terms. A *vocabulary* is a set of central terms with fixed meaning. Besides a concise natural language definition, the meaning of a term can be characterized by stating how this term is interrelated to the other terms. An *ontology* is a set of precise descriptive statements about some part of the world (usually referred to as the domain of interest or the subject matter of the ontology) [28]. An ontology describes the types of things that

exist (classes), the relationships between them (properties), and the logical ways those classes and properties can be used together (axioms) [29]. Ontologies (and vocabularies) are the means to make an explicit commitment to shared meaning among an interested community, but anyone can use these ontologies to describe their own data. Similarly, anyone can extend or reuse elements of an ontology if they so wish [42].

However, there is no clear distinction between the two terms: ontology and vocabulary[12]. The term "ontology" is used for more complex, and possibly formal collection of terms and it is more often described using OWL (a semantic extension of RDF). The OWL 2 Web Ontology Language (OWL 2), is an ontology language for the Semantic Web with formally defined meaning [28]. To the contrary, the term "vocabulary" is used when such strict formalism is not necessarily or only in a very loose sense. Vocabularies are more often described using RDF. An RDF vocabulary is a collection of IRIs intended for use in RDF graphs [32]. Henceforth, the term *semantic schema* will be used to refer to both vocabularies and ontologies or any combination of one or more of them.

The RDF framework allows to express statements about resources [41]. However, it is also required to define the terms which are intended to be used in those statements, to indicate that they describe specific kinds or classes of resources, and use specific properties in describing those entities [41]. To support the definition of vocabularies, RDF has the RDF Schema language (RDFS) [41] which provides a data-modelling vocabulary for RDF data that describes groups of related resources and relationships between them. W3C published a note regarding best practice for publishing RDF vocabularies [9]. Resources may be divided into groups called classes [11]. The members of a class are known as instances of the class. An RDF property is a relation between subject and object resources.

The IRIs in an RDF graph often begin with a common substring, known as the *namespace* IRI. Some namespace IRIs are associated by convention with a short name, the *namespace prefix*. In some RDF serializations, such as Turtle [5], IRIs are abbreviated, using the namespace prefix to reduce the size and assist readability. `prefix.cc`[13] is a namespace lookup service for human (via its user interface) and software (via its Web API) agents. In the same context, the Linked Open Vocabularies[14] provides an index of vocabularies.

Certain ontologies or vocabularies are recommended by W3C. By the time of writting, these are the organization ontology [38], and the data catalog (DCAT) [35], RDF data cube [17], and Web annotation vocabularies [39]. Besides the W3C recommended vocabularies, there are a few more which were published by W3C as notes. These are the vCard ontology [31], the Asset Description Metadata Schema (ADMS) [4], and the Registered Organization Vocabulary. Note that all of the aforementioned, besides the RDF data cube vocabulary, are used to semantically annotate the data in our use cases (Chapter 6). Besides the W3C recommended vocabularies, the Dublin Core Metadata Initiative (DCMI) Metadata Terms vocabulary[15] and Friend-Of-A-Friend vocabulary[16] (FOAF) are the two most popular vocabularies, as state by both the Linked Open Vocabularies (LOV) and LODstats [17].

[12] `https://www.w3.org/standards/semanticweb/ontology`
[13] prefix.cc, `http://prefix.cc`
[14] LOV, `http://lov.okfn.org`
[15] DCMI, `http://dublincore.org/documents/dcmi-terms/`
[16] FOAF, `http://xmlns.com/foaf/spec/`
[17] LODstats, `http://stats.lod2.eu/`

Data that is semantically annotated can be easily interpreted by software agents. But *how exactly are these semantic annotations applied?*

1.5 Linked Data lifecycle

To render data available as Linked Data, certain steps are required to be followed, namely (i) modeling, (ii) generation, (iii) validation, and (iv) publication. These steps do not need to be executed as consecutive steps. To the contrary, different approaches were proposed. Altogether they form a *Linked Data publishing workflow*.

An often quoted concern about the Semantic Web is the cost of ontology development and maintenance [42]. We go beyond this and consider that the concern does not lie only on the cost of ontology development and maintenance, but on the cost of applying them to data and, thus, generating Linked Data. Therefore, this PhD thesis aims to deal with this issue, and, more precisely, to facilitate Linked Data generation lowering its development and maintenance cost and enabling this way Semantic Web's adoption.

This section is organized as follows: in Section 1.5.1, we describe how mapping rules can be defined to specify the generation of Linked Data from raw data. In Section 1.5.2, we explain how such mapping rules can be executed to actually generate a Linked Data set. In Section 1.5.3, we introduce the concept of validation in the Linked Data generation workflow, and in Section 1.5.4 the concept of metadata and provenance. Last, in Section 1.5.5, the publication of Linked Data is described.

1.5.1 Modeling

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

- which **entities** appear in the dataset, by assigning *IRIs* to identify them;
- which **attributes** appear in the original data to describe the entities;
- what the **(data)types** are of these entities, by using *classes*, and of these attributes, by using `xsd`[18] or custom *datatypes*; and
- which **relationships** exist among entities, or between entities and attributes, 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. *Mapping rules* define correspondences between data in different schemas [21]. A *mapping* is a collection of such rules, all oriented in the same direction –

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

an oriented alignment [21]. However, directly editing them using these languages is difficult for human agents who are not Semantic Web experts. Therefore, graphical user interface tools, such as the RMLEditor [26] (see Section 2.6), are developed to ease modeling and support in defining mapping rules to semantically annotate data and generate Linked Data.

1.5.2 Generation

The next step in a typical Linked Data life cycle is the *generation* step. Mapping languages enable detaching mapping rules from the implementation that executes them and specifying in a declarative way how Linked Data is generated from raw data. However, dedicated tools *execute* these to generate the desired Linked Data as the mapping rules indicate.

Linked Data may be generated from different media, such as text, audio, or video. In this PhD thesis, we focus on textual data and more precisely on semi-structured data. In Chapter 2, we describe our mapping language and, in Section 3.4, we describe our tool for executing mapping rules to apply annotations to data and generate Linked Data. Most of the time, *Linked Data* originally stems from (semi-)structured formats (CSV, XML, etc.) and their RDF representation is obtained by repetitively applying mapping rules.

1.5.3 Validation

Linked Data validation aims at aiding the data owners to acquire high quality Linked Data. An RDF graph is inconsistent if it contains an internal contradiction [32]. The most frequent violations are related to the dataset's schema, namely the vocabularies used to annotate the original data [33]. 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 the same violations being repeatedly observed even within the same Linked Data set.

Only recently efforts focus on formalizing Linked Data quality tracking and assessment [45]. Nevertheless, most approaches remain independent of the Linked Data generation and publication workflow. Moreover, although more and more data is published as Linked Data, the Linked Data sets' quality varies significantly, ranging from expensively curated to relatively low quality Linked Data sets [45]. Nevertheless, high quality Linked Data is an important factor for the success of the envisaged Semantic Web, as machines are inherently intolerant to interpret unexpected input.

1.5.4 Provenance

The rapid growth of the Linked Open Data cloud comes with the advantage that the *availability* of Linked Data which could enable Semantic Web adoption, increases. However, it also comes with its disadvantages: a decline in Linked Data *quality* and often doubt about their *provenance*, thus, their trustworthiness 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. As such, automatically assert provenance and metadata information related to Linked Data generation is required, and even was encouraged by the Linked Data principles.

1.5.5 Publication

The next step in a typical Linked Data publishing workflow is *publication*. In this section, we discuss how Linked Data can be published. The Linked Data publication main goal is to render Linked Data able to be *retrieved* and *discovered*. Not only the data, but also the interface(s) through which it is published, should be machine-processable. *Linked Data Fragments* (LDF) [44] was introduced as a framework for comparing Linked Data publication interfaces with respect to the required server and client effort for querying them.

Data Dump A data dump is a file which contains a Linked Data set and is available in any RDF serialization. Data dumps may be available in a compressed archive such as gzip or HDT [22]. Agents need to download the data dump and load it locally if desired to query it. Hence, although little effort is required to publish the Linked Data in data dumps, significant effort is required from the client to query them.

Linked Data Document When a URI is *dereferenced*, the provided information can be returned as a *Linked Data document*, containing information about the entity that this URI identifies. This means that data becomes available in smaller fragments, which can be access on a per-entity basis. 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 [44].

SPARQL Query Result SPARQL endpoints are the most popular method for publishing Linked Data. Even though they are widely used, they suffer from significant availability issues [44]. The potentially high complexity of queries causes a very high load on servers that host SPARQL endpoints, leading often to significant downtime. This makes SPARQL endpoints a costly approach for publishing Linked Data, but, the required effort to query these endpoints is low, because the server performs the entire query evaluation process.

Triple Pattern Fragments The Triple Pattern Fragments (TPF) [44] were introduced as a trade-off between server and client effort for querying. The approach consists of a low-cost server interface while clients need to evaluate more complex SPARQL queries. Clients split up the queries into one or more triple pattern queries, sending them to a TPF interface, and joining the results. TPF requires less effort from the server compared to SPARQL endpoints, at the cost of slower query execution times [44]. This approach makes it possible to publish Linked Data at a low cost, while enabling efficient querying.

Depending on the publication approach, different storage solutions might be chosen, or vice-versa. Table 1.1 shows storage solutions per Linked Data publication approach.

Linked Data interface	Storage solution
data dump	RDF file, HDT, ...
Linked Data documents	RDF file
TPF	RDF file, HDT, SPARQL engine, ...
SPARQL endpoint	triplestore

Table 1.1: *A mapping of Linked Data interfaces to appropriate storage solutions.*

1.6 Problem Statement

The real power of the Semantic Web will be realized once a significant number of software agents that collect machine-readable Web content –but not only– from diverse sources, become available [8]. The original article about the Semantic Web [8] included many scenarios with intelligent agents undertaking tasks on behalf of human (or other software) agents. However, these scenarios are hand-crafted for particular tasks. Software agents still have limited ability to interact with heterogeneous data and information types [42].

The idea of the Semantic Web, as it was coined in 2001, was not adopted as eagerly as its inventors hoped, because realizing the Semantic Web is not so straightforward as it was thought by its inventors [42]. While there are multiple reasons why the Semantic Web does not reach its full potential yet, a crucial and determining factor is:

> (Intelligent) software agents which function with Semantic Web technologies do not have enough data to work with, and human agents do not want to put in effort to provide Linked Data until there are software agents that use it.

Thus, the Semantic Web comes with challenges to be addressed [19]: On the one hand, data owners need to invest effort to define semantic annotations without an immediate reward. On the other hand, semantic heterogeneity issues that arise when different data owners create semantically enhanced representations of data need to be tackled. Thus, the Semantic Web's effectiveness depends on the availability of machine-readable Web content (Linked Data) and automated services [8]. This PhD aims to address this problem by facilitating the Linked Data generation from heterogeneous data by reducing the semantic heterogeneity and increasing the quality. To achieve it, we propose a set of complementary techniques and corresponding implementations, that enable Semantic Web's adoption. Each one addresses a part of the envisaged high quality semantic enhancement and integration in the form of Linked Data from semi-structured heterogeneous data.

To be even more precise on the problem, despite the significant number of existing tools, only a limited amount of data is available as Linked Data, because acquiring its semantically enriched representation still remains complicated. For instance, governments publish their data as Open Data and often turn them into Linked Open Data afterwards. In practice,

they need to generate Linked Data from data which is originally in various formats. Therefore, they need to install, learn, use and maintain different tools for each format separately, which hampers their effort to ensure the integrity of their Linked Dataset even more.

Moreover, even though the Semantic Web technologies became more mature, no alternative methodologies were investigated to intensify Linked Data generation. To the contrary, deploying the five stars of the Linked Open Data schema[19] or barely applying best practices [30] is still the de-facto way of integrating data. Approaching the stars or best practices as a set of consecutive steps and separately applying them to distinct individual data sources, disregarding possible prior definitions and links to other entities, prohibits reaching the uppermost goal of actually generating interlinked data.

Following the aforementioned example, much of this Open Data complements each other in the description of different knowledge domains. For instance, two data sets with information regarding performances and exhibitions may also contain information about the venue where they take place. The venues are then described in each data set. Generating Linked Data by applying the stars and best practices would lead into assigning distinct identifiers (IRIs) for the same venues, as each data source is handled discreetly. Only once Linked Data is generated from each distinct data source, the identifiers are aligned (*interlinking*), leading to more triples, longer path traversals and more complicated queries.

Manually aligning entities that appear in certain data to their prior appearances is often performed by redefining their semantic representations, while links to other entities are defined after Linked Data is generated. Identifying, interlinking or replicating, and keeping them aligned is complicated and the situation aggravates the more data sources are incorporated. Existing approaches tend to generate multiple identifiers for same entities while duplicates can be found even within a data owner's Linked Data set.

Following again the aforementioned example, replying to a question that looks for all possible events that taking place in a certain region would end up with an answer that contains duplicate identifiers (IRIs) for the same venues, as the different data sources are discreetly incorporated in the Linked Data set and thus the identifiers are not aligned in the first place.

Last, although more and more data is published as Linked Data, the Linked Data sets' quality varies significantly, ranging from expensively curated to relatively low quality Linked Data sets [45]. Its quality depends on the incorrect usage of semantic annotations, for instance, assigning the property "gender" to a resource whose type is "plant", and their incorrect application to the original data sources, for instance, defining that a certain element of a data source should be annotated with the property "gender" whereas its values contain information regarding its color, as well as on errors in the original data source, for instance, in the color field, there is a value that is not color, e.g., "circle".

The aforementioned partially occurs because so far there was neither a uniform formalisation to define how to generate Linked Data from heterogeneous distributed data sources in an integrated and interoperable fashion, nor a complete solution that supports the whole Linked Data generation procedure. Hence, a well-considered policy regarding generating Linked Data in the context of a certain knowledge domain, to either integrate semantically enriched data in a knowledge graph or answer a query on-the-fly is required.

[19] five stars Linked Open Data, `http://5stardata.info/`

The fundamental problems are (i) the multiple different ways that data is expressible and interpretable, and (ii) the lack of maintenance of the correspondence between data and its annotations. There are no means to (i) uniformly declare how Linked Data is generated from heterogeneous data, (ii) uniformly execute the aforementioned declarations to actually generate this Linked Data, (iii) validate the consistency, quality, and integrity of the generated output, apart from manual user-driven controls, and (iv) automate these tests and incorporate them in the Linked Data generation procedure.

1.6.1 Hypotheses

Considering the aforementioned problem, the main hypotheses related to my research are:

Hypothesis #1 *Integrated Linked Data generation from heterogeneous data leads to more robust and complete sets with increased integrity and connectedness.*

Hypothesis #2 *Decoupling between representation and execution of rules for Linked Data generation, as well as data extraction and rules execution, increases the expressivity and modularity, while enables interoperability and primary interlinking.*

Hypothesis #3 *Assessing a Linked Data set's validity and consistency, and identifying the violations' root cause allows to refine inconsistencies which are otherwise propagated leading to higher quality Linked Data.*

1.6.2 Research Questions

These hypotheses lead to the following main question of my doctoral research:

How can we access and represent domain level information with high quality from distributed heterogeneous sources in an integratable and uniform way?

Decoupling the declaration of the rules that define how Linked Data is generated from their execution (Hypothesis #2) requires tackling each aspect separately and leads to distinct research questions. On the one hand, the representation aspect needs to be investigated:

Research Question #1 *How can we define how to combine and enrich data from different heterogeneous data sources and enrich this knowledge with new relationships?*

Replying to this question shows that mapping rules representation can be declared independently of their execution, validating Hypothesis #2. Nevertheless, uniformly addressing the heterogeneity is the challenge which needs to be tackled. Its answer lies in Chapter 2 where a language for mapping rules declaration is presented, RML, enabling integrated Linked Data generation from heterogeneous data, validating Hypothesis #1, and an exemplary interface, the RMLEditor, for facilitating its editing.

Research Question #2 *How can we assess, improve, and verify high quality semantically enhanced representations from heterogeneous data sources?*

Replying to this question enables assessing Linked Data validity and consistency, validating Hypothesis #3. Nevertheless, the challenge is not only to perform quality assessment but to do so as early as possible in a timely manner and in a way that allows resolving the violations at its root. Its answer lies in Chapter 4 where a uniform methodology for assessing the quality of both mapping rules and Linked Data is presented, as well as an exemplary implementation, the RMLValidator.

On the other hand, the accessing and executing aspect needs to be investigated too:

Research Question #3 *How can we enable accessing and retrieving data from distributed heterogeneous data sources on the Web (or not) in a uniform way?*

Research Question #4 *How can we generate, based on the aforementioned definitions, semantically enhanced representations from distributed heterogeneous data sources?*

Replying to these two questions shows that there can be algorithms and corresponding implementations that can execute the mapping rules declaration and generate Linked Data, and enables decoupling data extraction and rules execution, validating Hypothesis #2. The challenge here is to uniformly deal with not only hetergeneous data but also heterogeneous interfaces and to investigate alternative algorithms for the mapping rules execution. Its answer lies in Chapter 3 where factors that determine how the execution should occur are presented, as well as an exemplary implementation, the RMLMapper.

Last, in Chapter 5, the different components of the workflow which contribute in generating high quality Linked Data are brought together and a corresponding implementation is presented, the RMLWorkbench. Figure 1.4 gives an overview of the different aspects of the problem and the corresponding contributions that are presented in each chapter.

1.6.3 Outline

The PhD thesis consists of the following chapters: (i) *declaration* (Chapter 2), where a language is proposed to specify rules that define semantic annotations which are applied to data and its relations; (ii) *execution* (Chapter 3), where insights with respect to the execution of such mapping rules to generate Linked Data is described; (iii) *validation* (Chapter 4), where a methodology to validate the mapping rules and generated Linked Data is proposed to verify the output's high quality and (iv) *workflow composition* (Chapter 5), where workflows for the definitions of mapping rules, its execution and validation are described.

Each chapter consists of a short introduction, an outline of the state of the art, the theoretical foundations and a corresponding implementation. Besides those chapters, in Chapter 6 we present different use cases where the results of this PhD thesis were applied and in Chapter 7 we summarize the conclusions and future work of this PhD thesis.

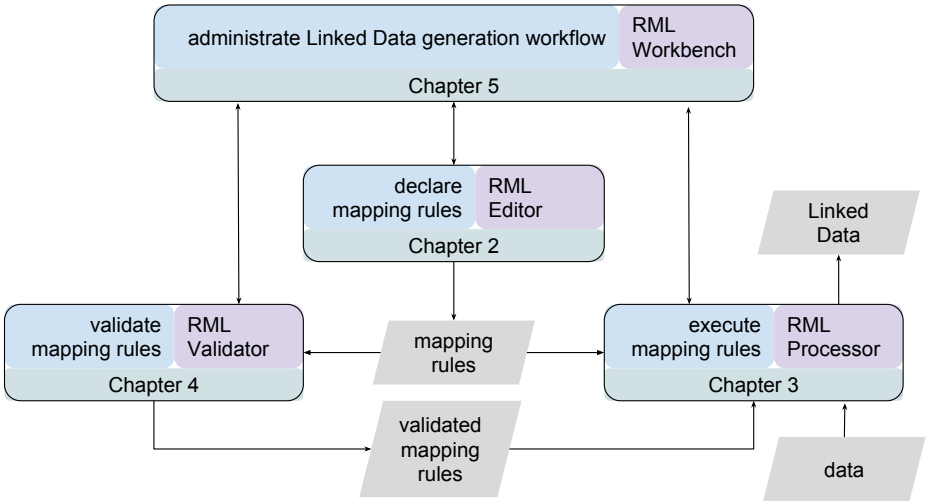


Figure 1.4: *Overview of this PhD thesis*

1.6.4 Outcomes

This PhD thesis led to 3 journal articles, (one accepted [3], one currently under major revision [27] and one under review [2]). Besides the aforementioned journal articles, during this PhD, I co-authored another 2 journal publications [15, 43]. This PhD also led to 4 publications [1, 18, 26, 36] accepted in the main track of the two top conferences of our field, namely European and International Semantic Web Conferences (ESWC and ISWC respectively). Besides the aforementioned, during this PhD thesis I co-authored another 43 publications, among which 10 are indexed by the Web of Science. Last, the results of my work during this PhD thesis received more than 300 citations by the time of writing.

This PhD thesis' results were accomplished and applied in the frame of the projects: EWI Open Data, COMBUST, SOaP, MOS2S, DiSSECT, CITADEL, DiVeRsiFy, and EcoDaLo.

References

[1] **Anastasia Dimou**, Dimitris Kontokostas, Markus Freudenberg, Ruben Verborgh, Jens Lehmann, Erik Mannens, Sebastian Hellmann, and Rik Van de Walle. Assessing and Refining Mappings to RDF to Improve Dataset Quality. In Marcelo Arenas, Oscar Corcho, Elena Simperl, Markus Strohmaier, Mathieu d'Aquin, Kavitha Srinivas, Paul Groth, Michel Dumontier, Jeff Heflin, Krishnaprasad Thirunarayan, and Steffen Staab, editors, *The Semantic Web – ISWC 2015*, volume 9367 of *Lecture Notes in Computer Science*, pages 133–149. Springer International Publishing, Oct 2015. doi: 10.1007/978-3-319-25010-6_8. URL `http://link.springer.com/chapter/10.1007/978-3-319-25010-6_8`.

[2] **Anastasia Dimou**, Ben De Meester, Pieter Heyvaert, Ruben Verborgh, Steven Latré, and Erik Mannens. RML Mapper: a tool for uniform Linked Data generation from heterogeneous data. *Semantic Web Journal*, 2017. (under review).

[3] **Anastasia Dimou**, Sahar Vahdati, Angelo Di Iorio, Christoph Lange, Ruben Verborgh, and Erik Mannens. Challenges as Enablers for High Quality Linked Data: Insights from the Semantic Publishing Challenge. *PeerJ Computer Science*, 3: e105, January 2017. ISSN 2376-5992. doi: 10.7717/peerj-cs.105. URL https://peerj.com/articles/cs-105/.

[4] Phil Archer and Gofran Shukair. Asset Description Metadata Schema (ADMS). W3C Working Group Note, W3C, August 2013. http://www.w3.org/TR/vocab-adms/.

[5] David Beckett, Tim Berners-Lee, Eric Prud'hommeaux, and Gavin Carothers. RDF 1.1 Turtle. W3C Recommendation, W3C, February 2014. URL http://www.w3.org/TR/turtle/.

[6] T. Berners-Lee, L. Masinter, and M. McCahill. Uniform Resource Locators (URL). Standard Track, IETF, December 1994. https://tools.ietf.org/html/rfc3986.

[7] T. Berners-Lee, R. Fielding, and L. Masinter. Uniform Resource Identifiers (URI): Generic Syntax. Standard Track, IETF, August 1998. https://tools.ietf.org/html/rfc3986.

[8] Tim Berners-Lee, James Hendler, and Ora Lassila. The Semantic Web. *Scientific American*, 284(5):34–43, May 2001. URL http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21.

[9] Diego Berrueta and Jon Phipps. Best Practice Recipes for Publishing RDF Vocabularies. W3C Working Group Note, W3C, August 2008. http://www.w3.org/TR/swbp-vocab-pub/.

[10] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked Data - the story so far. *International Journal on Semantic Web and Information Systems*, 5(3):1–22, 2009. URL https://www.igi-global.com/chapter/linked-data-story-far/55046.

[11] Dan Brickley and R. V. Guha. RDF Schema 1.1. W3c recommendation, W3C, February 2014. URL https://www.w3.org/TR/rdf-schema/.

[12] Gavin Carothers and Andy Seaborne. RDF 1.1 TriG. W3C Recommendation, W3C, February 2014. URL http://www.w3.org/TR/trig/.

[13] Gavy Carothers. RDF 1.1 N-Quads. W3C Recommendation, W3C, February 2014. http://www.w3.org/TR/n-quads/.

[14] Gavy Carothers and Andy Seaborne. RDF 1.1 N-Triples. W3C Recommendation, W3C, February 2014. http://www.w3.org/TR/n-triples/.

[15] Pieter Colpaert, Mathias Van Compernolle, Laurens De Vocht, Anastasia Dimou, Miel Vander Sande, Ruben Verborgh, Peter Mechant, and Erik Mannens. Quantifying the Interoperability of Open Government Datasets. *IEEE Computer*, 47(10):50–56, 2014.

[16] Dan Connolly. JSON-LD 1.0. W3C Recommendation, W3C, January 2014. http://www.w3.org/TR/json-ld/.

[17] Richard Cyganiak and Dave Reynolds. The RDF Data Cube Vocabulary. W3C Recommendation, W3C, January 2014. http://www.w3.org/TR/vocab-data-cube/.

[18] Ben De Meester, Wouter Maroy, **Anastasia Dimou**, Ruben Verborgh, and Erik Mannens. Declarative Data Transformations for Linked Data Generation: The Case of DBpedia. In Eva Blomqvist, Diana Maynard, Aldo Gangemi, Rinke Hoekstra, Pascal Hitzler, and Olaf Hartig, editors, *The Semantic Web: 14th International Conference, ESWC 2017, Portorož, Slovenia, May 28 – June 1, 2017, Proceedings, Part II*, pages 33–48, Cham, 2017. Springer International Publishing. ISBN 978-3-319-58451-5. doi: 10.1007/978-3-319-58451-5_3. URL http://dx.doi.org/10.1007/978-3-319-58451-5_3.

[19] AnHai Doan, Alon Halevy, and Zachary Ives. *Principles of Data Integration*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edition, 2012. ISBN 0124160441, 9780124160446.

[20] M. Duerst and M. Suignard. Internationalized Resource Identifiers (IRIs). Standard Track, IETF, January 2005. https://tools.ietf.org/html/rfc3987.

[21] Jérôme Euzenat and Pavel Shvaiko. *Ontology Matching*. Springer-Verlag, Heidelberg (DE), 2nd edition, 2013.

[22] Javier D. Fernández, Miguel A. Martínez-Prieto, Claudio Gutiérrez, Axel Polleres, and Mario Arias. Binary RDF representation for publication and exchange (HDT). *Web Semantics: Science, Services and Agents on the World Wide Web*, 19: 22 – 41, 2013. ISSN 1570-8268. doi: http://dx.doi.org/10.1016/j.websem.2013.01.002. URL http://www.sciencedirect.com/science/article/pii/S1570826813000036.

[23] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, and T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1. Standard Track, IETF, January 1997. https://tools.ietf.org/html/rfc2616.

[24] Fabien Gandon and Guus Schreiber. RDF 1.1 XML Syntax. W3C Recommendation, W3C, February 2014. URL http://www.w3.org/TR/rdf-syntax-grammar/.

[25] James Hendler. Is there an intelligent agent in your future? *Nature*, 1999. URL http://www.nature.com/nature/webmatters/agents/agents.html.

[26] 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 Harald Sack, Eva Blomqvist, Mathieu d'Aquin, Chiara Ghidini, Paolo Simone Ponzetto, and Christoph Lange, editors, *The Semantic Web – Latest Advances and New Domains (ESWC 2016)*, volume 9678 of *Lecture Notes in Computer Science*, pages 709–723. Springer, 2016. ISBN 978-3-319-34129-3. doi: 10.1007/978-3-319-34129-3_43. URL http://dx.doi.org/10.1007/978-3-319-34129-3_43.

[27] Pieter Heyvaert, **Anastasia Dimou**, Ben De Meester, Tom Seymoens, Aron-Levi Herregodts, Ruben Verborgh, Dimitri Schuurman, and Erik Mannens. Specification and Implementation of Mapping Rule Visualization and Editing: MapVOWL and the RMLEditor. *Journal of Web Semantics*, 2017. first two co-first authors, under revision.

[28] Pascal Hitzler, Markus Krötzsch, Bijan Parsia, Peter Patel-Schneider, and Sebastian Rudolph. OWL 2 Web Ontology Language. W3C Recommendation, W3C, December 2012. http://www.w3.org/TR/owl-primer.

[29] Bernadette Hyland, Ghislain Atemezing, Michael Pendleton, and Biplav Srivastava. Linked Data Glossary. W3C Working Group Note, W3C, June 2013. `http://www.w3.org/TR/ld-bp/`.

[30] Bernadette Hyland, Ghislain Atemezing, and Boris Villazón-Terrazas. Best Practices for Publishing Linked Data. W3C Working Group Note, W3C, January 2014. `http://www.w3.org/TR/ld-bp/`.

[31] Renato Iannella and James McKinney. vCard Ontology - for describing People and Organizations. W3C Interest Group Note, W3C, May 2014. `http://www.w3.org/TR/vcard-rdf/`.

[32] Graham Klyne, Jeremy J. Carroll, and Brian McBride. RDF 1.1 Concepts and Abstract Syntax. W3C Recommendation, W3C, February 2014. `https://www.w3.org/TR/rdf11-concepts/`.

[33] Dimitris Kontokostas, Patrick Westphal, Sören Auer, Sebastian Hellmann, Jens Lehmann, Roland Cornelissen, and Amrapali Zaveri. Test-driven evaluation of linked data quality. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14, pages 747–758, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2744-2. doi: 10.1145/2566486.2568002. URL `http://doi.acm.org/10.1145/2566486.2568002`.

[34] Maurizio Lenzerini. Data Integration: A Theoretical Perspective. In *Proceedings of the Twenty-first ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '02, pages 233–246, New York, NY, USA, 2002. ACM. ISBN 1-58113-507-6. doi: 10.1145/543613.543644. URL `http://doi.acm.org/10.1145/543613.543644`.

[35] Fadi Maali and John Erickson. Data Catalog Vocabulary (DCAT). W3C Recommendation, W3C, January 2014. `http://www.w3.org/TR/vocab-dcat/`.

[36] Wouter Maroy, **Anastasia Dimou**, Dimitris Kontokostas, Ben De Meester, Ruben Verborgh, Jens Lehmann, Erik Mannens, and Sebastian Hellmann. Sustainable Linked Data Generation: The Case of DBpedia. In Claudia d'Amato, Miriam Fernandez, Valentina Tamma, Freddy Lecue, Philippe Cudré-Mauroux, Juan Sequeda, Christoph Lange, and Jeff Heflin, editors, *The Semantic Web – ISWC 2017: 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part II*, pages 297–313, Cham, 2017. Springer International Publishing. ISBN 978-3-319-68204-4. doi: 10.1007/978-3-319-68204-4_28. URL `https://doi.org/10.1007/978-3-319-68204-4_28`.

[37] R. Moats. URN Syntax. Standard Track, IETF, May 1997. https://tools.ietf.org/html/rfc3986.

[38] Davy Reynolds. The Organization Ontology. W3C Recommendation, W3C, January 2014. https://www.w3.org/TR/vocab-org/.

[39] Robert Sanderson, Paolo Ciccarese, and Benjamin Young. Web Annotation Vocabulary. W3C Recommendation, W3C, February 2017. https://www.w3.org/TR/annotation-vocab/.

[40] Leo Sauermann and Richard Cyganiak. Cool URIs for the Semantic Web. W3C Interest Group Note, W3C, December 2008. https://www.w3.org/TR/cooluris/.

[41] Guus Schreiber and Yves Raimond. RDF 1.1 Primer. W3C Working Group Note, W3C, June 2014. https://www.w3.org/TR/rdf11-primer/.

[42] Nigel Shadbolt, Tim Berners-Lee, and Wendy Hall. The Semantic Web Revisited. *IEEE Intelligent Systems*, 21(3):96–101, May 2006. ISSN 1541-1672. doi: 10.1109/MIS.2006.62. URL http://dx.doi.org/10.1109/MIS.2006.62.

[43] Miel Vander Sande, Ruben Verborgh, **Anastasia Dimou**, Pieter Colpaert, and Erik Mannens. Hypermedia-based discovery for source selection using low-cost linked data interfaces. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 12(3):79–110, 2016.

[44] Ruben Verborgh, Miel Vander Sande, Olaf Hartig, Joachim Van Herwegen, Laurens De Vocht, Ben De Meester, Gerald Haesendonck, and Pieter Colpaert. Triple Pattern Fragments: a Low-cost Knowledge Graph Interface for the Web. *Journal of Web Semantics*, 37–38:184–206, March 2016. ISSN 1570-8268. doi: doi:10.1016/j.websem.2016.03.003. URL http://linkeddatafragments.org/publications/jws2016.pdf.

[45] Amrapali Zaveri, Anisa Rula, Andrea Maurino, Ricardo Pietrobon, Jens Lehmann, and Sören Auer. Quality Assessment for Linked Data: A Survey. *Semantic Web Journal*, 7(1):63–93, 2016. URL http://www.semantic-web-journal.net/content/quality-assessment-linked-data-survey.

Πενία τέχνας κατεργάζεται
Πλάτων

Necessity is the mother of invention
Plato

2 Declaration

A formalization for mapping rules definition

A lack of in-depth understanding of Linked Data generation complexity and the many degrees of freedom in modeling and representing knowledge prevents human and software agents from directly profiting of the Semantic Web potential. This chapter identifies the barriers that still keep Linked Data generation being a complicated task. Then we outline current solution's limitations and provide the requirements to easily generate Linked Data and integrate knowledge from multiple and heterogeneous data sources. We provide a solution that allows agents to declaratively define the mapping rules that specify how Linked Data is generated and a tool which facilitates the editing for human agents. This way, we expect to overcome the barriers for Linked Data generation and allow agents to build intelligent applications without being concerned on how to obtain Linked Data.

chapter's outline In this section, we introduce the needs that lead to declaratively defining rules for integrated Linked Data generation from heterogeneous data sources (Section 2.1), and we summarize the state-of-the-art with respect to mapping languages for defining rules that specify how Linked Data is generated (Section 2.2). Then, we outline the limitations of current solutions and requirements for more generic solutions (Section 2.3). In Section 2.4, we introduce the RDF Mapping Language (RML), which addresses the aforementioned limitations and fulfills the desired requirements, and its extensions (Section 2.5). Last, in Section 2.6, we introduce the RMLEditor, a user interface to facilitate human agents to specify mapping rules for their own data.

chapter's sources This chapter is based on the following papers:

1. **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. URL `http://ceur-ws.org/Vol-1184/ldow2014_paper_01.pdf`

2. **Anastasia Dimou**, Miel Vander Sande, Jason Slepicka, Pedro Szekely, Erik Mannens, Craig Knoblock, and Rik Van de Walle. Mapping Hierarchical Sources into RDF using the RML Mapping Language. In *Proceedings of the 2014 IEEE International Conference on Semantic Computing*, ICSC '14, pages 151–158, Washington, DC, USA, 2014. IEEE Computer Society. ISBN 978-1-4799-4003-5. doi: 10.1109/ICSC.2014.25. URL `http://dx.doi.org/10.1109/ICSC.2014.25`

3. **Anastasia Dimou**, Miel Vander Sande, Pieter Colpaert, Erik Mannens, and Rik Van De Walle. Extending R2RML to a Source-independent Mapping Language for RDF. In *Proceedings of the 2013th International Conference on Posters & Demonstrations Track - Volume 1035*, ISWC-PD'13, pages 237–240, Aachen, Germany, Germany, 2013. CEUR-WS.org. URL `http://dl.acm.org/citation.cfm?id=2874399.2874459`

4. 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 Harald Sack, Eva Blomqvist, Mathieu d'Aquin, Chiara Ghidini, Paolo Simone Ponzetto, and Christoph Lange, editors, *The Semantic Web – Latest Advances and New Domains (ESWC 2016)*, volume 9678 of *Lecture Notes in Computer Science*, pages 709–723. Springer, 2016. ISBN 978-3-319-34129-3. doi: 10.1007/978-3-319-34129-3_43. URL `http://dx.doi.org/10.1007/978-3-319-34129-3_43`

2.1 Introduction

Most data that agents would like to process as Linked Data exists in structures other than Linked Data. To this end, many languages, different tools, and manifold approaches were proposed to generate Linked Data. Even though relational databases account for a significant amount of data, a growing number of data is differently represented. For instance, government and scientific data is often published in spreadsheets or text delimited (e.g., CSV) files, while vast amounts of data is accessible via Web APIs that return data in XML or JSON. Indicatively, ProgrammableWeb.org[1], a Web APIs index, reports over 17,000 different APIs in 2017. Nevertheless, in real-world situations, multiple data sources with data in different formats are part of multiple domains, which, in their turn, are formed by integrating multiple sources and relations among them. When generating Linked Data from heterogeneous data, current approaches often fail to reach the goal of publishing *interlinked* data.

Each resource's semantic representation is defined independently, disregarding its possible prior definitions and links to other resources. Manual alignment to prior appearances is performed by redefining it, while links to other resources are defined *after* the data is generated and published. Nonetheless, as Linked Data sets are often shaped gradually, a demand emerges for well-considered policies with regard to generating and interlinking data in the context of certain knowledge domain. For instance, agents (humans or machines) publish data as Open Data and turn them into Linked Open Data *afterwards*. Much of this data complements each other in the description of different knowledge domain. Therefore, same entities appear in multiple Linked Data sets, and problematically, often with different identifiers or formats. There is no consideration on incorporating it in a well-considered and standardized Linked Data generation strategy; to the contrary, currently fragmented solutions prevail.

Furthermore, Linked Data is generated progressively, thus it is important that agents incorporate their data in what is already published. Reusing the same unique identifiers for the same entities is necessary to achieve this, but it is only possible if prior existing definitions in the same Linked Data set are discovered and if they can be replicated, instead of newly generated. Otherwise, duplicates will inevitably appear – even within a data publisher's own Linked Data set. Identifying, replicating, and keeping those definitions aligned is complicated and the situation aggravates the more data is generated and published.

Dealing with this problem requires a uniform, modular, interoperable, and extensible technology that supports gradually incrementing Linked Data generation. Such a solution can deal with Linked Data generation and primary interlinking, which should take place in a tightly coordinated way instead of two separate, consecutive actions. Deploying the five stars of the Linked Open Data schema[19] or following the Linked Data Best practices [22] is the *de-facto* way of generating Linked Data so far. Nevertheless, approaching the stars as consecutive steps and applying them to a single source every time –as most solutions tend to do– is not always optimal. To leverage heterogeneous data in Linked Data applications, not only solutions to express how Linked Data is generated are required, but also solutions to generate their *cross-data set* links to ensure Linked Data sets with higher integrity.

[1] Programmable Web, `https://www.programmableweb.com/`

To this end, the RDF mapping language (RML) was introduced [4], as an extension of R2RML[2] [14], the W3C recommendation for generating Linked Data from data in relational databases. RML supports Linked Data generation beyond databases, but also from data in different structures, formats, and serializations.

2.2 State of the Art

Several solutions exist to generate Linked Data from data in different structures and serializations. Most solutions though are *format-specific* or *case-specific*. Therefore they do not provide a generic solution for generating Linked Data from heterogeneous data.

2.2.1 Mapping Languages

Generating Linked Data from heterogeneous sources and interlinking them to describe a certain domain could be considered as a case of data integration. Linked Data from each different data source is defined and generated independently of the other sources, while different technics emerged to generate Linked Data from different formats. Therefore, multiple mapping languages were proposed so far. Nevertheless, most of them are format specific. Below we mention the most well-known mapping languages for each one of the most frequently encountered data structures and formats:

Databases For relational databases, different mapping languages were proposed. D2RQ Mapping Language[3] [7, 13] is a declarative language for generating Linked Data from relational database schemas. A D2RQ mapping has four logical elements: (i) for each class or group of similar classes, a record set from the database is specified; (ii) the record set is grouped according to the `groupBy` columns of the specific `ClassMap`; (iii) the class instances are created and assigned a URI or a blank node identifier; and (iv) the instance properties are created using datatype and object property bridges. D2RQ is the predecessor of R2RML, but differs on two aspects: D2RQ explicitly defines the database connectivity and uses different ways of expressing the joins.

R2RML [14] is the W3C-recommended language for expressing customized mapping rules from data in relational databases to generate Linked Data, using an agent's choice for a target semantic schema –which might be a combination of vocabularies and ontologies or a single one. R2RML is described in more details in Section 2.2.2. R2RML has a companion that defines direct mapping rules from relational databases to RDF [5]. In the direct mapping of a database, the structure of the resulting RDF graph directly reflects the structure of the database, the target RDF vocabulary directly reflects the names of database schema elements, and neither structure nor target vocabulary can be changed.

[2] R2RML, `http://www.w3.org/TR/r2rml`
[3] D2RQ, `http://d2rq.org/d2rq-language`

After R2RML became a W3C recommendation, Sparqlification Mapping Language (SML) [31] was the only language which was proposed to express how Linked Data is generated from relational databases, arguing that R2RML is very verbose and, thus, not as intuitive as it should be. SML has the same expressivity as R2RML [31], but uses a different syntactic approach, namely SML blends the traditional SQL `CREATE VIEW` statements with SPARQL `CONSTRUCT` queries.

Besides the aforementioned mapping languages, which are a few of the most prevalent ones, there were more mapping languages defined. Firstly, Hert et al. [18] outlined a few of them and then Spanos et al. [30] summarized in more details the most notable approaches on how to bring databases in the Semantic Web.

CSV Mapping languages were defined to support Linked Data generation from data in CSV and spreadsheets. They include the XLWrap's mapping language [25] that converts data in various spreadsheets to RDF, the declarative OWL-centric mapping language Mapping Master's M2 [27] that converts data from spreadsheets into the Web Ontology Language (OWL), and Tarql[4] [12] that follows a querying approach. The main drawback in the case of most CSV/spreadsheet-to-RDF mapping solutions is the assumption that each row describes an entity (*entity-per-row assumption*) and each column represents a property.

The CSV on the Web Working Group[5] defined a case-specific metadata vocabulary for tabular data on the Web [32]. CSVW outlines a data model and metadata for annotating tabular data that can be used as a basis for validation, display, or creating other formats. It also contains some non-normative guidance for publishing tabular data as CSV and how that maps into the tabular data model. Moreover, CSVW provides an annotated model of tabular data that can be supplemented by separate metadata about the table and defines how implementations should locate that metadata, given a file containing tabular data, and how to generate Linked Data, following a direct mapping approach (as for databases).

XML A larger variety of solutions exist to generate Linked Data from data in XML, but to the best of our knowledge, only a few specific languages were defined for this. To be more precise, GRDDL [10], that essentially provides the links to the algorithms (typically represented in XSLT) that define how to generate Linked Data, and the X3ML [26] language was defined for expressing custom mapping rules of data in XML to generate Linked Data.

While repurposing existing formalizations renders them suitable to be reused for generating Linked Data from data in the XML format, these formalizations remain focused on the format they are defined for, namely XML. For instance, Krextor [24] and AstroGrid-D[6] rely on XSLT [23], while Tripliser[7] on XPath [9], and XSPARQL [6] on XQuery [8]. XSPARQL, in particular, approximates a *Global-As-View* approach that performs dynamic query translation to convert different sources to Linked Data. XSPARQL generates Linked Data by integrating data from XML files and relational databases, once it has mapped them to RDF,

[4] Tarql, `https://github.com/cygri/tarql`
[5] CSVW, `http://www.w3.org/2013/csvw/wiki/Main_Page`
[6] AstroGrid-D, `http://www.gac-grid.de/project-products/Software/XML2RDF.html`
[7] Tripliser,`http://daverog.github.io/tripliser/`

and interlinks them with other RDF data, as Tarql's[8] function follows also a querying approach to generate Linked Data from data in the CSV format.

This renders it difficult, if not impossible, to be smoothly applied to other formats, as it would require an approach that uniformly deals with different data formats. These solutions for data in XML format lead to mapping rules on the syntactic level, rather than on the semantic level, or fail to provide a solution applicable to a broader domain than the one they are applied to, as it occurs, for instance, with Krextor.

2.2.2 R2RML

R2RML [14] is the W3C-recommended language to express customized mapping rules for data in relational databases to Linked Data sets represented using the RDF framework.

In R2RML, the mapping to the RDF data model is based on one or more Triples Maps (Listing 2.2, Lines 7 and 12) and occur over a Logical Table (Lines 7 and 12) iterating on a *per-row* basis. A Triples Map consists of three main parts: the Logical Table (`rr:LogicalTable`, Lines 7 and 12), the Subject Map and zero or more Predicate-Object Maps. The Subject Map (`rr:SubjectMap`, Line 39) defines the rule that generates unique identifiers (IRIs) for the RDF terms which are going to be generated and is used as the subject of all RDF triples that are generated from this Triples Map.

A Predicate-Object Map (Lines 9 and 10) consists of Predicate Maps (Lines 33 and 36), which define the rule that generates the triple's predicate and Object Maps (Line 19) or Referencing Object Maps (Line 22), which defines the rule that generates the triple's object. The Subject Map, Predicate Map, and Object Map are Term Maps, namely rules that generate an RDF term (an IRI, a blank node, or a literal). A Term Map can be a *constant-valued term map* (`rr:constant`, Line 18) that always generates the same RDF term, or a *column-valued term map* (`rr:column`, Line 34) that is a referenced column in a given Logical Table's row, or a *template-valued term map* (`rr:template`, Lines 30 and 39) that is a valid string template that can contain referenced columns.

R2RML supports cross-references between Triples Maps, when the Triples Map's subject is same as the object generated by a Predicate-Object Map. A Referencing Object Map (`rr:RefObjectMap`, Line 22) is then used to point to the Triples Map that generates on its Subject Map the corresponding resource, the so-called Referencing Object Map's Parent Triples Map. If the Triples Maps refers to different Logical Tables, a join between the Logical Tables is required. The join condition (`rr:joinCondition`, Line 23) performs the join exactly as a join is executed in SQL. The join condition consists of a reference to a column name that exists in the Logical Table of the Triples Map that contains the Referencing Object Map (`rr:child`, Line 24) and a reference to a column name that exists in the Logical Table of the Referencing Object Map's Parent Triples Map (`rr:parent`, Line 25).

All elements used in R2RML to define mapping rules are summarized in Figure 2.1.

[8] Tarql, `https://github.com/cygri/tarql`

```
title                                        short          city
13th Extended Semantic Web Conference        ESWC2016       Heraklion
15th International Semantic Web Conference   ISWC2016       Kobe
12th Extended Semantic Web Conference        ESWC2015       Portoroz
14th International Semantic Web Conference   ISWC2015       Bethlehem
...
```

Listing 2.1: *Data from a relational database*

```
@prefix rr: <http://www.w3.org/ns/r2rml#>.
@prefix ex: <http://example.com/ns#>.


=== Triples Maps ===

<#Conference_TM>  rr:logicalTable [ rr:tableName "conference" ];
                  rr:subjectMap         <#Conference_SM> ;
                  rr:predicateObjectMap <#City_POM> ;
                  rr:predicateObjectMap <#City_POM>.

<#Country_TM>     rr:logicalTable [ rr:tableName "country" ];
                  rr:subjectMap         <#Country_SM> .


=== Predicate Object Maps ===

<#City_POM>       rr:predicateMap       <#City_PM> ;
                  rr:objectMap          <#City_OM> .

<#Country_POM>    rr:predicateMap       <#Country_PM> ;
                  rr:objectMap          <#Country_ROM> ;
                  rr:join  [
                            rr:cild "country" ;
                            rr:parent "country_name"] .


=== Term Maps ===

<#Conference_SM> rr:template "http://ex.com/conference/{short}" ;
                 rr:class ex:Conference .

<#City_PM>       rr:constant ex:city .
<#City_OM>       rr:column "city" .

<#Country_PM>    rr:constant ex:country .
<#Country_ROM>   rr:parentTriplesMap <#Country_TM> ;

<#Country_SM>    rr:template "http://ex.com/country/{country_name}" ;
                 rr:class ex:Country .
```

Listing 2.2: R2RML *mapping rules for Linked Data generation from relational databases*

2.2.3 Editors

Mapping rules definition might occur manually, supported by an editor, or (semi-)automatically. The mapping rule creation and editing to specify how to generate Linked Data resulted in the development of two types of graphical user interface (GUI) tools: (i) *step-by-step wizards* and (ii) *visualization tools*. Step-by-step wizards prevailed, such as the fluidOps editor [28]. Nevertheless, they restrict human agents' editing options, hamper altering parameters in previous steps, and detach mapping rules from the overall knowledge modeling, as information is separated in different steps. These tools circumvent the underlying mapping languages' syntax, but agents are still required to have knowledge of the language's terminology.

A number of editors rely on graph visualizations to edit the mapping rules, such as Map-On [29] and the RMLEditor [21]. We distinguish three prevalent approaches for mapping rule visualizations: (i) visualizing mapping rules as Linked Data, e.g., the RMLx Visual Editor[9] which visualizes mapping rules using graphs as if the rules are Linked Data, but is is aiming to present them to the users, rather than enabling them editing the rules; (ii) visualizing separate graphs for the raw data and ontology, e.g., the Map-On [29] where two graphs, which are distinguished from each other using different colors, are created: one for the raw data structure and one for the target ontology, while the mapping rules are defined as the two graphs alignments, namely aligning the data fractions with the corresponding ontology elements; (iii) visualizing a single graph for both raw data and ontologies, e.g., RMLEditor [21] (more details regarding the RMLEditor are available in Section 2.6).

Only recently Juma [11] emerged, which is a visualization tool, not based on graph visualizations though but, instead, it provides an alternative visual representation based on the block metaphor paradigm.

2.3 Limitations and requirements

After outlining the limitations of existing solutions that generate Linked Data from (semi-)structured data, we present the factors that can improve the mapping rules to produce better Linked Data sets which integrate data from multiple data sources and interlink same entities appearing in different data sources.

2.3.1 Limitations

We identified the following limitations that prevent current Linked Data generation approaches from achieving well integrated Linked Data sets. This happens because Linked Data generation occurs on:

per-source basis Most current approaches generate Linked Data on a per-source basis, namely they consider only one source at once. As a result, data publishers can only

[9] RMLx, `http://pebbie.org/mashup/rml`

generate resources and links between data appearing within a single data source. Their mapping rules need to be aligned manually when the same resources already appear in the generated Linked Data set, but were already derived from another data source. Thus, data publishers need to redefine and replicate the patterns for the resources' IRIs definition every time they appear in a new mapping rule that refers to a new data source. Nevertheless, this is not always possible, as data included in a data source may not be sufficient to replicate the same unique identifiers (IRIs). This results in distinct IRIs for same resources, which leads to duplicates within a publisher's own Linked Data set, while interlinking of same resources has to be performed in addition afterwards.

per-format basis Most current solutions provide a *per-format* approach to generate Linked Data. To be more precise, only defining mapping rules from a certain source format (e.g., XML) are supported. In practice, data publishers need to consider various source formats to generate the desired Linked Data. Therefore, they need to install, learn, use, and maintain different tools for each format separately, which hampers their effort to ensure the integrity of their Linked Data sets even more. Alternatively, some data publishers end up implementing their own *case-specific* solutions.

lack of reusability The current solutions mapping rules are not reusable, as there is no standard formalisation for any data format, apart from relational databases, i.e., R2RML. In most cases, the mapping rules are not interoperable, as they are tied to and embedded in the implementation that executes them, which prevents their extraction and reuse across different implementations. For instance, the DBpedia dataset is generated relying on mapping rules, defined in a custom formalization, which are partially configurable and partially embedded in the DBpedia EF [17]. This prohibits the reuse of same mapping rules to generate Linked Data that describe the same model, when the data is originally serialized in different structures or formats.

2.3.2 Requirements

To achieve Linked Data sets with better integrated and richer interlinked resources, the aforementioned issues should be addressed when it is defined how Linked Data is generated, rather than later on. A set of factors that contribute to this are outlined below:

uniform and interoperable mapping rules Mapping rules that define how Linked Data is generated should be defined independently of the references to the input data. The same mapping rules may then be reused across different data sources —as long as they capture the same model— only by changing the reference to the input data source that holds the information.

For instance, a *performance* described in a JSON file (Listing 2.3) and an *exhibition* described in an XML file (Listing 2.4) may take place at the same location, indicated by an identical longitude/latitude pair. We only need a single mapping rule to describe their location, adjusted to point to respectively the JSON objects and the XML elements that hold the corresponding values or just to the same resource.

Therefore, we require a *modular language* in which references to data extracts and the mapping rules are distinct and not interdependent. Thereby, the mapping rules can be reused across different implementations for different source formats, reducing the implementation and learning costs.

robust cross-references and interlinking Redefining and replicating patterns every time a new data source is integrated should be avoided. Agents should be able to uniquely define the pattern that generates a resource in a mapping rule and refer to this mapping rule every other time this resource appears in another data source (in this way enriched). This has certain advantages: (i) possible modifications to patterns, or data values appearing in patterns that generate IRIs, are propagated to every other reference of a resource, making the interlinking more robust; (ii) taking advantage of this integrated solution, *cross-references* among sources become possible, i.e., links between resources in different data sources are defined already on generation level; and most significantly, (iii) when data publishers consider a new source, their new mapping rules are defined taking advantage of and automatically aligning with existing ones.

Extending the aforementioned example, the *venue* where the *performance* and *event* take place is the same. When Linked Data was generated based on the input data source for the *performances*, the mapping rules for the possible *venues* were defined considering certain identifiers to define their IRIs. Once Linked Data is about to be generated for *exhibitions*, the data publisher might not be able to reuse existing mapping rules for *venues* as the identifiers are not included in the dataset to replicate the same patterns. However, the *venue name* might be considered to determine the binding, and existing mapping rules can be referred to generate same IRIs. Thus, an existing resource is enriched with new attributes and the new Linked Data set is interlinked to existing ones.

scalable mapping language As the references to data extracts and mapping rules are distinct and not interdependent, the pointer to a data source's data can be adjusted to each case. Such a modular solution leads to correspondingly modular implementations that execute the mapping rules in a uniform way, independently of the input data source. They only adjust the respective extraction mechanism depending on the input data source. *Case-specific* solutions exist because complete generic solutions fail, as it is impossible to predict every potential input.

A scalable solution addresses what can be defined in a generic way for all possible different input data sources and scales over what cannot be defined generically. To support emerging needs, a generic solution needs to support references specific to each data source, and addressed on a case-specific level (where the case is determined in principle by the data format).

2.3.3 R2RML generalization

To reach a solution that overcomes the aforementioned limitations and fulfills the desired requirements, we investigated the potential of R2RML, as this is the only W3C-recommended mapping language, to be generalized. The following were identified as the most relevant aspects of R2RML for its generalization:

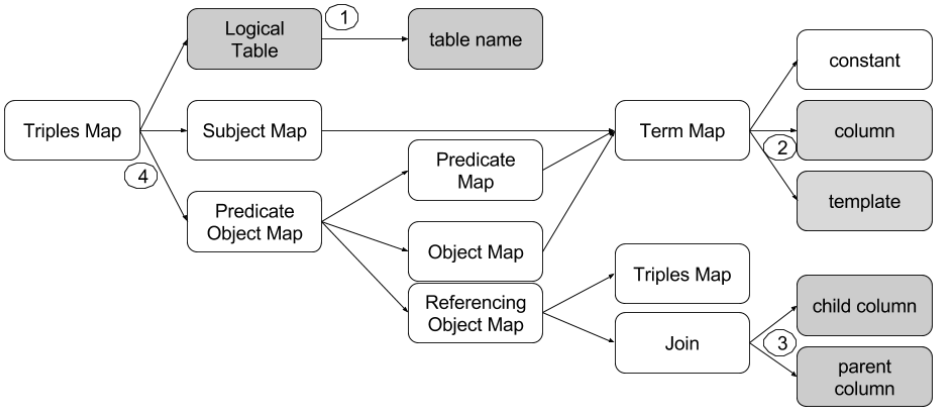


Figure 2.1: *R2RML extensible elements*

data sources coverage R2RML supports only Linked Data generation from data residing in relational databases. Nevertheless, its potential to support Linked Data generation from data residing in other data sources can be explored as well, than barely relational databases, accessed using different interfaces and having heterogeneous structures and serializations. To achieve that the rml:LogicalTable is required to be extended (see ① from Figure 2.1).

RDF term's generation R2RML supports RDF terms generation only from data derived from relational databases. Nevertheless, its potential to support RDF terms formed based on any data source can be explored, either this is a table row, or, for instance, an XML element or a JSON object. To achieve that the *column-valued* and *template-valued term map* is extended to cover every *reference-valued term map* where a *reference* may indicate any data extract derived from a data source. Therefore, the R2RML's `rr:column` property is required to be extended (see ② from Figure 2.1).

RDF triple's generation R2RML supports RDF triples generation relying only on RDF terms which were generated based on a relational database's columns values. Therefore, its potential to generate RDF triples from RDF terms can be explored which are generated relying on data from any data source. The Triples Map (`rr:TriplesMap`, see ④ from Figure 2.1) and Reference Object Map (`rr:ReferencingObjectMap`, see ③ from Figure 2.1) is required to be extended. This way, the potential to generate RDF triples based on integrated sources emerges.

2.4 RML Language

The *RDF Mapping language* (RML) [1, 4] is a general-purpose mapping language defined to express customized mapping rules that define how Linked Data is generated from heterogeneous data. RML is defined as a superset of the W3C-recommended mapping language R2RML [14], which was introduced in the previous subsection (Section 2.3.3), aiming to extend its applicability and broaden its scope.

Table 2.1: *R2RML Vs RML*

	R2RML	**RML**
data source	table name	data source description
data value	column	reference
iteration model	per row (implicit)	customly defined
source expression	SQL (implicit)	reference formulation

RML keeps the mapping rules as in R2RML, but excludes its database-specific references from the core model. The primary difference is the potential data source that is limited to a certain database in the case of R2RML, while it can be a broad set of data sources in the case of RML. Table 2.1 summarizes the RML's extensions over R2RML, entailed because of the broader set of possible input sources, while Figure 2.2 summarizes RML's elements, as opposed to Figure 2.1 which summarizes R2RML's.

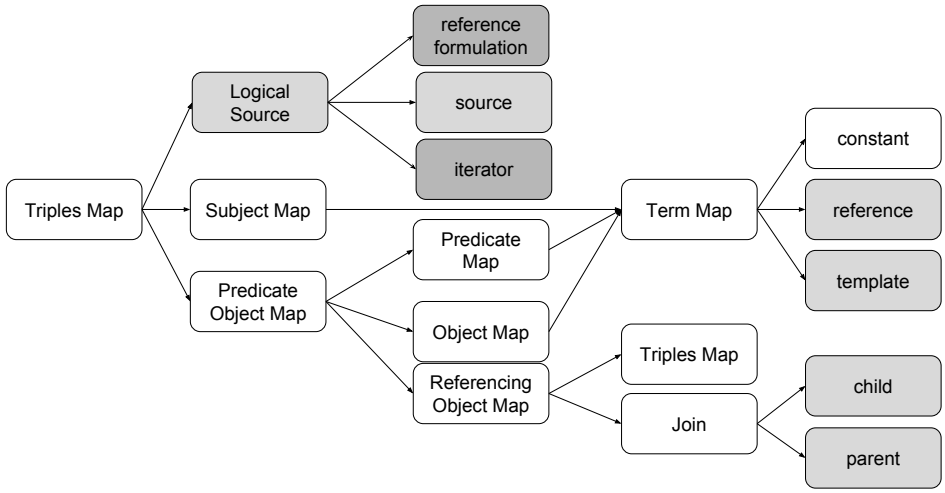


Figure 2.2: *Diagram with RML elements. In light gray the R2RML elements which were generalized are depicted. In dark gray, the extension over R2RML are depicted.*

RML provides a generic way of defining mapping rules that is easily transferable to cover references to other data structures and formats, combined with case-specific extensions, but always remains backward compatible with R2RML, as relational databases form such a specific case. RML considers that the mapping rules that define how Linked Data is generated from a sets of sources that all together describe a certain domain, can be defined in a combined and uniform way. This way, the mapping rules may be re-used across different sources that describe the same domain to incrementally form well-integrated Linked Data sets, as displayed at Figure 2.3.

An RML mapping rule follows the same syntax as R2RML. The RML vocabulary namespace is `http://semweb.mmlab.be/ns/rml#` and the preferred prefix is *rml*. More de-

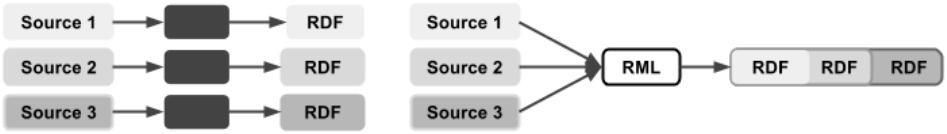


Figure 2.3: *Mapping sources without and with RML*

tails about the RDF mapping language (RML) can be found at `http://rml.io`. Defining and executing a mapping rule with RML requires that an agent provides a valid and well-formatted *input dataset* to be used to generate Linked Data and mapping rules (*mapping document*) according to which the rules will be executed to generate the data's representation using the RDF framework (*output dataset*).

```
{ ...  "Performance" :
         { "Perf_ID": "567",
           "Venue": {  "Name": "STAM",
                       "Venue_ID": "78" },
           "Location": { "long": "3.717222",
                         "lat": "51.043611" } } , ... }
```

Listing 2.3: *data source: `performances.json`*

```
<Events> ...
  <Exhibition id="398">
    <Venue> STAM </Venue>
    <Location>
      <lat>51.043611</lat>
      <long>3.717222</long>
    </Location>
  </Exhibition> ... ...
</Events>
```

Listing 2.4: *data source: `exhibitions.xml`*

Extracts of two heterogeneous input sources are displayed. Listing 2.3 provides an input data source in JSON format, while Listing 2.4 provides another input data source in XML format. Moreover, an example of a corresponding mapping rule is displayed at Listing 2.6 and the produced output at Listing 2.5.

Logical Source A Logical Source (`rml:LogicalSource`, Listing 2.6, Lines 7, 12 and 17) extends R2RML's Logical Table and is used to determine the input source with the data to be mapped. The R2RML Logical Table definition determines a database's table, using the Table Name (`rr:tableName`). In the case of RML, a broader reference to any data source is required. Thus, the Logical Source and source (`rml:source`, Lines 8, 13 and 18) are introduced respectively to specify the input dataset. A Logical Source's complete description, besides the source, it also includes a reference formulation and, optionally, an iterator (both of them are described in the next paragraphs in details).

```
ex:567       ex:venue    ex:78 ;
             ex:location ex:3.717222,51.043611 .
ex:398       ex:venue    ex:78 ;
             ex:location ex:3.717222,51.043611 .
ex:3.717222,51.043611   ex:lat      ex:3.717222
                        ex:long     ex:51.043611.
```

Listing 2.5: *The expected output.*

Reference Formulation RML needs to deal with different data structures, formats, and serialisations which use different ways to refer to their data fragments. But, as RML aims to be generic, not a uniform way of referring to each data source's fragments is defined. R2RML uses columns' names for this purpose. In the same context, RML considers that any reference to the Logical Source should be defined in a form relevant to the input data, e.g., XPath for XML files or JSONPath for JSON files. To this end, the Reference Formulation (`rml:referenceFormulation`, Listing 2.6, Lines 9, 14 and 19) declaration is introduced indicating the formulation (for instance, a standard or a query language) used to refer to a data source's fragments. Note that the identifiers are defined independently of the RML language and its vocabulary. The following reference formulations were indicatively defined for the most broadly used reference formulations: `ql:CSV`, `ql:XPath`, `ql:JSONPath`, `ql:CSS3` and `ql:WikiText`.

Iterator While in R2RML it is already known that a *per-row* iteration occurs, as RML remains generic, the iteration pattern, if any, can not always be implicitly assumed, but it needs to be determined. Thereafter, the iterator (`rml:iterator`, Listing 2.6, Lines 10, 15 and 20) is introduced. The iterator determines the iteration pattern over the input source and specifies the extract of data which are used for the RDF terms' generation during each iteration. For instance, the `"$[*]"` determines the iteration over a JSON file that occurs over the object's outer level. The iterator is not required in the case of tabular sources as the default *per-row* iteration is implied or if there is no need to iterate over the input data.

Reference A *column-valued term map*, according to R2RML, is defined using the property `rr:column` which determines a column's name. In the case of RML, a more generic property is introduced, the reference (`rml:reference`, Listing 2.6, Lines 64 and 67). Its value must be a valid reference to the data of the input dataset. Therefore, the reference's value should be a valid expression according to the Reference Formulation defined at the Logical Source, as well as the string template used in the definition of a *template-valued term map* and the iterator's value. For instance, the iterator, the subject's *template-valued term map* and the object's *reference-valued term map* are all valid JSONPath expressions.

Referencing Object Map The last aspect of R2RML that is extended in RML is the Referencing Object Map. The join condition's child reference (`rr:child`, Line 60) indicates the reference to the data value (using a reference) of the Logical Source that contains the Referencing Object Map. The join condition's *parent reference* (`rr:parent`, Line 61) indicates the reference to the data extract (reference) of the Referencing Object Map's Parent Triples

Map. The reference is specified using the Reference Formulation defined at the current Logical Source. The join condition's *parent reference* indicates the reference to the data extract (reference) of the Parent Triples Map. The reference is specified using the Reference Formulation defined at the Parent Triples Map Logical Source definition. Therefore, the *child reference* and the *parent reference* of a join condition may be defined using different Reference Formulations, if the Triples Map refers to sources of different formats.

2.5 RML Extensions

RML has certain inherited limitations from R2RML, mainly due to R2RML's assumption that data transformation, computation, or filtering are performed before generating RDF triples from the data of a relational database. In R2RML, this can be achieved by defining a SQL view in the input database. Therefore, certain extensions were required to increase RML's expressivity, as not all data structures, formats and serializations have corresponsding languages or formalizations which allow to do so. To this end, RML was extended to support (i) *graceful degradation*, if a certain mapping rule fails (Section 2.5.1); and (ii) *data transformations*, if data transformations or computations are required (Section 2.5.2).

2.5.1 Graceful Degradation

A mapping rule might fail to generate an RDF term. For instance, the input data source might have two fields, e.g., "field_A" and "field_B", with information for the Web site of an entity. The default mapping rule might generate RDF terms relying on "field_A", whereas, on its own turn, might sometimes be null, while "field_B" might have an entry which can be used instead. Thus, RML was extended to allow an alternative mapping strategy which ensures that a more complete Linked Data set is generated.

The namespace with the extension for graceful degradation is `http://semweb.mmlab.be/ns/crml#` and the preferred prefix is *crml*. Details about the extension can be found at `http://rml.io`.

To support graceful degradation with RML, the Fallback Map (`crml:FallbackMap`) was introduced. A Fallback Map can be defined to generate an RDF term if the default Term Map does not generate an RDF term. A Fallback Map specifies the Term Map(s) which are employed if the default Term Map does not generate an RDF term. RML was extended with the crml:fallback property (Listing 2.7, line 6), which specifies the alternative Term Map(s) – the Fallback Map (line 8). The crml:fallback property has a Predicate Object Map or a Term Map as domain and a Predicate Object Map or Term Map as range respectively.

2.5.2 Data Transformations

RML does not foresee describing data transformations, as an inherited dependency from R2RML. Nevertheless, agents might not desire to use the raw values as extracted from

```
@prefix rr: <http://www.w3.org/ns/r2rml#>.
@prefix ex: <http://example.com/ns#>.
@prefix rml:<http://semweb.mmlab.be/ns/rml\#>.

=== Logical Source ===

<#LogicalSource_JSON_Performance>
  rml:source "http://ex.com/performances.json";
  rml:referenceFormulation ql:JSONPath;
  rml:iterator "$.Performance.[*]" .

<#LogicalSource_JSON_Venue>
  rml:source "http://ex.com/performances.json";
  rml:referenceFormulation ql:JSONPath;
  rml:iterator "$.Performance.Venue.[*]" .

<#LogicalSource_XML_Exhibition>
  rml:source "http://ex.com/exhibitions.xml";
  rml:referenceFormulation ql:XPath;
  rml:iterator "/Events/Exhibition" .

=== Triples Maps ===

<#PerformancesMapping>
  rml:logicalSource <#LogicalSource_JSON_Performance> ;
  rr:subjectMap <#SM_Performance> ;
  rr:predicateObjectMap <#POM_PerformanceVenue> ;
  rr:predicateObjectMap <#POM_PerformanceLocation> .

<#VenueMapping>
  rml:logicalSource <#LogicalSource_JSON_Venue> ;
  rr:subjectMap <#SM_Venue> .

<#LocationMapping>
  rml:logicalSource <#LogicalSource_JSON_Performance> ;
  rr:subjectMap <#SM_Location> ;
  rr:predicateObjectMap <#POM_long> ;
  rr:predicateObjectMap <#POM_lat> .

<#ExhibitionMapping>
  rml:logicalSource <#LogicalSource_XML_Exhibition> ;
  rr:subjectMap <#SM_Exhibition> ;
  rr:predicateObjectMap <#POM_ExhibitionLocation> ;
  rr:predicateObjectMap <#POM_ExhibitionVenue> .

=== Term Maps ===

<#POM_PerformanceVenue> rr:predicate ex:venue;
  rr:objectMap [ rr:parentTriplesMap <#VenueMapping> ] .

<#POM_PerformanceLocation> rr:predicate ex:location;
  rr:objectMap [ rr:parentTriplesMap <#LocationMapping> ] .

<#POM_ExhibitionLocation> rr:predicate ex:location;
  rr:objectMap [ rr:parentTriplesMap <#LocationMapping> ] ].

<#POM_ExhibitionVenue> rr:predicate ex:venue;
  rr:objectMap [ rr:parentTriplesMap <#VenueMapping> ;
  rr:joinCondition [
    rr:child "$.Performance.Venue.Name" ;
    rr:parent "/Events/Exhibition/Venue" ] ] .

<#POM_long> rr:predicate ex:long;
  rr:objectMap [ rml:reference "long" ] .

<#POM_lat> rr:predicate ex:lat;
  rr:objectMap [ rml:reference "lat" ] .

<#SM_Performance> rr:template "http://ex.com/{Perf_ID}" .
<#SM_Venue> rr:template "http://ex.com/{Venue_ID}" .
<#SM_Location> rr:template "http://ex.com/{lat},{long}" .
<#SM_Exhibition> rr:template "http://ex.com/{@id}" .
```

Listing 2.6: *RML mapping rules*

```
1  <#POM_name>
2    rr:predicate ex:name;
3    rr:objectMap <#OM_name> .
4
5  <#OM_name> rml:reference "name" ;
6  <#OM_name> crml:fallback <#OM_name2> .
7
8  <#OM_name2> rml:reference "name2" .
```

Listing 2.7: *Fallback Map for Graceful Degradation*

a data source to determine how and whether an RDF term should be generated or not. For instance, an input dataset might not have the country names capitalized, for instance, *japan* and *south korea* instead of *Japan* and *South Korea*, respectively.

Therefore, an extension over the core RML language is required to support data transformations too. The namespace for RML's extensions with regard to data transformations is `http://semweb.mmlab.be/ns/fnrml#` and the preferred prefix is *fnrml*. More details about the data transformations supported by RML can be found at `http://fno.io`.

To achieve that, a new type of Term Map was introduced in addition to the one that generates RDF terms, the Function Map (`fnml:FunctionMap`) [17]. A Function Map is a Term Map that is generated by executing a function. In contrast to an RDF Term Map that considers references from a Logical Source to generate an RDF term, a Function Term Map considers references from a Logical Source to execute a function. Once the function is executed, its output value is the term generated by this Function Map. To this end, the `fnml:functionMap` property was introduced to indicate which instance of a function needs to be executed to generate an output and considering which values. Such a function can be described using the Function Ontology (FnO) [15, 16].

```
1  @prefix fno: <http://semweb.datasciencelab.be/ns/function#> .
2  @prefix grel: <http://semweb.datasciencelab.be/ns/grel#> .
3
4
5  grel:toUppercase a fno:Function ;
6    fno:name "to uppercase" ;
7    rdfs:label "to uppercase" ;
8    dcterms:description "converts a string into uppercase" ;
9    fno:expects ( grel:string ) ;
10   fno:output ( grel:stringOut ) .
```

Listing 2.8: *GREL functions expressed as FnO statements*

The *Function Ontology* (FnO) allows agents to declare functions without specifying an implementation. This way, the functions are uniformly and unambiguously described, independently of the technology that implements them. This allows a mapping processor to parse any function description, retrieve a suitable implementation, and then trigger its execution. A *function* (`fno:Function`, Listing 2.8, Line 5) is an activity which has input parameters, output, and implements certain algorithm(s). A *parameter* (`fno:Parameter`, Line 9) is the description of a function's input value. An *output* (`fno:Output`, Line 10) is the description of the function's output value. An *execution* (`fno:Execution`, Listing 2.9,

Line 11) assigns values to the parameters of a function. For instance, `grel:toUppercase` (Listing 2.9, Line 14) is a function that renders a given string into its corresponding uppercase value. It expects a string, indicated by the `grel:string` property (Line 9) as input. After an Execution (line 11) is instantiated and executed, its result is indicated by the `grel:stringOutput` property and forms the value returned by the Function Map.

GREL[10] is a list of functions, supported by the General Refine Expression Language (GREL) and implemented in Open Refine. The complete description of all GREL functions, using FnO, is available at `https://github.com/FnOio/grel-functions-fno`. A proof-of-concept implementation in Java[11] executes the GREL functions. A generic Function Processor[12] then uses the function declarations described in FnO to retrieve and execute their relevant implementations. The latter was included in the third version of the RMLMapper to handle the data values transformations (Chapter 3).

```
1  <#POM_name> rr:predicate ex:name ;
2    rr:objectMap <#FM_name> .
3
4  <#FM_name> a fnml:FunctionMap ;
5    fnml:functionMap <#CapitalizationMap> .
6
7  <#CapitalizationMap>
8    rml:logicalSource <#LogicalSource_X> ;
9    rr:subjectMap [
10     rr:termType rr:BlankNode ;
11     rr:class fno:Execution ] ;
12   rr:predicateObjectMap [
13     rr:predicate fno:executes ;
14     rr:objectMap [ rr:constant grel:toUppercase ] ] ;
15   rr:predicateObjectMap [
16     rr:predicate grel:string ;
17     rr:objectMap <#OM_name> ] .
18
19 <#OM_name> rml:reference "name" .
```

Listing 2.9: *Function Map*

2.6 RML Editor

Mapping languages allow to define how RDF terms and triples are generated from input datasets whose data extracts are used to form RDF terms and their relationships are annotated with ontology terms. Nevertheless, knowledge of the mapping language is required to define such mapping rules, while manually editing them requires a substantial amount of human effort [21]. This turns to be even more complicated when the agents who edit the mapping rules are not Linked Data experts.

We introduced the RMLEditor to facilitate mapping rules editing. It is a graph-based visualization tool for mapping rules that define how Linked Data is generated from multiple heterogeneous data sources [21]. It allows human agents to create and edit mapping

[10] GREL, `https://github.com/OpenRefine/OpenRefine/wiki/GREL-Functions`
[11] GREL functions in Java, `https://github.com/FnOio/grel-functions-java`
[12] Function Processor, `github.com/FnOio/function-processor-java`

rules, and preview the generated RDF terms and triples. This way, human agents who have domain knowledge, but no knowledge about the mapping language or the Semantic Web in general, can still generate their desired Linked Data on their own. The RMLEditor was used in two use cases which are included in this work (Sections 6.2 and 6.3).

The RMLEditor separates the presentation, logic of the mapping process, and access to the input data sources, using the presentation, application, and data access layer, respectively. The presentation layer consists of three panels which are used by the human agents to define the mapping rules. The application layer processes the interactions triggered through the panels and the data access layer deals with the required data, mapping rules, or ontologies which are required to define the mapping rules. Once the mapping rules are defined, the RML mapping rules might be exported or executed by the underlying RMLProcessor (see Chapter 3) to generate the desired Linked Data.

The RMLEditor's graphical user interface (GUI) consists of three views, the so-called panels. The RMLEditor implements the features for mapping rules editors and allows human agents to define their rules following the editing approach they prefer or fits their needs.

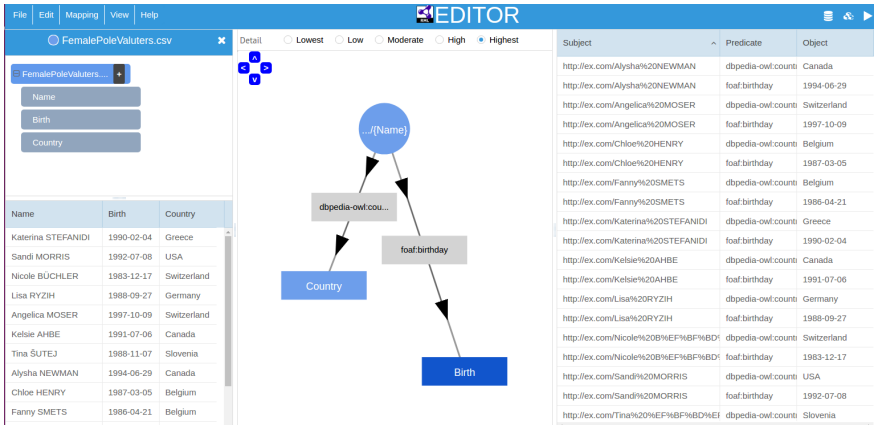


Figure 2.4: *RMLEditor Graphical User Interface*

Panels The RMLEditor's GUI consists of three panels, as shown in Figure 2.4:

Input Panel The *Input Panel* deals with the input data sources presentation. The Input Panel, on its own turn is divided into two subpanels. The top panel displays the data sources structure. The bottom panel shows the raw data of the source. Each data source is assigned with a unique color.

Modeling Panel The *Modeling Panel* uses a graph-based visualization. The color of each node and edge has the color of the data source whose data extract is used to form the RDF term. The human agents can use different ontologies and vocabularies to define the semantic annotations which can be either defined locally or online. The Linked Open Vocabularies (LOV) [33] and `http://prefix.cc` are integrated. Human agents can consult LOV to discover relevant classes and properties and `http://prefix.cc` to search for well-known prefixes and namespaces.

Results Panel The *Results Panel* shows the generated RDF triples when mapping rules defined in the Modeling Panel are executed for the data in the Input Panel.

Features The RMLEditor implements the features, which mapping rules editors should support, through its three panels [19]: The RMLEditor allows human agents to create mapping rules without being dependent on the language syntax (*independence of the mapping language*), the number of data sources which are required to represent the desired domain knowledge (*support multiple data sources*) or their original structure, format or serialization (*support heterogeneous data sources*). Moreover, the RMLEditor allows agent to model complementary or overlapping aspects of certain domain knowledge (*support multiple ontologies*) and choose the most adequate modeling approach for their needs (*support multiple modeling approaches*). Last, the RMLEditor allows human agents to follow the most adequate workflow to define their mapping rules (*support non-linear workflows*), and their scope lies beyond their execution (*independence of execution*).

Editing Approaches The RMLEditor supports all mapping generation approaches [20]. This way, it covers the agents' different needs and alternative usage scenarios:

data-driven Agents may consider any number of input data sources in any structure, format, or serialization and define mapping rules focusing on the data, namely each data fractions is associated to a corresponding mapping rule, i.e., the adequate schema(s) is associated to the data.

schema-driven Agents may consider an existing semantic schema as the basis for defining mapping rules and associating them to the applicable data from the input data source(s), i.e., the appropriate data is associated to the schema(s).

model-driven Agents can also start by defining the model which represents the knowledge domain, by defining abstract mapping rules, namely the entities, as well as their attributes and relationships without indicating the ontologies and vocabularies or the input data fractions to be used. This means that the input data and schema are associated to the model.

result-driven Agents can define mapping rules based on desired results, i.e., the model and schema is derived from the desired output, and the input data are associated.

Conclusions

In this chapter, we presented how mapping rules can be declaratively described using mapping languages independently of the structure and format the data originally has, how they can be aligned to complementary describe a knowledge domain, and how data transformations can be applied. Last, we have seen that user-friendly interfaces might be used to facilitate human agents in editing mapping rules.

References

[1] **Anastasia Dimou** and Miel Vander Sande. RML: RDF Mapping Language. W3C Recommendation, W3C, September 2014. http://rml.io/spec.html.

[2] **Anastasia Dimou**, Miel Vander Sande, Pieter Colpaert, Erik Mannens, and Rik Van De Walle. Extending R2RML to a Source-independent Mapping Language for RDF. In *Proceedings of the 2013th International Conference on Posters & Demonstrations Track - Volume 1035*, ISWC-PD'13, pages 237–240, Aachen, Germany, Germany, 2013. CEUR-WS.org. URL http://dl.acm.org/citation.cfm?id=2874399.2874459.

[3] **Anastasia Dimou**, Miel Vander Sande, Jason Slepicka, Pedro Szekely, Erik Mannens, Craig Knoblock, and Rik Van de Walle. Mapping Hierarchical Sources into RDF using the RML Mapping Language. In *Proceedings of the 2014 IEEE International Conference on Semantic Computing*, ICSC '14, pages 151–158, Washington, DC, USA, 2014. IEEE Computer Society. ISBN 978-1-4799-4003-5. doi: 10.1109/ICSC.2014.25. URL http://dx.doi.org/10.1109/ICSC.2014.25.

[4] **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. URL http://ceur-ws.org/Vol-1184/ldow2014_paper_01.pdf.

[5] Marcelo Arenas, Alexandre Bertails, Erik Prud'hommeaux, and Juan Sequeda. A Direct Mapping of Relational Data to RDF. W3C Recommendation, W3C, September 2012. https://www.w3.org/TR/rdb-direct-mapping/.

[6] Stefan Bischof, Stefan Decker, Thomas Krennwallner, Nuno Lopes, and Axel Polleres. Mapping between rdf and xml with xsparql. *Journal on Data Semantics*, 1(3):147–185, 2012. ISSN 1861-2040. doi: 10.1007/s13740-012-0008-7. URL http://dx.doi.org/10.1007/s13740-012-0008-7.

[7] Christian Bizer. D2R MAP - A Database to RDF Mapping Language, 2003. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.5.862.

[8] Scott Boag, Don Chamberlin, Mary F Fernández, Daniela Florescu, Jonathan Robie, Jérôme Siméon, and Mugur Stefanescu. XQuery 1.0: An XML query language. W3C Recommendation, W3C, 2002. https://www.w3.org/TR/xquery/.

[9] James Clark and Steve DeRose. XML Path Language (XPath) Version 1.0. W3C Recommendation, W3C, November 1999. http://www.w3.org/TR/xpath.

[10] Dan Connolly. Gleaning Resource Descriptions from Dialects of Languages (GRDDL). W3C Recommendation, W3C, September 2007. http://www.w3.org/TR/grddl/.

[11] Ademar Crotti Junior, Christophe Debruyne, and Declan O'Sullivan. Juma: an Editor that Uses a Block Metaphor to Facilitate the Creation and Editing of R2RML Mappings. In Harald Sack, Giuseppe Rizzo, Nadine Steinmetz, Dunja Mladenić, Sören Auer, and Christoph Lange, editors, *The Semantic Web: ESWC 2017 Satellite*

Events, Portorož, Slovenia, May 28 – June 1, 2017, Revised Selected Papers, Cham, 2017. Springer International Publishing.

[12] Richard Cyganiak. Tarql – SPARQL for Tables: Turn CSV into RDF using SPARQL syntax. Technical report, W3C, January 2015. `http://tarql.github.io/`.

[13] Richard Cyganiak, Chris Bizer, Jörg Garbers, Oliver Maresch, and Christian Becker. The D2RQ Mapping Language. Technical report, W3C, March 2012. `http://d2rq.org/d2rq-language`.

[14] Souripriya Das, Seema Sundara, and Richard Cyganiak. R2RML: RDB to RDF Mapping Language. W3C Recommendation, W3C, September 2012. `http://www.w3.org/TR/r2rml/`.

[15] Ben De Meester and **Anastasia Dimou**. The Function Ontology. Unofficial Draft, imec, October 2016. `http://users.ugent.be/~bjdmeest/function/`.

[16] Ben De Meester, **Anastasia Dimou**, Ruben Verborgh, and Erik Mannens. An Ontology to Semantically Declare and Describe Functions. In Harald Sack, Giuseppe Rizzo, Nadine Steinmetz, Dunja Mladenić, Sören Auer, and Christoph Lange, editors, *The Semantic Web: ESWC 2016 Satellite Events, Heraklion, Crete, Greece, May 29 – June 2, 2016, Revised Selected Papers*, pages 46–49, Cham, 2016. Springer International Publishing. ISBN 978-3-319-47602-5. doi: 10.1007/978-3-319-47602-5_10. URL `http://dx.doi.org/10.1007/978-3-319-47602-5_10`.

[17] Ben De Meester, Wouter Maroy, **Anastasia Dimou**, Ruben Verborgh, and Erik Mannens. Declarative Data Transformations for Linked Data Generation: The Case of DBpedia. In Eva Blomqvist, Diana Maynard, Aldo Gangemi, Rinke Hoekstra, Pascal Hitzler, and Olaf Hartig, editors, *The Semantic Web: 14th International Conference, ESWC 2017, Portorož, Slovenia, May 28 – June 1, 2017, Proceedings, Part II*, pages 33–48, Cham, 2017. Springer International Publishing. ISBN 978-3-319-58451-5. doi: 10.1007/978-3-319-58451-5_3. URL `http://dx.doi.org/10.1007/978-3-319-58451-5_3`.

[18] Matthias Hert, Gerald Reif, and Harald C. Gall. A comparison of RDB-to-RDF Mapping Languages. In *Proceedings of the 7th International Conference on Semantic Systems*, I-Semantics '11, pages 25–32, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0621-8. doi: 10.1145/2063518.2063522. URL `http://doi.acm.org/10.1145/2063518.2063522`.

[19] Pieter Heyvaert, **Anastasia Dimou**, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. Towards a Uniform User Interface for Editing Mapping Definitions. In *Proceedings of the 4th Workshop on Intelligent Exploration of Semantic Data*, volume 1472 of *CEUR Workshop Proceedings*, October 2015. URL `http://ceur-ws.org/Vol-1472/IESD_2015_paper_4.pdf`.

[20] 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*, volume 1486 of *CEUR Workshop Proceedings*, October 2015. URL `http://ceur-ws.org/Vol-1486/paper_70.pdf`.

[21] 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 Harald Sack, Eva Blomqvist, Mathieu d'Aquin, Chiara Ghidini, Paolo Simone Ponzetto, and Christoph Lange, editors, *The Semantic Web – Latest Advances and New Domains (ESWC 2016)*, volume 9678 of *Lecture Notes in Computer Science*, pages 709–723. Springer, 2016. ISBN 978-3-319-34129-3. doi: 10.1007/978-3-319-34129-3_43. URL http://dx.doi.org/10.1007/978-3-319-34129-3_43.

[22] Bernadette Hyland, Ghislain Atemezing, and Boris Villazón-Terrazas. Best Practices for Publishing Linked Data. W3C Working Group Note, W3C, January 2014. http://www.w3.org/TR/ld-bp/.

[23] Michael Kay and Saxonica. XSL Transformations (XSLT) Version 2.0. W3C Recommendation, W3C, January 2007. http://www.w3.org/TR/xslt20/.

[24] Christoph Lange. Krextor - an Extensible Framework for Contributing Content Math to the Web of Data. In *Proceedings of the 18th Calculemus and 10th International Conference on Intelligent Computer Mathematics*, MKM'11, pages 304–306, Berlin, Heidelberg, 2011. Springer-Verlag. ISBN 978-3-642-22672-4. URL http://dl.acm.org/citation.cfm?id=2032713.2032745.

[25] Andreas Langegger and Wolfram Wöß. Xlwrap – querying and integrating arbitrary spreadsheets with sparql. In Abraham Bernstein, David R. Karger, Tom Heath, Lee Feigenbaum, Diana Maynard, Enrico Motta, and Krishnaprasad Thirunarayan, editors, *The Semantic Web - ISWC 2009: 8th International Semantic Web Conference, ISWC 2009, Chantilly, VA, USA, October 25-29, 2009*, pages 359–374, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. ISBN 978-3-642-04930-9. URL http://dx.doi.org/10.1007/978-3-642-04930-9_23.

[26] Yannis Marketakis, Nikos Minadakis, Haridimos Kondylakis, Konstantina Konsolaki, Georgios Samaritakis, Maria Theodoridou, Giorgos Flouris, and Martin Doerr. X3ML mapping framework for information integration in cultural heritage and beyond. *International Journal on Digital Libraries*, pages 1–19, 2016. ISSN 1432-1300. doi: 10.1007/s00799-016-0179-1. URL http://dx.doi.org/10.1007/s00799-016-0179-1.

[27] Martin J. O'Connor, Christian Halaschek-Wiener, and Mark A. Musen. Mapping master: A flexible approach for mapping spreadsheets to owl. In Peter F. Patel-Schneider, Yue Pan, Pascal Hitzler, Peter Mika, Lei Zhang, Jeff Z. Pan, Ian Horrocks, and Birte Glimm, editors, *The Semantic Web – ISWC 2010: 9th International Semantic Web Conference, ISWC 2010, Shanghai, China, November 7-11, 2010, Revised Selected Papers, Part II*, pages 194–208, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. ISBN 978-3-642-17749-1. doi: 10.1007/978-3-642-17749-1_13. URL http://dx.doi.org/10.1007/978-3-642-17749-1_13.

[28] Kunal Sengupta, Peter Haase, Michael Schmidt, and Pascal Hitzler. Editing R2RML Mappings Made Easy. In *Proceedings of the 2013th International Conference on Posters & Demonstrations Track - Volume 1035*, ISWC-PD'13, pages 101–104, Aachen, Germany, Germany, 2013. CEUR-WS.org. URL http://dl.acm.org/citation.cfm?id=2874399.2874425.

[29] Álvaro Sicilia, German Nemirovski, and Andreas Nolle. Map-On: A web-based editor for visual ontology mapping. *Semantic Web*, (Preprint):1–12, 2016. URL http://www.semantic-web-journal.net/system/files/swj1266.pdf.

[30] Dimitrios-Emmanuel Spanos, Periklis Stavrou, and Nikolas Mitrou. Bringing Relational Databases into the Semantic Web: A Survey. *Semantic Web Journal*, 3 (2):169–209, April 2012. ISSN 1570-0844. doi: 10.3233/SW-2011-0055. URL http://dx.doi.org/10.3233/SW-2011-0055.

[31] Claus Stadler, Jörg Unbehauen, Patrick Westphal, Mohamed Ahmed Sherif, and Jens Lehmann. Simplified RDB2RDF Mapping. In *Proceedings of the WWW2015 Workshop on Linked Data on the Web*, volume 1409 of *CEUR Workshop Proceedings*, 2015. URL http://ceur-ws.org/Vol-1409/paper-09.pdf.

[32] Jeni Tennison, Gregg Kellogg, and Ivan Herman. Model for Tabular Data and Metadata on the Web. W3C Working Draft, W3C, April 2015. http://www.w3.org/TR/2015/WD-tabular-data-model-20150416/.

[33] Pierre-Yves Vandenbussche, Ghislain A Atemezing, María Poveda-Villalón, and Bernard Vatant. Linked Open Vocabularies (LOV): a gateway to reusable semantic vocabularies on the Web. *Semantic Web*, 8(3):437–452, 2017. URL http://www.semantic-web-journal.net/content/linked-open-vocabularies-lov-gateway-reusable-semantic-vocabularies-web-1.

Πάντα στον νου σου νάχεις την Ιθάκη.
Το φθάσιμον εκεί είν' ο προορισμός σου.
Αλλά μη βιάζεις το ταξείδι διόλου.

Ιθάκη, Καβάφης

Keep Ithaka always in your mind.
Arriving there is what you are destined for.
But do not hurry the journey at all.

Ithaca, Cavafy

3

Execution

Methodologies and implementation for executing mapping rules definition independently of their definition

Decoupling the mapping rules representation from the implementation that executes them, requires designing algorithms and implementing corresponding systems which efficiently execute those mapping rules. In this chapter, we outline (i) factors which determine alternative approaches for executing rule-based Linked Data generation, (ii) the RMLMapper, a tool for Linked Data generation from heterogeneous data that implements an alternative for each option, and (iii) results of its evaluation over heterogeneous data.

chapter's outline In this chapter, we introduce Linked Data execution (Section 3.1). In Section 3.2, the state of the art on tools for Linked Data generation is summarized. In Section 3.3, different approaches for performing Linked Data generation are presented. Then in Section 3.4 we present the RMLMapper, and in Section 3.5 we present the results of RMLMapper's evaluation. Last, we discuss our conclusions and future work.

chapter's sources This chapter is based on the following papers:

1. **Anastasia Dimou**, Ben De Meester, Pieter Heyvaert, Ruben Verborgh, Steven Latré, and Erik Mannens. RML Mapper: a tool for uniform Linked Data generation from heterogeneous data. *Semantic Web Journal*, 2017. (under review)

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. URL `http://ceur-ws.org/Vol-1184/ldow2014_paper_01.pdf`

3. Wouter Maroy, **Anastasia Dimou**, Dimitris Kontokostas, Ben De Meester, Ruben Verborgh, Jens Lehmann, Erik Mannens, and Sebastian Hellmann. Sustainable Linked Data Generation: The Case of DBpedia. In Claudia d'Amato, Miriam Fernandez, Valentina Tamma, Freddy Lecue, Philippe Cudré-Mauroux, Juan Sequeda, Christoph Lange, and Jeff Heflin, editors, *The Semantic Web – ISWC 2017: 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part II*, pages 297–313, Cham, 2017. Springer International Publishing. ISBN 978-3-319-68204-4. doi: 10.1007/978-3-319-68204-4_28. URL `https://doi.org/10.1007/978-3-319-68204-4_28`

3.1 Introduction

Generating Linked Data from heterogeneous data sources remains a complicated and intensive engineering process, despite the significant number of existing tools, because, they provide their own –thus not interoperable– *format-* and *source-specific* approaches.

Mapping languages detach the rules which define how Linked Data is generated from the implementation that executes them. This renders the rules interoperable across different systems, whilst the systems become use-case independent, as the mapping rules cover the use-case specific details. Nevertheless, to leverage these mapping rules, we need to design algorithms and implement corresponding systems which efficiently execute those mapping rules. Such a Linked Data generation algorithm needs to cover different factors which affect the generation process, while taking into consideration data heterogeneity.

Those different factors determine how an algorithm will be designed and a corresponding system will be implemented. A *rule-based Linked Data generator* generates Linked Data from raw (semi-)structured data according to certain rules, the so-called *mapping rules*. Its function involves (i) fetching data and mapping rules; and (ii) applying mapping rules to data to semantically annotate and interlink them. This way, the desired Linked Data is generated. A use case's needs and requirements determines which one of the alternative approaches to perform a Linked Data generation process is the most adequate.

How such algorithms should be designed and how corresponding implementations should be implemented, was not thoroughly investigated so far. So far, most agents opt for ad-hoc solutions, their choices are driven by the corresponding use case, while reuse rarely occurs [3]. Moreover, the fact that different implementations do not concretely describe the algorithms that drive their implementation, while optimizations are applied according to use cases, decreases the chances of the tool to be reused.

Our evaluation shows that a general-purpose rule-based Linked Data generator (i) does generate RDF terms and triples from heterogeneously structured and formatted data sources, (ii) in reasonable time, of the same order of magnitude as custom generators for the same data source, and (iii) without being affected by the data's structure and format.

In this chapter, we present a list of key factors and the RMLMapper, a Linked Data generator from heterogeneous (semi-structured) data derived from multiple data sources. The rationale behind our tool is to cover an alternative option of one or more of these factors and provide a single, modular, and extensible tool that enables agile, fine-grained, and uniform Linked Data generation from data in various heterogeneous formats. By these means, we aim to ensure that data owners are able to generate Linked Data without being restricted due to the data structure, format, serialization or access interface.

3.2 State of the Art

To the best of our knowledge, there are no other implementations that *homogeneously* deal with Linked Data generation from data in different structures, formats and serializa-

tions, namely no tools provide the same functionality for heterogeneous data so far. Therefore, we firstly discuss other tools that generate Linked Data from certain or multiple data formats (Section 3.2.1), and existing R2RML implementations (Section 3.2.2).

3.2.1 Structure- and format-specific tools

In the past, *structure-* and even more *format-specific* tools, i.e., dedicated to the original data source, prevailed for Linked Data generation. We identify two major categories: tools for *tabular-structured* (Section 3.2.1.1) and *hierarchical-structured data* (Section 3.2.1.2). In the latter case, we focus on tools for data in XML format, as there exist no generic tools for data in JSON format, to the best of our knowledge.

3.2.1.1 Tabular data

There are multiple tools which were developed to generate Linked Data from tabular-structured data. Although different tools refer to data residing in databases and others to data in spreadsheets or CSV format. Below, we list a few of the most well-known tools for databases which rely on another language for defining the mapping rules than R2RML.

Triplify Triplify [13] is a light-weight software component, which can be easily integrated into and deployed by the numerous, widely installed Web applications. Triplify is based on mapping HTTP-URI requests onto relational database queries and transforming the resulting relations into RDF statements. Triplify neither defines nor requires to use a new mapping language, but exploits and extends certain SQL notions with suitable conventions for transforming database query results (or views) into Linked Data.

D2RQ The D2RQ platform[1] is a system for accessing relational databases as virtual, read-only RDF graphs. It offers RDF-based access to the content of relational databases without having to replicate it into a triplestore. It consists of (i) the D2RQ engine which relies on D2RQ mapping rules to rewrite Jena API calls to SQL queries which, on their own turn, are executed against the underlying database, and several implementations already exist[2]; and (ii) the D2RQ server, a tool for publishing relational databases on the Semantic Web. It provides a Linked Data view, an HTML view for debugging, and a SPARQL endpoint.

A drawback of tabular-structured data that do not reside in a relational database, such as spreadsheets and comma-separated values (CSV), is their simplicity which often results in tables that do not follow the database design best practices [32]. Compared to relational databases, which have fixed schema, data types and integrity constraints, spreadsheets and CSV are more difficult because the implicit schema needs to be captured [40]. Therefore, dedicated tools emerged. A detailed discussion about tools for translating spread-

[1] D2RQ, `http://d2rq.org/`

[2] R2RML implementations, `http://www.w3.org/2001/sw/rdb2rdf/wiki/Implementations`

sheets to Linked Data that also covers tools which are not clearly related to RDF generation is available at [40]. Below, we mention indicatively a few of them:

Tarql Tarql[3] [23] also follows a *query-driven approach* to generate Linked Data from data which is originally in CSV format. Extracts from the data in CSV format are input into the query. This allows manipulation of CSV data relying on SPARQL syntax, and in particular Linked Data generation using CONSTRUCT queries. Tarql is a command-line tool, written in Java and based on Apache ARQ.

RDF123 RDF123[4] [32] is a tool for translating data residing in a spreadsheet to Linked Data. It relies on its own customely-defined grammar that determines how RDF terms and triples should be generated. The RDF123 consists of two components: (i) the RDF123 application, which allows users to create and edit RDF123 mapping rules, and (ii) the Web service, which generates RDF documents from online spreadsheets relying on RDF123 mapping rules specified in the service or the spreadsheet itself.

XLWrap XLWrap[5] [40] is another tool for translating tabular data to Linked Data. More precisely, it supports Microsoft Excel and OpenDocument spreadsheets, as well as comma- (and tab-) separated value (CSV) and can load data from local files or download remote files via HTTP. More, XLWrap supports cross tables and tables where data is not aligned in rows. Those two are its fundamental difference compared to RDF123. The mapping rules are defined on its own syntax[6], which is in the TriG [18] serialization of RDF, and features a full expression algebra based on the syntax of OpenOffice Calc[7]. XLWrap can be used (i) in-process via the Jena API, or (ii) via the bundled XLWrap-server, which provides a SPARQL endpoint (by integrating Joseki[8], which is nowadays replaced by Apache Jena Fuseki[9]), as well as a Linked Data browser. XLWrap-server is triggered by either a mapping document (*mapping-driven approach* for execution a Linked Data generation process [3]) or a spreadsheet is modified or added (*data-driven approach* [3]) and supports both *query-* and *data-driven* approaches for Linked Data generation [35], as RDF123 does too.

3.2.1.2 Hierarchical data

A variety of solutions exist to generate Linked Data from data in XML format. In principle, they repurpose existing formalizations by rendering them suitable for generating Linked Data from data in XML format. Below we mention a few of the most well-known:

[3] Tarql, `https://github.com/cygri/tarql`
[4] RDF123, `http://rdf123.umbc.edu/`
[5] XLWrap, `http://xlwrap.sourceforge.net/`
[6] XLWrap vocabulary, `http://xlwrap.sourceforge.net/vocab/xlwrap#`
[7] XLWrap, `https://www.openoffice.org/product/calc.html`
[8] Joseki, `https://sourceforge.net/projects/joseki/`
[9] Apache Jena Fuseki, `https://jena.apache.org/documentation/fuseki2/`

XML2RDF The XML2RDF tool[10] [31] allows translating data in XML format to Linked Data. The semantic annotations are explicited by the XSD to OWL mapping rules of the involved XSDs using the XSD2OWL tool[11]. The XSD2OWL is based on an XSL script that performs a partial mapping from XML schema to OWL.

Krextor Krextor [39] is an extensible XSLT-based framework for generating Linked Data from data in XML format. It supports multiple input languages and output RDF notations. Krextor was originally dedicated to OMDoc (Open Mathematical Documents), but it was turned into a generic XSLT-based framework which allows to define translations, the so-called extraction modules. The Linked Data is generated in the RXR notation (Regular XMLRDF)[12], RDF/XML [30] and Turtle [14].

XSPARQL XSPARQL [16] relies on XQuery. It performs dynamic query translation to convert different sources to Linked Data. XSPARQL provides a *query-driven approach* that combines XQuery [17] and SPARQL [34, 47]. This way, it allows to query data in XML and RDF, using the same framework, and transform data from one format into the other, namely XSPARQL supports both RDF generation from XML (*lifting*), and XML from RDF (*lowering*).

X3ML Engine X3ML engine relies on X3ML language, its own custom language for Linked Data generation from data in XML format. It is designed according to the following principles: (i) simplicity, (ii) transparency with respect to its core design and documentation, (iii) collaborative mapping memory with respect to mapping rules definition, (iv) separation between schema mapping and URI generation, so different expertise can be applied, (v) reuse of standards and technologies, and (vi) facilitating instance matching.

3.2.1.3 Various data

In general, most existing tools deploy Linked Data generation from a certain source format to RDF (*source-centric approaches*). Few tools generate Linked Data from *various different* source formats; and those tools actually employ separate source-centric approaches for each of the formats they support. Below, we mention a few of the most well known.

Datalift Datalift [48] generates Linked Data from raw data (CSV, RDF, XML, GML and Shape files) following certain rules depending on the submitted file format. Firstly, the data are mapped to raw RDF without taking into account vocabularies, namespaces or links to existing Linked Data sets (*direct mapping*). Then they are converted to "well-formed" RDF and enriched with the selected ontologies using SPARQL construct queries. Implementation wise, the different file formats are handled independently by Datalift. Different processing occurs for different file types. In the case of CSV and databases, each line

[10] XML2RDF, `http://rhizomik.net/html/redefer/xml2rdf/`
[11] XSD2OWL, `http://rhizomik.net/html/redefer/xsd2owl/`
[12] RXR, `https://www.dajobe.org/papers/xmleurope2004/`

becomes a resource and each column an RDF property, as the W3C direct mapping, to automatically generate Linked Data.In the case of XML files, a generic XSLT transformation is performed to produce Linked Data from a wide range of data in XML documents.

Open Refine OpenRefine[13] (formely Google Refine) is a tool for primarily cleaning and transforming data interactively. The latter might also involve transforming data from a format to another. RDF refine extension[14] allows human agents to generate Linked Data from different data structures and formats. In more details, it reconciles against SPARQL endpoints and RDF dumps and searches the Web for related Linked Data sets. Last, through its user interface, it allows users to define how the raw data is modeled as Linked Data by importing their own vocabularies or reusing existing ones. However, the rules that human agents define can only be exported in a custom JSON format and their execution is only performed by Open Refine via its own custom scripts. Moreover, on modeling level, the human agents are not allowed to use more than one raw data source to describe the knowledge and a uniform Linked Data set cannot be generated in the first place.

Virtuoso Sponger The Virtuoso Sponger[15] is Virtuoso's middleware for generating Linked Data from a variety of data and formats[16]. To be more detailed, it is transparently integrated in Virtuoso's SPARQL query processor. Its core functionality is provided by Cartridges. Each cartridge includes Data Extractors to extract data from one or more data sources, and Ontology Mappers to annotate the extracted data to a certain schema to generate Linked Data. Although, similarly as Open Refine, Virtuoso Sponger relies on custom scripts for each different format to generate the corresponding Linked Data. For instance, it supports RSS or iCalendar, but if a certain data source includes data in another type of Web feed or file format for calendars, a dedicated Cartridge is needed.

Karma Karma is another tool that enables human agents to integrate data from a variety of data formats, such as data in databases, spreadsheets, XML, JSON, and KML. Human agents can semantically annotate their data according to a semantic schema of their choice, relying on Karma's user interface that automates much of the process. Karma turns all different data formats into a tabular structure, following the rely Nested Relational Model (NRM) [44] as an intermediate form to represent data. This way, Karma avoids including references to format-specific data extracts. Nevertheless, its mapping rules can only be used by it, as the data's internal representation relies on how they transform the original data to the Nested Relational Model.

In Table 3.1, we compare the different implementations with the RMLMapper.

[13] Open Refine, `http://openrefine.org/`

[14] RDF Refine, `http://refine.deri.ie/`

[15] Virtuoso Sponger, `http://virtuoso.openlinksw.com/dataspace/doc/dav/wiki/Main/VirtSponger`

[16] Virtuoso's supported formats, `http://vos.openlinksw.com/owiki/wiki/VOS/VirtSpongerCartridgeSupportedDataSourcesNonRDF`

Table 3.1: *Mapping rules definition and supported data formats per implementation.*

	rules	DB	CSV	JSON	XML	GML	KML	shapes
Datalift	embedded in the implementation (direct mapping)	✓	✓		✓	✓		✓
Open Refine	custom		✓	✓	✓			
Karma	R2RML (all formats as tabular)	✓	✓	✓	✓		✓	✓
RMLMapper	RML	✓	✓	✓	✓	✓	✓	✓

3.2.2 R2RML processors

In this section, we outline R2RML processors whose implementation is accompanied, in most cases, by a corresponding scientific paper.

Ultrawrap Ultrawrap [49] is a tool that supports R2RML and implements a query-driven approach for Linked Data generation from data residing in relational databases. Ultrawrap firstly applies direct mapping and then refines the generated RDF graph.

Morph Morph[17] [46] is another tool for Linked Data generation from data residing in relational databases. It supports two approaches: (i) *data upgrade*, which generates Linked Data from a relational database, according to certain R2RML mapping rules; and (ii) *query translation*, which allows evaluating SPARQL queries over a virtual Linked Data set, by rewriting those queries into SQL, according to certain R2RML mapping rules. Morph employs a query translation algorithm from SPARQL to SQL [21] with different optimizations during the query rewriting process, to generate more efficient SQL queries.

Sparqlify Sparqlify[18] [50] is a tool for Linked Data generation that relies on the Sparqlification Mapping Language (SML) to define the mapping rules, but also supports R2RML. Sparqlify blends the traditional SQL `CREATE VIEW` statements with SPARQL `CONSTRUCT` queries, therefore it supports a *query-driven* approach. Its SPARQL-to-SQL translation algorithm relies on a small set of essential operations that cover mapping rules for Linked Data generation from relational databases, for which an Extended Backus-Naur Form (EBNF) of an expression grammar was introduced.

[17] Morph, `http://mayor2.dia.fi.upm.es/oeg-upm/index.php/es/technologies/315-morph-rdb/`

[18] Sparqlify, `http://sparqlify.org/`

R2RML Parser R2RML Parser[19] [38] is a tool that relies on R2RML mapping rules to generate Linked Data from relational databases. R2RML Parser deals in principle with incremental Linked Data generation. In more details, each time a Linked Data generation activity is executed, not all of the input data should be used, but only the one that changed (so-called *incremental transformation*). Such an approach for Linked Data generation can be characterized as asynchronous and not real-time.

DB2Triples DB2Triples[20] is a tool for extracting data from relational databases, semantically annotating the data extracts according to R2RML mapping rules and generating Linked Data. It implements both W3C specifications, i.e., R2RML and Direct Mapping [12]. The first version of the RMLMapper is based on DB2Triples.

Besides data in XML, XSPARQL [16] was also used for direct mapping and as R2RML processor for data in relational databases [43]. Virtuoso RDF views, besides its own mapping language, also supports R2RML.

3.3 Execution Factors

Different key factors determine how an algorithm should be designed for executing mapping rules to generate Linked Data in different use cases. Those factors are related to (and often dependent on) the elements, which are involved in a Linked Data generation activity:

data both *raw data* to generate the desired Linked Data, as well as existing Linked Data.

rules mapping rules that define how Linked Data are generated relying on available data.

tools tools that apply mapping rules to available data and generate Linked Data.

Multiple key factors determine how Linked Data is generated serving different use case's requirements. Therefore, different tools follow alternative execution approaches. The different approaches are outlined below and are determined depending on (i) what the generation purpose is (Section 3.3.1), (ii) its direction (Section 3.3.2), its output (Section 3.3.3), where it occurs (Section 3.3.4), what drives the generation (Section 3.3.5) when an execution occurs (Section 3.3.7), and what the nature of the data structure is (Section 3.3.8).

Use case Since alternative approaches serve different use cases, let us consider a scenario that illustrates each dimension with the help of an example. The use case is about an intelligent transportation search engine, which relies on Linked Data derived from heterogeneous data sources. The search engine obtains information about airports from an airline data source and train stations from a train data source. Information about the location of countries, cities, and addresses is obtained from a data source with spatial data.

[19] R2RML, `http://github.com/nkons/r2rml-parser`
[20] DB2Triples, `https://github.com/antidot/db2triples`

Table 3.2: *Factors affecting Linked Data generation and determining how corresponding algorithms should be designed and tools should be implemented respectively*

factor	**element**		**generation's execution**		
	data	rules	before	during	after
purpose			✓		
direction	✓				✓
materialization					✓
location	✓			✓	
driving force	✓	✓	✓		
trigger			✓		
synchronization			✓		
data structure	✓			✓	

3.3.1 Purpose

Each use case has a different purpose that triggers the Linked Data generation. On a high level, we identify: *production* and *consumption*. The fundamental difference lies on the extend of use cases that the Linked Data generation task aims to cover. The purpose that drives the generation affects the design choices of the execution algorithm. In any case though, it remains independent of the involved elements.

Production Linked Data generation can be driven by a production need, i.e., a *data owner* generates Linked Data to turn the data available as Linked Data. *Production*-driven generation remains independent of the data's potential consumption, but aims to cover any possible use case that might occur. Its consumption should then be adjusted to the Linked Data as it becomes available.

Consumption Linked Data generation can also occur due to a certain consumption need, namely a *data consumer* desires to fulfill an activity that requires processing Linked Data which is required to be generated. Thus, the generated Linked Data is the response to a request which is adjusted to this particular consumption need.

Example In our use case, NMBS, the Belgian train provider, has a legal obligation to publish information about train stations as Linked Data. Different data consumers can profit of this data, which is already *produced*, to build intelligent applications. The Belgian Airlines, Belgium's national airlines, want to identify all airports where its airplanes can fly to. This *consumption* need leads to generating Linked Data specifically for this purpose.

3.3.2 Direction

Linked Data generation might occur following different directions [40]: *source-* or *target-*centric. Which direction is taken is determined before the generation activity is triggered,

and depends mainly on the available data, raw or linked, and on certain occasions on the mapping rules. Different execution algorithms are, thus, designed and implemented that support either the one or the other, or both directions.

Target-centric In this case, the execution is focused on describing a set of views over the data source(s). The approach is same as the Global-As-View (GAV) formalism for data integration [28, 42]. Namely, when mapping among different data models, it is possible to define one of the data models as a view onto the other data model [40]. The target might be (i) a certain *graph pattern* which is derived from existing Linked Data, whose schema is desired to be replicated, or (ii) from a given query (*results-driven* editing approach [35]); or (iii) a given *schema* (a combination of ontologies and vocabularies – *schema-driven* editing approach [35]); or (iv) a set of mapping rules (*model-driven* editing approach [35]). For instance, a *data owner* has a data source, while others are already described as Linked Data. Then he generates his own data source, considering a certain target.

Source-centric In this case, the execution is focused on describing the entities of each data source, independently of other data sources (*data-driven* editing approach [35]). The approach is similar to the Local-as-View (LAV) formalism for data integration [28, 42], as a mapping occurs from the original data source(s) to the mediated schema (Linked Data or schema), making it easier to add and remove data sources. For instance, a *data owner* has two data sources. He defines mapping rules to semantically annotate those data sources, without being concerned about similar or complementary data which is already available as Linked Data.

Example Following our use case, the Belgian Airlines specify a set of SPARQL queries which act as the *target*. The mapping rules are defined for each data source specifically, so the resulting Linked Data matches the SPARQL queries' graph patterns. NMBS specified a set of mapping rules to generate its own Linked Data set from its own available *sources*.

3.3.3 Materialization

In relational databases, views simplify a database's conceptual model with the definition of a virtual relation [10]. A *materialized view* is a database that contains results, while the process of setting up a materialized view is called *materialization* [10]. To achieve this, different materialization strategies exist [33].

In the same context, how the Linked Data generation is materialized differs, affecting the corresponding algorithms. Therefore, the materialization occurs following two alternative approaches [40]: *dumping* or *on-the-fly* (as named by Langegger and Wöß [40]). The fundamental difference lies in when the consumption occurs. On the former case, long term consumption is expected, whereas, on the latter, direct. The materialization, as the purpose of execution, does not depend on the components which are involved in the Linked Data generation, and it has an impact after Linked Data is generated. In more detail:

Dumping A *data dump* is generated into a volatile or persistent triplestore. The scope is to provide a view of the data (similar to a *materialized view* in relational databases).

On-the-fly This occurs when the Linked Data generation process takes place *on-the-fly* (similar to a *non-materialized view*).

Example Following our use case, NMBS *dumps* the train station Linked Data in a triplestore, which is used for storing and retrieving Linked Data, whereas the Belgian Airlines generates the airports Linked Data *on-the-fly* when the query is executed.

3.3.4 Location

The elements involved in Linked Data generation might reside on different sites. The fundamental difference lies in where the data and rules reside, and where the execution takes place. That is determined before the Linked Data generation is initiated and affects how the algorithms are designed. For instance, how the input data is retrieved or processed differs. We identify the following alternatives: local and remote. In more details:

Local Linked Data generation is performed *locally* when it is addressed by the same machine that holds both the generator and the data. For instance, a *data owner* has the data and mapping rules locally stored and in the same place as the generator that executes the mapping rules to generate the desired Linked Data.

Remote Linked Data generation occurs *remotely* when the generator does not reside on the same site as the data and rules. For instance, some data is locally stored, while the generator is a remote service, e.g., Software-as-a-Service (SaaS). Contrariwise, the generator might reside locally, but the data and rules not.

Example Following our use case scenario, the train stations Linked Data set is generated *locally*, as both the Linked Data generator and the data might be on the same machine. To the contrary, in the case of the airports Linked Data set, the data might reside *remotely* from the machine where the Linked Data generator is.

3.3.5 Driving force

The mapping rules to generate Linked Data can be executed using any of the two alternative driving forces [3], namely *mapping* and *data*, or any combination of the two (*hybrid*), and algorithms are affected depending on the element that drives the Linked Data generation. Which approach is followed depends either on the data or rules which are involved in the Linked Data generation. In more details:

Mapping-driven In this case, the processing is driven by the mapping rules. Namely, mapping rules trigger the Linked Data generation and adequate data is employed

to generate the desired Linked Data. For instance, a *data consumer* poses a query which is translated to mapping rules or directly provides the mapping rules based on which the Linked Data generation occurs.

Data-driven In this case, the processing is driven by the data. The data triggers the Linked Data generation and the adequate mapping rules are executed. For instance, a sensor provides new data on frequent time intervals. Once this data reaches a Linked Data generator, a new mapping execution is triggered to generate the corresponding Linked Data based on the associated to this data mapping rules.

Example Once an updated version of the train stations is made available, the data might be sent to a Linked Data generator and trigger a new generation round. Then again the airports dataset generation is triggered by the formation of the mapping rules which specify the corresponding data sources.

3.3.6 Trigger

Linked Data generation can occur *real-time* or *ad-hoc* [38]. While it is independent of the elements which are involved in Linked Data generation, as it occurs with the purpose and materialization, it affects how different algorithms are designed and implemented, e.g., *synchronous* execution requires timely generation. In more details:

Real-time *Synchronous* execution, namely in real-time, is related to the notions of event, i.e., "*any occurrence that results in a change in the sequential flow of program execution*" and response time, i.e., "*the time between the presentation of a set of inputs and the appearance of all the associated outputs*".

On-demand *On demand* execution occurs if an agent triggers the execution that generates Linked Data when desired.

Example The train stations generation occurs *real-time*, as every time the data is updated, a new Linked Data set is generated. If the train stations Linked Data set is not generated every time a new version is available, then its generation is *on-demand*.

3.3.7 Synchronization

While the *driving force* specifies what initiates an execution and the *trigger* when it starts, this factor determines how the generation will be fulfilled: *synchronously* or *asynchronously*.

synchronous A synchronous execution blocks other activities till one is completed.

asynchronous An asynchronous execution is non-blocking, it only initiates an execution.

example NMBS generates different Linked Data sets independently if an *asynchronous* generation is followed for each one, but if generating the one requires to have finished with the previous Linked Data set's generation, then a *synchronous* generation occurs.

3.3.8 Dynamicity

A data source's nature might differ too and that influences how the Linked Data generation is executed. Thus, it affects how an algorithm is designed and a corresponding tool is implemented. For instance, the memory allocation is influenced. This depends on the data, but not on the mapping rules, and affects the generation while it is executed.

Static data A static data structure refers to a data collection that has a certain size. Nevertheless, the original dataset's size can further influence how the Linked Data generation is accomplished. For instance, very big datasets are required to be treated differently than smaller datasets.

Dynamic data A dynamic data structure refers to a data collection that has the flexibility to grow or shrink in size. For instance, it might not be possible to obtain all data, as it could be that the data is infinite in size.

Example The train stations original dataset is *static* –when the Linked Data generation is triggered, the original raw dataset's size is known– whereas the airport's dataset is *dynamic* –its returned size might not be foreseen, as it depends on a query's answers.

All in all, the *purpose*, *driving force*, *trigger*, and *synchronization* affects the Linked Data generation *before* the execution occurs. The *direction* and *materialization* affects the Linked Data generation *after* the execution takes places, whereas the *location* and *data structure* affects the generation during execution. All these should be taken into consideration when designing the corresponding algorithms.

3.4 RML Mapper

RML is an extensible mapping language towards new sources and data structures, allowing different levels of support. On processing level that adds some complexity, as it demands the processor to be scalable to support different input sources in a uniform way.

To deal with this, a rule-based generator is an efficient solution to have a modular architecture where the extraction and mapping are executed independently of each other, in separate modules. When the mapping rules are processed, the mapping module parses and executes them, and thus the final RDF terms and triples are generated. The extraction module deals with the references to the input source which are expressed in the target language's expressions, namely extracting the indicated values from the retrieved data and providing them to the mapping module to generate the desired Linked Data.

The RMLMapper is a rule-based Linked Data generator for multiple data sources accessed using different protocols and containing data in various structures, formats, and serializations. It is an open-source system released under MIT license[21]. The RMLMapper is written in Java and can be used on its own via a command-line interface or its modules can be used in different interfaces, e.g., as a library or remote service. It allows its users to generate Linked Data, as it is defined in mapping rules expressed in RML [3]. To achieve that, the RMLMapper can process a set of mapping rules, parse them, execute them, and generate Linked Data in the end. The generated Linked Data set is represented in RDF and might have different serializations.

The RMLMapper mainly aims to facilitate Linked Data generation when agents desire to make their data available for any kind of consumption (*production*). It follows a *source-centric* direction without being limited though to a single source, while the Linked Data is *dumped* to a certain output. The RMLMapper can support Linked Data generation either locally, i.e., both the mapping rules and data, as well as the processor residing at the same machine, or *remotely* where any of the elements may not reside at the same machine as the rest. It follows a *mapping-driven* approach which is required to be triggered *on-demand* and the procedure is accomplished *synchronously*. As far as the data is concerned, the RMLMapper is adequate to be used with static data. The RMLMapper's source code is publicly available at `https://github.com/RMLio/RML-Mapper`.

3.4.1 Phases

The RMLProcessor was built in three phases:

1st phase (*proof-of-concept*)

The first version of the RMLMapper was a *proof-of-concept* implementation based on DB2triples[22] (details about DB2triples is available at Section 3.2), which proved that RML can generalize R2RML and a corresponding implementation is feasible.

2nd phase (*modular approach*)

The second version distinguishes the RMLMapper's modules. Its modular approach allowed the quick development of other tools, such as the RMLValidator [4] and RMLWorkbench [6], and the RMLMapper's integration in other systems, as it occurred later on with the DBpedia EF [27].

3rd phase (*data-processing*)

The third version enabled data-processing [27]. This allowed the RMLMapper to generate Linked Data relying on data values beyond the ones which are extracted from the data source. This way, the RML language could be used to describe the DBpedia mapping rules and the RMLMapper to be integrated in the DBpedia EF [27].

[21] MIT license, `https://opensource.org/licenses/MIT`
[22] DB2triples, `https://github.com/antidot/db2triples`

3.4.2 Architecture

The RMLMapper's architecture is shown in Figure 3.1. It consists of the *mapping* –red components ①-③ in Figure 3.1– and *extraction module* –green components ④-⑥ in Figure 3.1. The *mapping module* takes care of parsing and extracting the mapping rules, and generating RDF terms and triples. The *extraction module* takes care of retrieving the data which is required to generate the desired RDF terms and triples. More, the RMLMapper consists of the *data transformation module* –blue components ⑦ in Figure 3.1 which takes care of the data transformations. Last, the RMLMapper includes a *metadata module* –orange components ⑧ in Figure 3.1– which takes care of the metadata and provenance.

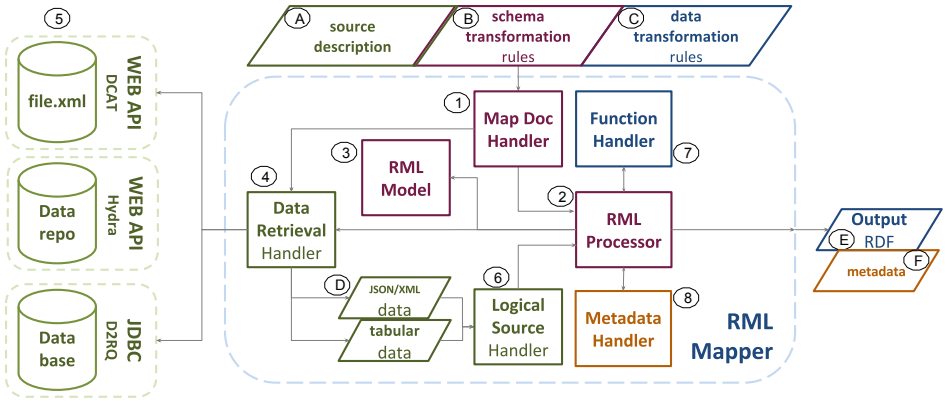


Figure 3.1: *RMLMapper's components and overall architecture*

3.4.3 Modules

The RMLMapper consists of different modules from its second version on. Each module is described below and all modules are summarized in Table 3.3. As in the previous section, we continue the use case scenario to make the role of each module more comprehensive.

Use Case Following the use case which was introduced in Section 3.3, NMBS publishes its data in XML format. The data (Ⓓ in Figure 3.1) is retrieved from its Web API (⑤ in Figure 3.1), abstracted and iterated (*Logical Source Handler* – ⑥ in Figure 3.1), and based on the parsed mapping rules (Ⓐ-Ⓒ in Figure 3.1), the mapping rules are executed (*Mapping Document Handler* – ① in Figure 3.1), and the corresponding Linked Data (Ⓔ in Figure 3.1) is generated (*RML Processor* – ② in Figure 3.1). Additional metadata (Ⓕ in Figure 3.1) can be generated in the process (*Metadata Handler* – ⑧ in Figure 3.1), and even raw data values can be transformed during execution (*Function Handler* – ⑦ in Figure 3.1).

The RMLMapper consists of the following modules:

RML Model This module (③ in Figure 3.1) describes the RML model's internal represen-

Table 3.3: RMLMapper's components

Components	Description
RML Model ③	internal (Java) representation for RML mapping rules
RML Processor ②	core module for RDF terms & triples generation [3]
Map Doc Handler ①	extracts & generates mapping rules and documents [3]
Data Retrieval Handler④	retrieve data based on dataset & service descriptions [5]
Logical Source Handler ⑥	handles references & iterations over the input data [1, 3]
Function Handler ⑦	processor for data transformations [27]
Metadata Handler ⑧	handles metadata & provenance generation [6]

tation. A separate project, the *RML-Model* was introduced as part of the RMLMapper which is available at `https://github.com/RMLio/RML-Model`.

Data Retrieval Handler The RML Data Retrieval Handler (④ in Figure 3.1) deals with the data retrieval from the data sources where the input data originally reside [5]. This module is necessary to abstract how the data retrieval is handled.

The *Data Retrieval Handler* relies on the *Data Source*'s access description (Ⓐ in Figure 3.1 –more details on such data source descriptions are available at Section 5.2.1) to retrieve the *Logical Source* (Ⓓ in Figure 3.1). For instance, in our use case, the data is provided via a WebAPI. The description of this API is required to retrieve the corresponding data. The access description can be provided by the agent who either publishes (*data owner*) or consumes the data (*data consumer*). Both descriptions are equally treated by the RMLMapper, but the one provided by the *data consumer* prevails, when both are available [5].

A separate project, the *RML-DataRetrievalHandler*, which is available at `https://github.com/RMLio/RML-DataRetrieval`, was introduced to deal with data retrieval. Adding a new data access interface just requires additionally incorporating a corresponding extractor. The *RML-DataRetrievalHandler* currently supports the access interfaces described in [5] and Section 5.3.1, i.e., database connectivity, local files, and Web resources. This module is reused in other projects besides the RMLMapper: the RMLEditor [36] and RMLWorkbench [8] to fetch a data sample and assist data owners to define mapping rules and generate the desired Linked Data.

Logical Source Handler The *Logical Source Handler* (⑥ in Figure 3.1) deals with the iteration and data values extraction that is required by the core RMLProcessor (② in Figure 3.1) to generate the desired Linked Data. For instance, in our use case, it is specified that the iteration should occur over the stations elements and this is specified relying on an XPath expression, as this reference formulation is specified. This module is necessary to abstract the original structure and format from the core RMLProcessor and to apply the mapping rules, namely generating the Linked Data disregarding the structure or format of the input data. The current Logical Source Handler supports data in databases, as well as in CSV, JSON, XML, HTML, and wikitext.

The Logical Source Handler transforms any input data into an iterator of a set of data items, whose fields can be retrieved using a reference formulation. It actually returns

a certain iteration or a value for a certain reference of a certain iteration. This module is triggered by the core RMLProcessor as it follows the *mapping-driven* paradigm, but it would trigger the RMLProcessor in the case of the *data-driven* paradigm.

A separate project, the *RML-LogicalSourceHandler* which is available at `https://github.com/RMLio/RML-LogicalSourceHandler`, was introduced as part of the RMLMapper. The latest version supports data in the following structures and formats: tabular data in relational databases which can be referred relying on SQL or in CSV format [3, 5]; hierarchical data in XML or JSON format ([1, 3]); or semi-structured data in HTML format ([2]). Any other data format may be added by including its corresponding library (in this case, Java or Scala is supported).

Mapping Document Handler The *Mapping Document Handler* (① in Figure 3.1) deals with the parsing and processing of a set of mapping rules. It triggers the RMLProcessor (② in Figure 3.1) in the case of the *mapping-driven* paradigm, but it is triggered by the RMLProcessor in the case of the *data-driven* paradigm. The set of mapping rules might be provided either as a document, e.g., a local file or a file published on the Web, as a Linked Data set, e.g., an RDF dump, or RDF triples published and accessed via a SPARQL endpoint or an LDF server [51].

A separate project was introduced in the RMLMapper, the *RML-MapDocHandler* which is available at `https://github.com/RMLio/RML-MapDocHandler`. This module is reused for other projects besides the RMLMapper: the RMLValidator [4] and DBpedia EF [27]. In the former case, it is used to retrieve the mapping rules and provide them to the RMLValidator to assess their quality. In the latter case, it is even used when the mapping rules are transformed from wikitext to RML.

RML Processor (core) The *RML Processor* (② in Figure 3.1) is the core module that orchestrates all other modules to generate the desired Linked Data. The *RML Processor* is available at `https://github.com/RMLio/RML-Processor`.

Metadata Handler The *Metadata Handler* (⑧ in Figure 3.1) deals with the Linked Data metadata generation [6]. This module is required to keep track of the origin of the different data which is used to generate the desired Linked Data and provides metadata (Ⓕ in Figure 3.1) regarding the generated Linked Data set (Ⓔ in Figure 3.1).

The RMLMapper currently supports the VOID vocabulary [11] for the Linked Data set's metadata and the W3C recommended PROV-O [41] for its provenance. Another separate project was introduced in the RMLMapper, the *RML-MetadataHandler*, which is available at `https://github.com/RMLio/RML-MetadataHandler`, to deal with metadata and provenance generation.

Function Handler The *Function Handler* (⑦ in Figure 3.1) connects the RMLProcessor with the independent *Function Processor*. The latter uses function declarations described in FnO [25] to retrieve and execute their corresponding implementations. Whenever the RMLProcessor encounters a data transformation, the *Function Handler* is used to extract the function identifier (i.e., its URI) and parameter values, and provide them to the *Function Processor*. The *Function Processor* discovers the relevant implementations online relying on the function's identifier [26], and obtains an implementation to be executed locally. Based on the function's description, the *Function Processor* automatically detects how to execute the function

and returns the value to the RMLProcessor. The *Function Handler* is available at `https://github.com/FnOio/function-processor-java`.

3.4.4 Workflow

The RMLMapper can serve both *consumption-* and *production-driven* demands for Linked Data generation. It is *source-centric*, but it can support multiple data sources, follows the *mapping-driven* approach and materializes the generation by *dumping* the generated RDF triples to a file. The RMLMapper can be used both for *local* and *remote* processing. For instance, when Linked Data is generated using the RMLEditor, the RMLMapper is accessed from a remote server [37]. It is designed for *asynchronous* Linked Data generation and the current implementation expects static data.

Algorithm 1 Algorithm for Linked Data generation

```
mapping rules
data sources
for all tm in TMs do
    ds = retrieveDataSource(tm)
    for all r in ds do
        s = generateSubject(sm,r)
        POMs = retrievePOMs(tm)
        while pom in POMs do
            PMs = retrievePMs(pom)
            OMs = retrieveOMs(pom)
            while pm in PMs do
                p = generatePredicate(pm,r)
                while om in OMs do
                    o = generateObject(om,r)
                    generateTriple(s,p,o)
                end while
            end while
        end while
    end for
end for
```

The algorithm RMLMapper workflow functions as follows:

Mapping rules processing A set of RML mapping rules (① in Figure 3.1) are provided to the RMLProcessor (②). The RMLProcessor triggers the *mapping module* – components in red colour in Figure 3.1 – to parse and extract the mapping rules. The *Mapping Document Handler* processes the provided mapping rules and represents them following the internal representation, as defined by the *RML Model* component (③).

Input data retrieval Once the mapping rules are extracted and parsed, the RMLProcessor identifies the different *Triples Maps* and, more precisely, their *Data Sources*. Then, the RMLProcessor processes each *Triples Map* consecutively. For each Triples Map, it firstly requests an extract of data from the *extraction module* – components in green colour in Figure 3.1. The *Data Retrieval Handler* (④ in Figure 3.1) processes the provided data access description and triggers the corresponding concrete factory (⑤). The current *Data Retrieval Handler* supports Web resources, databases, and local files. If another data access interface is desired, a corresponding concrete factory can be easily added.

Extracted data processing Once the data is retrieved (⑥ in Figure 3.1), the *Data Retrieval Handler* passes the extracted data to the *Logical Source Handler* (⑦). The latter, considering the specified *Logical Source* and based on the defined *Reference Formulation*, delegates the data to a reference-specific *sub-extractor*. For each *Triples Map*, its delegated sub-extractor iterates over the data as the *Triples Map*'s *Iterator* specifies.

Iteration processing For each iteration, the *extraction module* (*Logical Source Handler*) returns a data extract. The *mapping module* of the RMLProcessor applies mapping rules to data extracts of each iteration. The defined *Subject* and *Predicate-Object Maps* are applied and RDF triples are generated. The execution of dependent *Triples Maps*, due to joins, is triggered by the *Parent Triples Map* and a *nested* mapping process occurs.

Functions processing If the mapping rules contain data transformations, namely statements in FnO, instead of barely relying on the *mapping module*, the RMLProcessor requests the *Function Handler* (⑦ in Figure 3.1) to take care of the data transformation. The *Function Handler* calls the *Function Processor*, and the latter is responsible to discover corresponding implementations for the functions which are mentioned. The *Function Processor* executes the function based on the data values which are passed as parameters and returns the result values.

Metadata Handler The RMLMapper may also generate metadata about the generated RDF terms and triples, and provenance statements. Depending on the desired level of details, the RMLProcessor triggers the *Metadata Handler* (⑧ in Figure 3.1) to keep track – components in orange colour in Figure 3.1. For instance, if detailed provenance information is desired, e.g., on RDF triple's level, after the data is retrieved (using the *Data Retrieval Handler*), the RMLProcessor triggers the *Metadata Handler* to track its provenance.

Output The Linked Data which is generated by the RMLMapper may be represented in any of the RDF's recommended serializations. To be more precise, the RMLProcessor generates RDF triples in any of the following serializations: Turtle [14], NTriples [20], NQuads [19], RDF/XML [30], RDF-JSON [24], and JSON-LD [22].

Note that we deliberately ignore storing files into memory, which would solve the multiple passes for the mapping-driven approach. We only consider a *streaming* solution, since

RML can be used to process datasets too big for the processor' s memory. We accept a longer mapping time in trade of lower memory usage. A side-effect of a streaming approach, is the inability to support some features of expression languages. For instance, XPath has look-ahead functionality that requires access to data which is not yet known. Thus, we can only support a subset. Nevertheless, in practice, most of the expressions only require functionality within this subset.

3.4.5 Graphical Interfaces

Two Graphical User Interfaces (GUI) were built on top of the RMLMapper to facilitate users with no technical background to use the RMLMapper: the RMLEditor [36] for editing mapping rules (see Section 2.6) and RMLWorkbench [7] for administrating data sources, mapping rules and Linked Data generation in general (see Section 5.6).

3.5 Evaluation

In this section, we firstly define the number of RDF terms and triples which are generated from a certain mapping rule, iteration, and in general from a data source, independently of the data source structure, format, and serialization (Section 3.5.1). Then, we use both synthetic and real data to showcase that the RMLMapper (i) does generate RDF terms and triples from heterogeneous data sources with respect to structure and format, (ii) in reasonable time, of the same magnitude as a custom generator for the same data source, and (iii) without being affected by the data's structure and format (Section 3.5.2).

3.5.1 Formal

In this subsection, we provide a formal description of Linked Data generation and estimate the maximum number of generated RDF terms and triples if the data of the input data source is complete, namely there are no missing values, and the mapping process occurs without any failure, namely no mapping rule fails to generate an RDF term or triple.

number of RDF terms per mapping rule If $\mathcal{R}$ is the set of references to the input source, and $\mathcal{V}_t(r)$ the values which are returned for reference $r \in \mathcal{R}$ and for iteration $i \in \mathcal{I}$, where $\mathcal{I}$ is a set of iterations and $|\mathcal{I}|$ is the number of iterations, i.e., the number of elements in $\mathcal{I}$ is equal to the number of iterations. Then the maximum total number of RDF terms generated from this mapping rule is: $\prod_{r \in \mathcal{R}} \mathcal{V}_i(r)$. This means that the maximum number of RDF terms which are generated from a single mapping rule grows *geometrically* in function of both the number of references and returned values. For instance, assuming a mapping rule with 2 references to the input, where up to 2 values can be returned for the first reference and up to 3 values for the second, generates 6 RDF terms per iteration. Applied to a data source with 1,000 records, this could generate up to 6,000 RDF terms.

number of RDF triples per iteration The maximum total number of RDF triples generated per iteration is $\mathcal{S}_{sm} \cdot \mathcal{P}_{pm} \cdot \mathcal{O}_{om}$ where $S_{sm} = \prod_{r \in \mathcal{R}}^{sm} \mathcal{V}_i(r)$ is the maximum number of RDF terms which are generated from a mapping rule $sm \in \mathcal{M}$, where $\mathcal{M}$ is the set of mapping rules. In the same context, $\mathcal{O}_{om} = \prod_{r \in \mathcal{R}}^{om} \mathcal{V}_i(r)$ is the maximum number of RDF terms which are generated from a mapping rule $om \in \mathcal{M}$ that defines how the RDF triple's object is generated, and $\mathcal{P}_{om} = \prod_{r \in \mathcal{R}}^{pm} \mathcal{V}_t(r)$. For instance, if the aforementioned example defines the maximum number of objects, $\mathcal{O}_{om}$, assuming that $\mathcal{S}_{sm}$ and $\mathcal{P}_{sm}$ are equal to one and two RDF term(s) respectively, then the maximum total number of RDF triples which are generated for the mapping rule that defines how the RDF triple is generated for a certain iteration is equal to twelve (12) RDF triples.

number of RDF triples per data source The total number of RDF triples which are generated from a data source, data_α is $|\mathcal{I}| \cdot \mathcal{S}_{sm} \cdot \mathcal{P}_{pm} \cdot \mathcal{O}_{om}$. This means that the maximum number of RDF triples which are generated from a data source grows *linearly* in function of the number of iterations. For instance, for the aforementioned data source with the one-thousand records, the maximum number of RDF triples which are generated is equal to eighteen thousand (18,000) RDF triples, only for this mapping rule. The maximum total number of RDF triples which are generated is equal to the sum of all RDF triples which are generated for each mapping rule applied to the data for each iteration, namely $\sum_{\{sm,pm,om\} \subset \mathcal{M}} |\mathcal{I}| \cdot \mathcal{S}_{mp} \cdot \mathcal{P}_{pm} \cdot \mathcal{O}_{om}, \forall \{sm, pm, om\} \subset \mathcal{M}$.

In a single formula, the total number of generated RDF triples is equal to:

$$\sum_{\{sm,pm,om\} \subset \mathcal{M}} |\mathcal{I}| \cdot \prod_{r \in \mathcal{R}}^{sm} \mathcal{V}_t(r) \cdot \prod_{r \in \mathcal{R}}^{pm} \mathcal{V}_t(r) \cdot \prod_{r \in \mathcal{R}}^{om} \mathcal{V}_t(r)$$

3.5.2 Experimental

We performed two evaluations, using (i) synthetic artificial data and (ii) a real data source (DBpedia). First, we show that our approach supports multiple data structures and formats (Section 3.5.2.1), and we compare performance for data of different sizes and depth (in the case of hierarchical data structures). Next, we compare the RMLMapper with the custom generator for generating the DBpedia Linked Data set (Section 3.5.2.2).

3.5.2.1 Synthetic data

data For the performance evaluations we used the following set-up: A data generator-which is able to generate data in different structures and formats. For the artificial part, we generated data sources whose size differs with respect to the number of records. To be more precise, the Linked Data generation was assessed for datasets ranging from ten to a hundred thousand triples, in logarithmic steps of base ten. In the end, 200 data sources with sizes ranging from ten to one hundred thousand records were generated.

set-up The machine, where we executed the evaluations, consisted of 24 cores (Intel Xeon CPU E5-2620 v3 @ 2.40GHz) and 128GB RAM. All evaluations were performed using docker images, and the different tests were orchestrated using custom scripts. All timings include docker images' initialization time.

Table 3.4: *Execution time in seconds for Linked Data generation from data sources in different formats and whose number of records ranges between 10 and 100,000 records*

data source size	CSV	JSON	XML
10	11	11	12
100	14	14	16
1,000	24	24	32
10,000	160	158	221
100,000	1,493	1,478	1,977

functionality To show that the RMLMapper functionally support different data structures and formats, data in CSV, XML, and JSON format and Linked Data were generated in all cases, as shown in Figure 3.2. Given the RMLMapper was originally developed as an extension of DB2triples, we consider that data residing in databases is supported by default.

time From the same figure, it is also shown that the Linked Data generation takes approximately the same time, independently of the data format. XML seems to take a bit more time, but this is mainly due to the library used in the RMLMapper's implementation. Execution time was averaged over all data sources of the same size, as no significant difference in execution time was found. The RMLMapper generates Linked Data without significant delays when the original data remain small in size, whereas its execution time increases exponentially when the data's size increases. This occurs for all types of data formats as shown in Figure 3.2. The median execution per data source size time can be seen in Table 3.4.

Moreover, the time which is required to generate some Linked Data depends also on the depth of the tree in the case of hierarchical structured data. We indicatively show in Figure 3.3 the execution time for different sizes of a data source in JSON format.

3.5.2.2 Real case: DBpedia

The RMLMapper was integrated in the DBpedia EF [45] and it was compared with the existing DBpedia EF with respect to *coverage*, and *performance*.

Coverage We generated Linked Data, both with the original Mapping Extractor of the DBpedia EF and the RMLMapper, for 16,244,162 pages of the English Wikipedia (ver-

Table 3.5: *Execution time in seconds for Linked Data generation from data sources in different formats and whose number of records ranges between 10 and 100,000 records*

data source size	Level 1	Level 3	Level 5
10	11	6	9
100	14	8	18
1,000	24	31	61
10,000	160	200	504
100,000	1,493	1,908	5,260

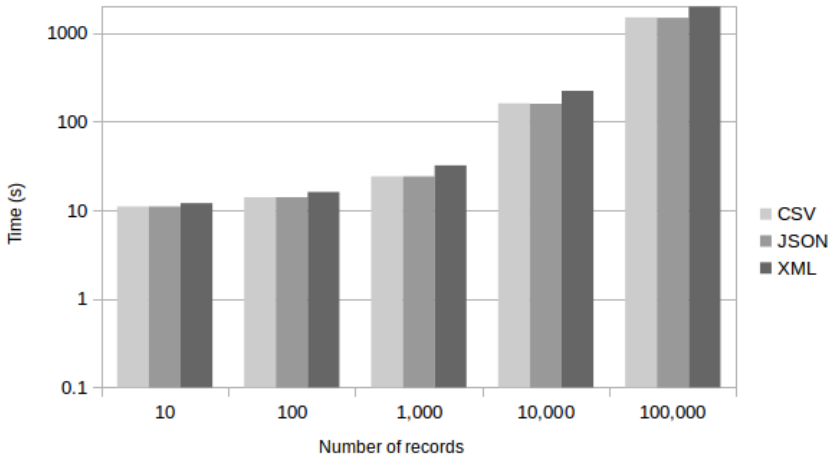


Figure 3.2: *Linked Data generation from data in* CSV*,* JSON *and* XML *format (logarithmic seconds)*

sion 20170501[23]). The former extraction yielded 62,474,123 RDF triples in total, the latter 52,887,156. In terms of entities, 4,747,597 are extracted with the DBpedia EF's generator, whereas 4,683,709 entities are extracted with RMLMapper, offering 98% coverage.

The Linked Data sets differ because certain mapping rules were *deliberately* omitted, due to the following reasons: (i) *unsustainable URIs*; the DBpedia EF itself generates intermediate entities whose URI is generated relying on an iterator, e.g., `http://dbpedia.org/resource/Po_(river)__mouthPosition__1`. Assigning URIs this way does not generate sustainable identifiers. (ii) *custom DBpedia datatypes*; the DBpedia EF generates additional RDF triples with custom DBpedia datatypes for units of properties. The mapping rules for (i) and (ii) were temporarily omitted, until the DBpedia community decides on how they should be modeled. (iii) *RDF triples added by the DBpedia EF;* These are RDF triples generated from a value of an infobox, if the article's text contains a URI with this value.

[23] `https://dumps.wikimedia.org/enwiki/20170501/enwiki-20170501-pages-articles.xml.bz2`

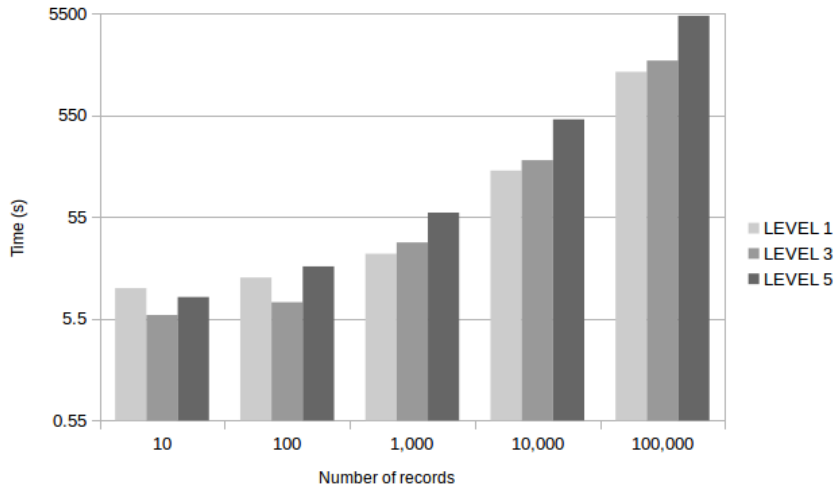


Figure 3.3: *Linked Data generation from data in* JSON *format with diverse depth (from 1 to 5 levels – logarithmic seconds)*

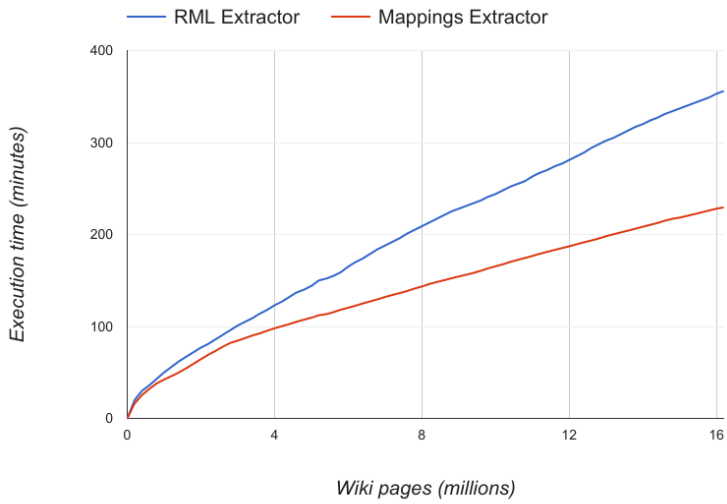


Figure 3.4: *Performance comparison: Mappings Extractor vs RML Extractor*

Performance The RMLMapper generates the DBpedia Linked Data set on reasonable time. The RMLMapper is still on average 0.46ms slower per page than the original framework. Namely, the RMLMapper requires 1.31ms/page, compared to the original DBpedia EF which requires 0.85ms/page. On a larger scale the RMLMapper generates Linked Data from 16,244,162 pages in 5 hours and 56 minutes, compared to the current DBpedia EF

which requires 3 hours and 50 minutes (35% slower than the DBpedia EF, Figure 3.4).

Even though the RMLMapper is slower compared to the DBpedia EF, it still performs fast enough, as it remains in the same order of magnitude compared to the corresponding solution which is designed and implemented exactly for this purpose. Performance is put aside for this work in general, as we opted for more sustainable Linked Data generation of high quality. Achieving sustainability is much more important than speed which may be optimized in the future.

Moreover, statistics from both LODLaundromat [15] and LODStats [29], that provide a comprehensive picture of the current state of the LOD cloud, show that most Linked Data sets are small in size. For instance, in the case of LODLaundromat, 83% of all Linked Data sets contain fewer than one million triples, and 60% contains at most 100,000 triples. This means that for a data source of 100,000 records, a single RDF triple is generated per record, or for a data source of 10,000 records, 10 RDF triples are generated per record. Thus, the RMLMapper can be used for efficiently generating more than half of the Linked Data sets which are currently published on the LOD cloud, but not only as our evaluation against the DBpedia Linked Data set showed.

Conclusions

In this chapter, we outline factors that determine how the mapping rules execution should occur, present our approach for executing Linked Data quality for (semi-)structured data, originally stemming in heterogeneous data sources, and we detail on our implementation.

The distinction between mapping rules declaration and execution allows the implementation of tools dedicated and optimized for each task separately. This, essentially, allows agents to define once the mapping rules which declare how Linked Data is generated and execute those mapping rules multiple times, periodically and over different data sources.

Our implementation, the RMLMapper, was originally designed and implemented for Open Data. Thus it is more adequate for data sources of small and medium size. However, even for larger data sources, such as Wikipedia, it performs in the same order of magnitude as a custom implementation which is developed deliberately for this data source.

The evaluation shows that our methodology is applicable for (i) heterogeneous data sources with data in different formats, (ii) data sources with diverse sizes and hierarchical data structures of different depths, and (iii) large Linked Data sets, such as DBpedia, with its execution time being in the same order of magnitude as for a custom implementation deliberately designed and optimized for this data source.

References

[1] **Anastasia Dimou**, Miel Vander Sande, Jason Slepicka, Pedro Szekely, Erik Mannens, Craig Knoblock, and Rik Van de Walle. Mapping Hierarchical Sources into

RDF using the RML Mapping Language. In *Proceedings of the 2014 IEEE International Conference on Semantic Computing*, ICSC '14, pages 151–158, Washington, DC, USA, 2014. IEEE Computer Society. ISBN 978-1-4799-4003-5. doi: 10.1109/ICSC.2014.25. URL http://dx.doi.org/10.1109/ICSC.2014.25.

[2] **Anastasia Dimou**, Miel Vander Sande, Pieter Colpaert, Laurens De Vocht, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. Extraction and Semantic Annotation of Workshop Proceedings in HTML using RML. In Valentina Presutti, Milan Stankovic, Erik Cambria, Iván Cantador, Angelo Di Iorio, Tommaso Di Noia, Christoph Lange, Diego Reforgiato Recupero, and Anna Tordai, editors, *Semantic Web Evaluation Challenge: SemWebEval 2014 at ESWC 2014, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers*, pages 114–119, Cham, 2014. Springer International Publishing. URL http://dx.doi.org/10.1007/978-3-319-12024-9_15.

[3] **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. URL http://ceur-ws.org/Vol-1184/ldow2014_paper_01.pdf.

[4] **Anastasia Dimou**, Dimitris Kontokostas, Markus Freudenberg, Ruben Verborgh, Jens Lehmann, Erik Mannens, Sebastian Hellmann, and Rik Van de Walle. Assessing and Refining Mappings to RDF to Improve Dataset Quality. In Marcelo Arenas, Oscar Corcho, Elena Simperl, Markus Strohmaier, Mathieu d'Aquin, Kavitha Srinivas, Paul Groth, Michel Dumontier, Jeff Heflin, Krishnaprasad Thirunarayan, and Steffen Staab, editors, *The Semantic Web – ISWC 2015*, volume 9367 of *Lecture Notes in Computer Science*, pages 133–149. Springer International Publishing, Oct 2015. doi: 10.1007/978-3-319-25010-6_8. URL http://link.springer.com/chapter/10.1007/978-3-319-25010-6_8.

[5] **Anastasia Dimou**, Ruben Verborgh, Miel Vander Sande, Erik Mannens, and Rik Van de Walle. Machine-interpretable Dataset and Service Descriptions for Heterogeneous Data Access and Retrieval. In *Proceedings of the 11th International Conference on Semantic Systems*, pages 145–152, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3462-4. doi: 10.1145/2814864.2814873. URL http://doi.acm.org/10.1145/2814864.2814873.

[6] **Anastasia Dimou**, Tom De Nies, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. Automated Metadata Generation for Linked Data Generation and Publishing Workflows. In Sören Auer, Tim Berners-Lee, Christian Bizer, and Tom Heath, editors, *Proceedings of the 9th Workshop on Linked Data on the Web*, volume 1593 of *CEUR Workshop Proceedings*, April 2016.

[7] **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 Takahiro Kawamura and Heiko Paulheim, editors, *Proceedings of the 15th International Semantic Web Conference: Posters and Demos*, volume 1690 of *CEUR Workshop Proceedings*, October 2016. URL http://ceur-ws.org/Vol-1690/paper98.pdf.

[8] **Anastasia Dimou**, Dimitris Kontokostas, Markus Freudenberg, Ruben Verborgh, Jens Lehmann, Erik Mannens, and Sebastian Hellmann. DBpedia Mappings Quality

Assessment. In Takahiro Kawamura and Heiko Paulheim, editors, *Proceedings of the 15th International Semantic Web Conference: Posters and Demos*, volume 1690 of *CEUR Workshop Proceedings*, October 2016. URL http://ceur-ws.org/Vol-1690/paper97.pdf.

[9] **Anastasia Dimou**, Ben De Meester, Pieter Heyvaert, Ruben Verborgh, Steven Latré, and Erik Mannens. RML Mapper: a tool for uniform Linked Data generation from heterogeneous data. *Semantic Web Journal*, 2017. (under review).

[10] Daniel J Abadi, Daniel S Myers, David J DeWitt, and Samuel R Madden. Materialization strategies in a column-oriented DBMS. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pages 466–475. IEEE, 2007.

[11] Keith Alexander, Richard Cyganiak, Michael Hausenblas, and Jun Zhao. Describing Linked Datasets with the VoID Vocabulary. Interest Group Note, W3C, March 2011. http://www.w3.org/TR/void/.

[12] Marcelo Arenas, Alexandre Bertails, Erik Prud'hommeaux, and Juan Sequeda. A Direct Mapping of Relational Data to RDF. W3C Recommendation, W3C, September 2012. https://www.w3.org/TR/rdb-direct-mapping/.

[13] Sören Auer, Sebastian Dietzold, Jens Lehmann, Sebastian Hellmann, and David Aumueller. Triplify: Light-weight Linked Data Publication from Relational Databases. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, pages 621–630, New York, NY, USA, 2009. ACM. doi: 10.1145/1526709.1526793. URL http://doi.acm.org/10.1145/1526709.1526793.

[14] David Beckett, Tim Berners-Lee, Eric Prud'hommeaux, and Gavin Carothers. RDF 1.1 Turtle. W3C Recommendation, W3C, February 2014. URL http://www.w3.org/TR/turtle/.

[15] Wouter Beek, Laurens Rietveld, Hamid R. Bazoobandi, Jan Wielemaker, and Stefan Schlobach. LOD Laundromat: A Uniform Way of Publishing Other People's Dirty Data. In Peter Mika, Tania Tudorache, Abraham Bernstein, Chris Welty, Craig Knoblock, Denny Vrandečić, Paul Groth, Natasha Noy, Krzysztof Janowicz, and Carole Goble, editors, *The Semantic Web – ISWC 2014: 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I*, pages 213–228, Cham, 2014. Springer International Publishing. ISBN 978-3-319-11964-9. doi: 10.1007/978-3-319-11964-9_14. URL https://doi.org/10.1007/978-3-319-11964-9_14.

[16] Stefan Bischof, Stefan Decker, Thomas Krennwallner, Nuno Lopes, and Axel Polleres. Mapping between rdf and xml with xsparql. *Journal on Data Semantics*, 1(3):147–185, 2012. ISSN 1861-2040. doi: 10.1007/s13740-012-0008-7. URL http://dx.doi.org/10.1007/s13740-012-0008-7.

[17] Scott Boag, Don Chamberlin, Mary F Fernández, Daniela Florescu, Jonathan Robie, Jérôme Siméon, and Mugur Stefanescu. XQuery 1.0: An XML query language. W3C Recommendation, W3C, 2002. https://www.w3.org/TR/xquery/.

[18] Gavin Carothers and Andy Seaborne. RDF 1.1 TriG. W3C Recommendation, W3C, February 2014. URL http://www.w3.org/TR/trig/.

[19] Gavy Carothers. RDF 1.1 N-Quads. W3C Recommendation, W3C, February 2014. http://www.w3.org/TR/n-quads/.

[20] Gavy Carothers and Andy Seaborne. RDF 1.1 N-Triples. W3C Recommendation, W3C, February 2014. http://www.w3.org/TR/n-triples/.

[21] Artem Chebotko, Shiyong Lu, and Farshad Fotouhi. Semantics preserving SPARQL-to-SQL translation. *Data and Knowledge Engineering*, 68(10):973 – 1000, 2009. ISSN 0169-023X. doi: http://dx.doi.org/10.1016/j.datak.2009.04.001. URL http://www.sciencedirect.com/science/article/pii/S0169023X09000469.

[22] Dan Connolly. JSON-LD 1.0. W3C Recommendation, W3C, January 2014. http://www.w3.org/TR/json-ld/.

[23] Richard Cyganiak. Tarql – SPARQL for Tables: Turn CSV into RDF using SPARQL syntax. Technical report, W3C, January 2015. http://tarql.github.io/.

[24] Ian Davis, Thomas Steiner, and Arnaud Le Hors. RDF 1.1 JSON Alternate Serialization (RDF/JSON). W3C Working Group Note, W3C, November 2013. http://www.w3.org/TR/rdf-json/.

[25] Ben De Meester and **Anastasia Dimou**. The Function Ontology. Unofficial Draft, imec, October 2016. http://users.ugent.be/~bjdmeest/function/.

[26] Ben De Meester, **Anastasia Dimou**, Ruben Verborgh, and Erik Mannens. Discovering and Using Functions via Content Negotiation. In Takahiro Kawamura and Heiko Paulheim, editors, *Proceedings of the 15th International Semantic Web Conference: Posters and Demos*, volume 1690 of *CEUR Workshop Proceedings*, October 2016. URL http://ceur-ws.org/Vol-1690/paper110.pdf.

[27] Ben De Meester, Wouter Maroy, **Anastasia Dimou**, Ruben Verborgh, and Erik Mannens. Declarative Data Transformations for Linked Data Generation: The Case of DBpedia. In Eva Blomqvist, Diana Maynard, Aldo Gangemi, Rinke Hoekstra, Pascal Hitzler, and Olaf Hartig, editors, *The Semantic Web: 14th International Conference, ESWC 2017, Portorož, Slovenia, May 28 – June 1, 2017, Proceedings, Part II*, pages 33–48, Cham, 2017. Springer International Publishing. ISBN 978-3-319-58451-5. doi: 10.1007/978-3-319-58451-5_3. URL http://dx.doi.org/10.1007/978-3-319-58451-5_3.

[28] AnHai Doan, Alon Halevy, and Zachary Ives. *Principles of Data Integration*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edition, 2012. ISBN 0124160441, 9780124160446.

[29] Ivan Ermilov, Jens Lehmann, Michael Martin, and Sören Auer. LODStats: The Data Web Census Dataset. In Paul Groth, Elena Simperl, Alasdair Gray, Marta Sabou, Markus Krötzsch, Freddy Lecue, Fabian Flöck, and Yolanda Gil, editors, *The Semantic Web – ISWC 2016: 15th International Semantic Web Conference, Kobe, Japan, October 17–21, 2016, Proceedings, Part II*, pages 38–46, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46547-0. doi: 10.1007/978-3-319-46547-0_5. URL https://doi.org/10.1007/978-3-319-46547-0_5.

[30] Fabien Gandon and Guus Schreiber. RDF 1.1 XML Syntax. W3C Recommendation, W3C, February 2014. URL http://www.w3.org/TR/rdf-syntax-grammar/.

[31] Roberto García. *A Semantic Web Approach to Digital Rights Management*. PhD thesis, 2010.

[32] Lushan Han, Tim Finin, Cynthia Parr, Joel Sachs, and Anupam Joshi. RDF123: From Spreadsheets to RDF. In Amit Sheth, Steffen Staab, Mike Dean, Massimo Paolucci, Diana Maynard, Timothy Finin, and Krishnaprasad Thirunarayan, editors, *The Semantic Web - ISWC 2008: 7th International Semantic Web Conference, ISWC 2008, Karlsruhe, Germany, October 26-30, 2008.*, pages 451–466, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg. ISBN 978-3-540-88564-1. URL http://dx.doi.org/10.1007/978-3-540-88564-1_29.

[33] Eric N Hanson. *A performance analysis of view materialization strategies*, volume 16. ACM, 1987.

[34] Steve Harris and Andy Seaborne. SPARQL Query Language for RDF. W3C Recommendation, W3C, March 2013. URL http://www.w3.org/TR/rdf-sparql-query/.

[35] 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*, volume 1486 of *CEUR Workshop Proceedings*, October 2015. URL http://ceur-ws.org/Vol-1486/paper_70.pdf.

[36] 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 Harald Sack, Eva Blomqvist, Mathieu d'Aquin, Chiara Ghidini, Paolo Simone Ponzetto, and Christoph Lange, editors, *The Semantic Web – Latest Advances and New Domains (ESWC 2016)*, volume 9678 of *Lecture Notes in Computer Science*, pages 709–723. Springer, 2016. ISBN 978-3-319-34129-3. doi: 10.1007/978-3-319-34129-3_43. URL http://dx.doi.org/10.1007/978-3-319-34129-3_43.

[37] Pieter Heyvaert, **Anastasia Dimou**, Ben De Meester, Tom Seymoens, Aron-Levi Herregodts, Ruben Verborgh, Dimitri Schuurman, and Erik Mannens. Specification and Implementation of Mapping Rule Visualization and Editing: MapVOWL and the RMLEditor. *Journal of Web Semantics*, 2017. first two co-first authors, under revision.

[38] Nikolaos Konstantinou, Dimitrios-Emmanuel Spanos, Dimitris Kouis, and Nikolas Mitrou. An approach for the Incremental Export of Relational Databases into RDF Graphs. *International Journal on Artificial Intelligence Tools*, 24, 2015. ISSN 1793-6349. doi: https://doi.org/10.1142/S0218213015400138. URL http://www.worldscientific.com/doi/abs/10.1142/S0218213015400138.

[39] Christoph Lange. Krextor - an Extensible Framework for Contributing Content Math to the Web of Data. In *Proceedings of the 18th Calculemus and 10th International Conference on Intelligent Computer Mathematics*, MKM'11, pages 304–306, Berlin, Heidelberg, 2011. Springer-Verlag. ISBN 978-3-642-22672-4. URL http://dl.acm.org/citation.cfm?id=2032713.2032745.

[40] Andreas Langegger and Wolfram Wöß. Xlwrap – querying and integrating arbitrary spreadsheets with sparql. In Abraham Bernstein, David R. Karger, Tom Heath, Lee Feigenbaum, Diana Maynard, Enrico Motta, and Krishnaprasad Thirunarayan, editors, *The Semantic Web - ISWC 2009: 8th International Semantic Web Conference, ISWC 2009, Chantilly, VA, USA, October 25-29, 2009*, pages 359–374, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. ISBN 978-3-642-04930-9. URL http://dx.doi.org/10.1007/978-3-642-04930-9_23.

[41] Timothy Lebo, Satya Sahoo, and Deborah McGuinness. PROV-O: The PROV Ontology. W3C Recommendation, W3C, April 2013. https://www.w3.org/TR/prov-o/.

[42] Maurizio Lenzerini. Data Integration: A Theoretical Perspective. In *Proceedings of the Twenty-first ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '02, pages 233–246, New York, NY, USA, 2002. ACM. ISBN 1-58113-507-6. doi: 10.1145/543613.543644. URL http://doi.acm.org/10.1145/543613.543644.

[43] Nuno Lopes, Stefan Bischof, Stefan Decker, and Axel Polleres. On the Semantics of Heterogeneous Querying of Relational, XML and RDF Data with XSPARQL. In *Proceedings of the 15th Portuguese Conference on Artificial Intelligence*, EPIA'11, 2011.

[44] Akifumi Makinouchi. A consideration on Normal Form of Not-necessarily-normalized Relation in the Relational Data Model. In *Proceedings of the Third International Conference on Very Large Data Bases - Volume 3*, VLDB '77, pages 447–453. VLDB Endowment, 1977. URL http://dl.acm.org/citation.cfm?id=1286580.1286627.

[45] Wouter Maroy, **Anastasia Dimou**, Dimitris Kontokostas, Ben De Meester, Ruben Verborgh, Jens Lehmann, Erik Mannens, and Sebastian Hellmann. Sustainable Linked Data Generation: The Case of DBpedia. In Claudia d'Amato, Miriam Fernandez, Valentina Tamma, Freddy Lecue, Philippe Cudré-Mauroux, Juan Sequeda, Christoph Lange, and Jeff Heflin, editors, *The Semantic Web – ISWC 2017: 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part II*, pages 297–313, Cham, 2017. Springer International Publishing. ISBN 978-3-319-68204-4. doi: 10.1007/978-3-319-68204-4_28. URL https://doi.org/10.1007/978-3-319-68204-4_28.

[46] Freddy Priyatna, Oscar Corcho, and Juan Sequeda. Formalisation and Experiences of R2RML-based SPARQL to SQL Query Translation Using Morph. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14, pages 479–490, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2744-2. doi: 10.1145/2566486.2567981. URL http://doi.acm.org/10.1145/2566486.2567981.

[47] Eric Prud'hommeaux and Andy Seaborne. SPARQL Query Language for RDF. W3C Recommendation, W3C, January 2008. URL http://www.w3.org/TR/rdf-sparql-query/.

[48] François Scharffe, Ghislain Atemezing, Raphaël Troncy, Fabien Gandon, Serena Villata, Bénédicte Bucher, Fayçal Hamdi, Laurent Bihanic, Gabriel Képéklian, Franck

Cotton, Jérôme Euzenat, Zhengjie Fan, Pierre-Yves Vandenbussche, and Bernard Vatant. Enabling linked data publication with the Datalift platform. In *Proceedings of the AAAI workshop on semantic cities*, Toronto, Canada, July 2012. URL https://www.aaai.org/ocs/index.php/WS/AAAIW12/paper/download/5349/5678.

[49] Juan F. Sequeda and Daniel P. Miranker. Ultrawrap: {SPARQL} execution on relational data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 22:19 – 39, 2013. ISSN 1570-8268. doi: https://doi.org/10.1016/j.websem.2013.08.002. URL http://www.sciencedirect.com/science/article/pii/S1570826813000383.

[50] Claus Stadler, Jörg Unbehauen, Patrick Westphal, Mohamed Ahmed Sherif, and Jens Lehmann. Simplified RDB2RDF Mapping. In *Proceedings of the WWW2015 Workshop on Linked Data on the Web*, volume 1409 of *CEUR Workshop Proceedings*, 2015. URL http://ceur-ws.org/Vol-1409/paper-09.pdf.

[51] Ruben Verborgh, Miel Vander Sande, Olaf Hartig, Joachim Van Herwegen, Laurens De Vocht, Ben De Meester, Gerald Haesendonck, and Pieter Colpaert. Triple Pattern Fragments: a Low-cost Knowledge Graph Interface for the Web. *Journal of Web Semantics*, 37–38:184–206, March 2016. ISSN 1570-8268. doi: doi:10.1016/j.websem.2016.03.003. URL http://linkeddatafragments.org/publications/jws2016.pdf.

Κακής απ' αρχής γίγνεται τέλος κακόν.
Από κακή αρχή το τέλος είναι κακό.

Ευριπίδης

Bad from the beginning is the end evil.

Euripides

4

Quality

Methodology for uniform mapping rules and Linked Data assessment and refinement for generating higher quality Linked Data

This chapter discusses the quality assessment aspects of Linked Data generation. Looking into the mapping rules that generate Linked Data, this chapter addresses the quality assessment which is applied to mapping rules and allows to refine the mapping rules that generate Linked Data, instead of assessing the quality of the generated Linked Data. Thus, this chapter explains a methodology for mapping rules quality assessment and refinement and it details how useful this methodology is to generate Linked Data of higher quality.

chapter's outline In the next section (Section 4.1), we introduce the problem of assessing Linked Data sets' quality, not only once they are consumed, but primarily before they are even generated. In Section 4.2, we present the state of the art with respect to Linked Data quality. Then, in Section 4.3, we present our methodology for assessing, not directly the generated Linked Data sets, but the mapping rules which define how they should be generated. In Section 4.4, we present a primary approach for refining mapping rules and, in Section 4.6, we present the results of our solution's evaluation.

chapter's sources This chapter is based on the following papers:

1. **Anastasia Dimou**, Dimitris Kontokostas, Markus Freudenberg, Ruben Verborgh, Jens Lehmann, Erik Mannens, Sebastian Hellmann, and Rik Van de Walle. Assessing and Refining Mappings to RDF to Improve Dataset Quality. In Marcelo Arenas, Oscar Corcho, Elena Simperl, Markus Strohmaier, Mathieu d'Aquin, Kavitha Srinivas, Paul Groth, Michel Dumontier, Jeff Heflin, Krishnaprasad Thirunarayan, and Steffen Staab, editors, *The Semantic Web – ISWC 2015*, volume 9367 of *Lecture Notes in Computer Science*, pages 133–149. Springer International Publishing, Oct 2015. doi: 10.1007/978-3-319-25010-6_8. URL `http://link.springer.com/chapter/10.1007/978-3-319-25010-6_8`

2. **Anastasia Dimou**, Dimitris Kontokostas, Markus Freudenberg, Ruben Verborgh, Jens Lehmann, Erik Mannens, Sebastian Hellmann, and Rik Van de Walle. Test-driven Assessment of [R2]RML Mappings to Improve Dataset Quality. In *Proceedings of the 14th International Semantic Web Conference: Posters and Demos*, volume 1486 of *CEUR Workshop Proceedings*, October 2015. URL `http://ceur-ws.org/Vol-1486/paper_108.pdf`

3. **Anastasia Dimou**, Dimitris Kontokostas, Markus Freudenberg, Ruben Verborgh, Jens Lehmann, Erik Mannens, and Sebastian Hellmann. DBpedia Mappings Quality Assessment. In Takahiro Kawamura and Heiko Paulheim, editors, *Proceedings of the 15th International Semantic Web Conference: Posters and Demos*, volume 1690 of *CEUR Workshop Proceedings*, October 2016. URL `http://ceur-ws.org/Vol-1690/paper97.pdf`

4.1 Introduction

The Linked Open Data (LOD) cloud[1] consisted of 12 Linked Data sets in 2007, grew to almost 300 in 2011[2], and, by the end of 2014, counted up to 1,100[3]. Although more and more data is published as Linked Data, the Linked Data sets' quality varies significantly, ranging from expensively curated to relatively low quality Linked Data sets [31]. Nevertheless, high quality Linked Data is an important factor for the success of the envisaged Semantic Web, as machines are inherently intolerant to interpret unexpected input.

Data quality is commonly conceived as "fitness for use" for a certain application or use case [21]. A *data quality assessment metric, measure,* or *indicator* is a procedure for measuring a data quality dimension [7]. A *data quality assessment methodology* is defined as the process of evaluating if a piece of data meets the information that consumers need in a specific use case [7]. There are four sets of dimensions that can be applied to assess Linked Data quality [31]: (i) *accessibility*, involving aspects related to access, authenticity and retrieval of Linked Data to obtain either the entire or some portion of the data for a particular use case; (ii) *intrinsic*, focusing on whether information correctly (syntactically and semantically), compactly and completely represents the real world and whether information is logically consistent in itself; (iii) *contextual*, that highly depends on the context of the task at hand; and (iv) *representational*, capturing aspects related to the data design, such as the representational conciseness, interoperability, interpretability, and versatility.

In this work, we are interested in the quality of the generated Linked Data compared to the ontologies and vocabulary definitions of its schema. The uppermost goal is aiding agents to acquire valid and high quality Linked Data by annotating (semi-)structured data. Therefore, among the different quality dimensions, we focus, with this work, on the ones which are related to the *intrinsic* dimensions of Linked Data quality [31]. More particular, we cover the following: (i) *syntactic validity*, the degree to which certain Linked Data conforms to the serialization format's specification; (ii) *semantic accuracy* the degree to which data values correctly represent the real world facts; and (iii) *consistency*, the degree to which Linked Data is free of (logical/formal) contradictions with respect to particular knowledge representation and inference mechanisms.

In this context, a violation of a quality metric might derive from (i) *incorrect usage of schemas* in the mapping rules; and (ii) errors in the *original data source*. The latter can be resolved by cleansing the data, thus, we focus on the former, which is related to the Linked Data generation process. Kontokostas et al. [24] observed that the most frequent violations are similar and related to the Linked Data set's schema, namely the vocabularies or ontologies terms used to annotate the original data. In the case of (semi-)structured data, the Linked Data set's schema derives from the set of classes and properties specified within the mapping rules. A mapping rule might use a single ontology or vocabulary to annotate the data, or a proprietary vocabulary can be generated as the data is semantically annotated. Lately, combinations of different ontologies and vocabularies are often used to annotate data [30], which increases the likelihood of encountering such violations.

[1] Linked Data cloud, `http://lod-cloud.net/`

[2] Linked Data cloud state 2007, `http://lod-cloud.net/state`

[3] Linked Data cloud 2011, `http://linkeddatacatalog.dws.informatik.uni-mannheim.de/state/`

Only recently research efforts focused on Linked Data quality tracking and assessment, as it is summarized by Zaveri et al. [31]. Nevertheless, such approaches remain independent of the Linked Data generation and publication workflow –quality assessment is not even mentioned in the best practices for publishing Linked Data [20], but it is mentioned in the more recent – but also more generic, as it refers to data on the Web in general– W3C recommendation for Data on the Web Best Practices [13] and its accompanying Data Quality vocabulary [6]. Existing quality assessment refers to already published Linked Data and, in most cases, is performed by third parties rather than data publishers.

Therefore, incorporating quality assessment results corresponds to a *Linked Data feedback loop*: existing Linked Data infrastructures still neither intuitively process end-users' input, nor properly propagate the modifications to the mapping rules and original data. Consequently, the results are rarely and, if so, manually used to adjust the Linked Data set, with the risk of being overwritten when a new version of the original data is published.

In this work, Linked Data quality assessment is extended from being applied during data *consumption* to also cover data *publication*. The assessment process, which is normally applied to the final Linked Data set, is adjusted, so it can be applied to the mapping rules as well. This allows agents to discover inconsistencies in Linked Data, before they are even generated. Our methodology (i) augments the generation and publishing workflow of (semi-)structured data sources with *systematic Linked Data quality assessment* for both mapping rules and resulting Linked Data; and (ii) *automatically suggests mapping refinements* based on results of the uniform quality assessment. We consider iterative, uniform, and gradual test-driven quality assessment to improve the Linked Data set's overall quality.

4.2 State of the Art

Different approaches were developed that tackle various aspects of Linked Data quality by introducing systematic methodologies. These approaches can be broadly classified into (i) manual, for instance, Acosta et al. [5], Bizer and Cyganiak [7], Mendes et al. [27], Zaveri et al. [31]; (ii) semi-automated, for instance, Flemming [14], Hogan et al. [19]; or (iii) automated, for instance, Debattista et al. [12], Guéret et al. [17], Kontokostas et al. [24]. Depending on the approach, drawbacks still remain: inability to produce easily interpretable results, considerable amount of user involvement, application on specific Linked Data sets only, or inability to evaluate a complete Linked Data set during the assessment.

Test-case based quality assessment is inspired by test-driven software development [24]. In software development, every function should be accompanied by a set of unit tests to ensure the correct behaviour of that function through time. As Linked Data generation can be considered an engineering task, similar to software development, test-case based quality assessment fits well to the generation and publishing workflow. In the same context, in test-case based quality assessment, every vocabulary, ontology, dataset, or application can be associated by a set of data quality test cases. A *test-case* then is a data constraint that involves one or more triples, while a *test-set* is a set of test cases for assessing a Linked Data set. Assigning test cases in ontologies results in tests that can be reused by Linked Data sets sharing the same semantic schema.

Many constraint languages, which act as test-case definition languages, were defined to describe constrains, such as IBM Resource shapes [29], Description Set Profiles [9], and Shape Expressions [28]. However, the Shapes Constraint Language (SHACL), a language for validating RDF graphs against a set of conditions, recently became candidate for W3C recommendation [22]. The prevalent test-case assessment implementation is RDFUnit [24]. Other implementations, that also support SHACL, are Corese SHACL[4], Netage SHACL Engine[5], TopBraid SHACL API[6], and RDFUnit. The test-case definition language of RDFUnit is SPARQL, as it is convenient to directly query for identifying violations. However, other schema languages are also supported. A library of common patterns was introduced in Kontokostas et al. [24, Table 1] and was further extended[7]. Moreover, RDFUnit includes support for automatic schema enrichment via DL-Learner [25] machine learning algorithms. RDFUnit can check a Linked Data set against multiple schemas without performing any reasoning/action to detect inconsistencies between the different schemas.

The approach described by Fürber and Hepp [15, 16] also advocates the use of SPARQL and SPIN for Linked Data quality assessment by defining a set of generic SPARQL queries to identify missing or invalid literal values and datatypes as well as functional dependency violations. Similarly, the *Pellet Integrity Constraint Validator*[8], translates OWL integrity constraints into SPARQL queries to assess a Linked Data set.

4.3 Quality Assessment

The earlier a Linked Data set's quality is assessed, the better, as the cost of fixing a bug rises exponentially when a task progresses [8]. Therefore, we extended Linked Data quality assessment from being applied during *consumption* to also cover *production*. To achieve that, the assessment process, which is normally applied to Linked Data, is adjusted, so it can be applied to mapping rules too. This way, our methodology augments the generation workflow of (semi-)structured data sources with *systematic Linked Data quality assessment* applicable to both mapping rules and generated Linked Data set.

To be more precise, instead of assessing Linked Data for its schema consistency, quality assessment can be applied directly to the corresponding mapping rules, *before* they are used to generate Linked Data. This is possible because their assessment results are correlated, since mapping rules specify how Linked Data will be formed. For instance, violations related to the range of a certain property can be assessed by inspecting the corresponding mapping rules, which define how Linked Data with this property is generated.

Even though mapping rules' quality assessment can cover many violations related to vocabularies and ontologies which are used to annotate the data, some schema-related violations depend on how the mapping rules are *instantiated* on the original data. For instance, a violation occurs if an object of integer datatype is instantiated with a floating-point

[4] Corese SHACL, `http://wimmics.inria.fr/corese`
[5] Netage SHACL Engine, `http://www.netage.nl/`
[6] TopBraid SHACL API, `https://github.com/TopQuadrant/shacl`
[7] `https://github.com/AKSW/RDFUnit/blob/master/configuration/patterns.ttl`
[8] Pellet Integrity Constraint Validator, `http://clarkparsia.com/pellet/icv/`

value from the original source. Thus, a *uniform* way of incrementally assessing the quality of Linked Data should cover both the mapping rules and the generated Linked Data set.

In the next subsection (Section 4.3.1), we briefly introduce the Linked Data quality assessment methodology [24] which is applied to Linked Data sets and is adjusted to be applied to mapping rules as well as described in Section 4.3.2.

4.3.1 Linked Data Quality Assessment

The basic notation of the quality assessment methodology introduced by Kontokostas et al. [24] goes as follows: A *Data Quality Test Pattern* (DQTP) is a template using %%v%% as syntax for variable placeholders. A Test Auto Generator, based on a DQTP, takes a Linked Data set as input, searches for its schema information and returns test cases. The test cases are generated by applying a pattern binding, i.e., a valid DQTP variable replacement, to a DQTP. This results in an executable test case. Each result of a test case is considered to be a violation of a test case, whereas empty results means that the test case is successful. Initially, Kontokostas et al. [24] methodology supported the most commons OWL axioms. Nevertheless, the methodology is easily extensible to support other constraints constructs, such as the recently W3C recommended SHACL [22].

RDFUnit [24] is a Linked Data validation framework that implements the test driven data assessment methodology, relying on SPARQL query templates to describe DQTP. The schema can be in the form of RDFS or OWL axioms that RDFUnit translates into SPARQL under Closed World Assumption (CWA) and Unique Name Assumption (UNA). RDFUnit can check a Linked Data set against multiple schemas but when this occurs, RDFUnit does not perform any reasoning/action to detect inconsistencies between the different schemas. For rapid test case instantiation, RDFUnit supports a pattern-based SPARQL-template engine which can easily bind variables into patterns. It is implemented in a Java component[9] and released as open source under the Apache licence.

An initial library of 17 common patterns was developed in [24, Table 1] which was further extended[10]. These test cases cover validation against: domain, range, class and property disjointness, (qualified) cardinality, (inverse) functionality, (a)symmetricity, irreflexivity, and deprecation. The test cases applied to the mapping rules too, are outlined in Table 4.1.

```
1  <http://example.com/4> a foaf:Project ;
2  <http://example.com/4> foaf:age "42"^^xsd:float .
```

Listing 4.1: *extract of example Linked Data set*

For instance, the extracted predicate of the aforementioned RDF triple is `foaf:age` (Listing 4.1, Line 2). This predicate is normally used with instances of `foaf:Agent` type for its domain and an integer datatype for its range, but the RDF triple's subject is of `foaf:Project` type (Line 1) and the object has float as its datatype. The test case pat-

[9] RDFUnit, `https://github.com/AKSW/RDFUnit/`

[10] RDFUnit SPARQL templates, `https://github.com/AKSW/RDFUnit/blob/master/configuration/patterns.ttl`

terns applied to the aforementioned example for assessing its datatype and its instantiation are indicatively presented. On the left, the `where` clause of a SPARQL template query that assesses the datatype is presented. On the right it is presented how it is instantiated:

```
1  ?resource %%P1%% ?c.
2  FILTER (DATATYPE(?c) != %%D1%%)
```

Listing 4.2: *template of quality assessment test-case applied to Linked Data*

```
1  ?resource foaf:age ?c.
2  FILTER (DATATYPE(?c) != xsd:int)
```

Listing 4.3: *instantiated template of a corresponding test-case applied to Linked Data*

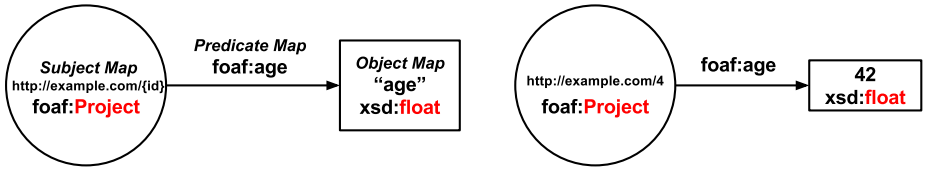


Figure 4.1: *Similarly occurs for the Subject Map.*

4.3.2 Mapping Rules Quality Assessment

RML mapping rules have a native semantically enhanced representation and are written from the viewpoint of the generated Linked Data. Therefore, they can be processed as Linked Data. Our quality assessment process targets (i) *syntactic validation* against the corresponding serialization's specification, which is handled directly by a Linked Data parser, for instance turtle for R2RML; (ii) *consistency validation* of the mapping rules per se against both the R2RML and RML schema which is handled directly by RDFUnit, and mainly (iii) *consistency validation* and *quality assessment* of the Linked Data to be generated against the schema defined in the mapping rules which is handled by emulating the Linked Data to assess its schema conformance. In more details:

Mapping rules per se The mapping rules' validation against the [R2]RML schema is directly handled by RDFUnit by extending the supported OWL axioms. New RDFUnit test cases auto generators were defined to support all OWL axioms in R2RML and RML ontology. For instance, each Triples Map should have exactly one Subject Map. This part of the assessment produces 78 automatically generated test cases.

Linked Data projected by its mapping rules To validate Linked Data based only on mapping rules that state how this Linked Data is generated, we considered the same set of schema validation patterns normally applied on the Linked Data set (cf. Table 4.1). Nevertheless, instead of validating the predicate against the subject and object, we extract the predicate from the Predicate Map and validate it against the Term Maps that define how the subject and object will be formed. For instance, the extracted predicate expects a Literal as object, but the Term Map that generates the object can be a Referencing Object Map that generates resources instead.

Table 4.1: *Semantic violations detected after assessing the mapping rules' validity. The first column describes the type of violation, the second its level (Warning or Error), the third the expected* RDF *term according to the schema (ontology or vocabulary), while the fourth the* RML *term map which define how the* RDF *term is generated. The last specifies potential refinement that can be applied.*

OWL axiom – Violation type	Level	Expected	Defined
class disjointness	E	SbjMap	SbjMap
property disjointness	E	PreMap	PreMap
`rdfs:range` – class type	E	PreMap	(Ref)ObjMap
`rdfs:range` – IRI instead of literal	E	PreMap	(Ref)ObjMap
`rdfs:range` – literal instead of IRI	E	PreMap	ObjMap
`rdfs:range` – missing datatype	E	PreMap	(Ref)ObjMap
`rdfs:range` – incorrect datatype	E	PreMap	(Ref)ObjMap
missing language	E	ObjMap	ObjMap
`rdfs:domain`	E	PreMap	SbjMap
missing `rdf:type`	W	SbjMap	SbjMap
deprecation	W	PreMap	PreMap
`owl:complementOf`	W	PreMap	SbjMap

To achieve this, the properties and classes in the mapping rules are identified and their namespaces are used to retrieve the schemas and generate test cases as if the actual Linked Data was assessed. The RDFUnit test cases were extended to apply to the mapping rules by adjusting the assessment queries and generating corresponding so called "manual test cases"[11].

For instance, a SPARQL test case's `WHERE` clause that assesses a missing language is:

```
1  ?resource ?P1 ?c .
2  FILTER (lang(?c) = '')
```

Listing 4.4: *WHERE clause of test-case applied to Linked Data and assess the language*

To detect the same violation from a mapping rule, the `WHERE` clause is adjusted as follows:

```
1  ?poMap rr:predicate ?P1 ;
2         rr:objectMap ?resource .
3  ?P1 rdfs:range rdf:langString .
4  FILTER NOT EXISTS {?resource rr:language ?lang}
```

Listing 4.5: *WHERE clause of test-case applied to mapping rules that assess the language*

The test case patterns applied to the above example for assessing its datatype but adjusted to the corresponding mapping rules and its instantiation are indicatively presented.

[11] `https://github.com/AKSW/RDFUnit/blob/master/data/tests/Manual/www.w3.org/ns/r2rml/rr.tests.Manual.ttl`

```
1  <#Mapping> rr:subjectMap [
2         rr:template "http://example.com/{id}";
3        rr:class foaf:Project ];
4        rr:predicateObjectMap [
5             rr:predicate foaf:age;
6             rr:objectMap [
7                 rml:reference "age" ;
8                 rr:datatype xsd:float ] ].
```

Listing 4.6: *Mapping rules for Linked Data generation*

Then the `where` clause of the same test case that assesses the datatype consistency applied to the mapping rules is adjusted. On the left, the `where` clause of a SPARQL template query that assesses the mapping rule's datatype is presented. On the right it is presented how it is instantiated:

```
1  ?resource rr:predicateObjectMap ?poMap.
2  ?poMap rr:predicate %%P1%%;
3         rr:objectMap ?objM.
4  ?objM rr:datatype ?c.
5  FILTER (?c != %%D1%%)
```

Listing 4.7: *template of quality assessment test-case applied to mapping rules*

```
1  ?resource rr:predicateObjectMap ?poMap.
2  ?poMap rr:predicate foaf:age;
3         rr:objectMap ?objM.
4  ?objM rr:datatype ?c.
5  FILTER (?c != xsd:int)
```

Listing 4.8: *instantiated template of quality assessment test-case applied to mapping rules*

The Linked Data quality assessment is *predicate-driven* in principle, thus the corresponding mapping rules assessment is *Predicate-Map-driven* as well. The expected value for `foaf:age`, as the predicate is derived from the Predicate Map, is compared to the defined one, as derived from the corresponding Object Map, (Listing 4.6, Line 8). Similarly, applying a test case to the aforementioned mapping rule (cf. Listing 4.6), to assess the domain's consistency, a violation is registered, as `foaf:age` has `foaf:Person` and not `foaf:Project` as domain, given that the ontology does not define `foaf:Person` as equivalent or subclass of `foaf:Project`. This test case's SPARQL query has as follows:

```
1  SELECT DISTINCT ?resource WHERE {
2    ?mappingTo rr:subjectMap ?resource .
3    { ?resource rr:class ?T1 . } UNION {
4      ?mapping rr:predicateObjectMap ?classPoMap .
5      ?classPoMap rr:predicate rdf:type ;
6                  rr:objectMap/rr:constant ?T1 . }
7    ?mappingFrom rr:predicateObjectMap ?poMap .
8    ?poMap rr:predicate/rdfs:range ?T2  ;
9           rr:objectMap ?objM .
10   ?objM rr:parentTriplesMap ?mappingTo .
11   FILTER NOT EXISTS {
12    ?T2 (rdfs:subClassOf|(owl:equivalentClass|^owl:equivalentClass))* ?T1.}}
```

Listing 4.9: *complete SPARQL query test-case for validating a mapping rule's domain*

In order for our assessment to be complete, the defined test cases cover all possible alternative ways of defining equivalent mapping rules that generate the same triples. For instance, the default way to generate the type for a resource is through the rr:class property in the Subject Map (e.g., Line 3 of Listing 4.6). However, one may also define the type via a Predicate Object Map having rdf:type in its Predicate Map.

RDFUnit can annotate test cases by requesting additional variables and binding them to specific result properties. Using the example of Listing 4.9 we map, for instance, variable

?T1 as `spin:violationValue` and variable ?T2 as the expected class. When a violation is identified, the annotations are applied and a result like the following is registered:

```
1  <5b7a80b8> a rut:ExtendedTestCaseResult;
2    rut:testCase rutt:rr-produces-range-errors ;
3    # (...) Further result annotations
4    spin:violationRoot ex:objectMapX ;
5    spin:violationPath rr:class ;
6    spin:violationValue foaf:Project ;
7    rut:missingValue foaf:Person ;
8    ex:erroneousPredicate  foaf:age ;
```

Listing 4.10: *Quality assessment's results*

Limitations Some of the test cases normally applied to Linked Data rely on the final values or refer to the complete Linked Data set and thus, can only be validated after the Linked Data generation is performed –detected at *data-level* quality assessment (DQA). Such examples are (qualified) cardinality, (inverse) functionality, (a)symmetricity, and irreflexivity. For instance, we cannot validate an inverse functional property, such as foaf:homepage without the actual values. Invalid mapping rules can occur, as the mapping rules are instantiated based on the input source, even though the mapping rules appear to be valid. For instance, if, instead of "http://dbpedia.org/resource/United_States", the input data returns "American", it would result in generating the URI <American>, which is invalid.

4.4 Mapping Rules Refinements

Mapping rules refinements alleviate Linked Data quality assessment issues on consumption-side. If violations are only corrected in the resulting Linked Data set, they will have to be corrected every time a new version of this set is generated. Also, when a violation is found, it is not straightforward to discover its cause, as the connection with the mapping rules and source data is not always apparent. A more effective approach is to refine the mapping rules that generate this Linked Data, so the violation cannot occur in future versions. More, if the violation is associated with a mapping rule, it can be addressed directly on the place where it occurred, and instead of having to regenerate the entire Linked Data set, only the Linked Data affected by the refinement needs to be regenerated to correct the violation.

The results of the Mapping Quality Assessment (MQA) can be used to suggest modifications or even automatically refine mapping rules. The RDFUnit ontology provides multiple result representations in different formats [23], including RDF-based serialisations, using the rut:ExtendedTestCaseResult result type. Therefore, its results are easily processed by a quality assessment and refinement agent that can automatically add and delete triples or suggest actions to a human agent (data publisher). In Table 4.2, we outline all examined violation patterns and indicate which Term Map should be refined and how. The suggested refinements are the minimum required actions to refine the mapping rules, e.g., turn an Object Map into generated resources instead of literals, by turning its termtype from `rr:IRI` to `rr:Literal`, and serve as indicative *proof-of-concept* of the automation's feasibility.

Below, the most frequent *domain* and *range-level* potential refinements, applicable in the case of our running example, are explained:

Table 4.2: *Semantic violations detected after assessing the mapping rules' validity. The first column describes the type of violation, while the second specifies potential refinement that can be applied.*

OWL axiom – Violation type	**Refinement**
class disjointness	–
property disjointness	–
`rdfs:range` – class type	DEL: ObjMap ADD: PreMap domain to RefObjMap
`rdfs:range` – IRI instead of literal	DEL: (Ref)ObjMap ADD: ObjMap with literal termType
`rdfs:range` – literal instead of IRI	DEL: ObjMap ADD: (Ref)ObjMap or ADD: ObjMap with IRI termType
`rdfs:range` – missing datatype	DEL: ObjMap ADD: ObjMap with PreMap datatype
`rdfs:range` – incorrect datatype	DEL: (Ref)ObjMap ADD: ObjMap with PreMap datatype
missing language	–
`rdfs:domain`	ADD: PreMap domain to SbjMap
missing `rdf:type`	ADD: PreMap domain to SbjMap
deprecation	–
`owl:complementOf`	–

domain-level refinements Automatically refining *domain-level* violations requires comparing recursively the type(s) assigned to the Subject Map with each predicate's domain, as specified at the different Predicate Maps. If not explicitly defined or inferred via a subclass, the predicate's domain is additionally assigned. This also requires a follow-up check for disjoint classes, which is of crucial importance especially when composition of different vocabularies and ontologies occurs. For instance, in the case of our running example, the following additions and deletions are required:

```
1  DEL: ex:subjectMapX rr:class foaf:Project .
2  ADD: ex:subjectMapX rr:class foaf:Person .
```

Listing 4.11: *Domain-level refinement*

(Linked) Data quality is "fit for use", thus its improvement also depends on the use case. What needs to be improved, namely refining the mapping rules or adjusting the semantic schema, might depend on the use case. For instance, in our example, besides refining the mapping rules, adjustments on ontology level, such as defining `foaf:Project` as equivalent or subclass of `foaf:Person`, could be applied too.

range-level refinements Dealing with *range-level* violations requires different actions, depending on the value of the Object Map or Referencing Object Map. The Predicate Map is used to retrieve the property and identify its range, which is then compared to the corresponding Object Map or Referencing Object Map.

If the Predicate Map contains an *object property*, for instance, but the object is generated by a Referencing Object Map, which generates resources with type different than

the predicate's range –as defined by the corresponding vocabulary or ontology– the predicate's range is added as class to the Referencing Object Map. Nevertheless, a new round of assessment is required, as an adjustment might cause new violations, such as disjointness violation due to conflicts with other properties' expected domain. If the Predicate Map contains a *datatype property*, as the aforementioned example, which expects a datatype different than the defined one, the object's datatype needs to be adjusted in the corresponding Object Map, for instance as follows:

```
1  DEL: ex:objectMapX rr:datatype xsd:float .
2  ADD: ex:objectMapX rr:datatype xsd:integer .
```

Listing 4.12: *Range-level refinement on datatype*

The number of automated resolutions for violations detected at mapping rules level depends on the number of (i) iterations over the data chunks of the input source (e.g., number of rows); (ii) references to the input source (e.g., number of referred columns); and (iii) returned values from the input source for each reference.

To be more precise, if $\mathcal{I}$ is the number of iterations, $\mathcal{R}$ is the set of references to the input source, and $\mathcal{V}(r)$ values are returned for $r \in \mathcal{R}$, then the total number of errors per violation is equal to the number of triples generated from this mapping definition: $\mathcal{I} \cdot \prod_{r \in \mathcal{R}} \mathcal{V}(r)$. This means that the number of errors per violation identified (and resolved) at mapping level grows linearly in function of the number of iterations, and geometrically, in the worst case, if multiple references and returned values occur.

For instance, assuming a mapping rule with 2 references to the input, where up to 3 values can be returned for the first reference and up to 4 values for the second, contains a violation. Applied to a data source with data in XML format which contains 1,000 elements, this could cause up to 12,000 error-prone triples in the worst case.

4.5 RML Validator

The RMLValidator is a tool which can be used to perform quality assessment both on *Linked Data*, as well as on *mapping rules* level. In principle, the RMLValidator wraps the RMLProcessor and RDFUnit together.

The RMLValidator consists of sub-modules, as shown in Figure 4.2. In more details:

MapDocHandler parses and skolemizes the mapping rules. Skolemization is required because, typically, the Term Maps are defined as blank nodes. Thus once a violation is identified, there is no sustainable IRI to indicate where the violation occurred. This module has been straightforwardly taken from the RMLMapper (Section 3.4);

RDFUnit is used to validate the mapping rules which is derived after the MapDocHandler's processing of the original set of mapping rules (MQA), and the resulting Linked Data (DQA) as they are generated by the RMLProcessor;

RMLMapper is used to execute the mapping rules and generate the Linked Data.

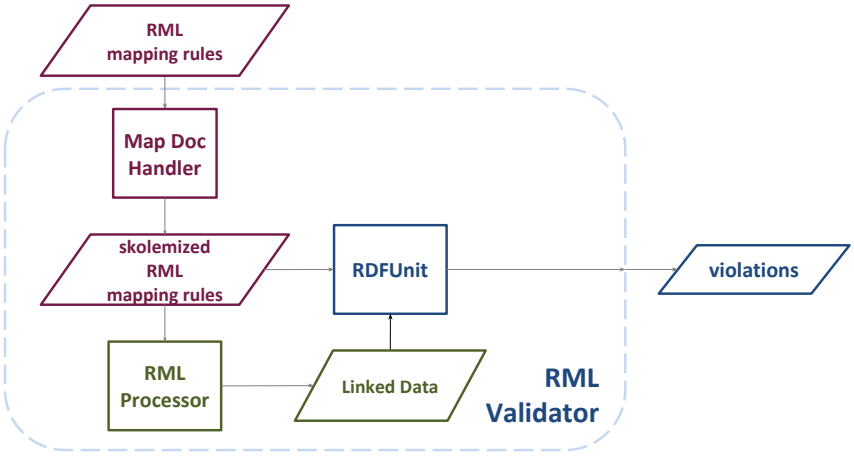


Figure 4.2: *RMLValidator architecture*

4.6 Evaluation

We evaluated our approach against different use cases (Section 4.6.1) and here we present our results (Section 4.6.2).

4.6.1 Use cases

The following Linked Data sets were used for our evaluation:

DBpedia [26] (more details is Section 6.4) provides a *collaborative mapping approach* of Wikipedia infoboxes to the DBpedia ontology[12] through the DBpedia *mappings wiki*[13]. DBpedia uses a wiki markup syntax for the mapping rules and the output is adjusted in conformance to the DBpedia ontology. Although DBpedia uses the same wikitext syntax as Wikipedia –its original source– to define the mapping rules, the quality of wikitext-based mapping rules cannot be assessed directly, and thus certainly not in the same way as the result Linked Data set. Thus, we automated the conversion of all DBpedia mapping rules to RML to make them machine-processable. We introduced a *wikitext serialisation* as a new Reference Formulation, since RML can be extended to express mapping rules for any type of input source.

In total, we generated 674 distinct mapping documents for English, 463 for Dutch and a total of 4,468 for all languages. We used the DBpedia 2014 release and focused on a complete evaluation on the English and Dutch language editions, as well as a mapping-only evaluation of all languages supported in the DBpedia mappings wiki. DBpedia originates from crowdsourced and (semi-)structured content and can

[12] DBpedia ontology, `http://wiki.dbpedia.org/Ontology`
[13] DBpedia mapping rules, `http://mappings.dbpedia.org`

thus be considered a noisy Linked Data set. The Mapping Quality Assessment report was provided to the DBpedia community[14], who took advantage to manually refine the DBpedia mapping rules. Automated refinements were not applicable, as the DBpedia framework still functions with the original mapping rules in wiki markup.

Faceted DBLP The Computer Science bibliography (DBLP) collects open bibliographic information from major computer science journals and proceedings. Faceted DBLP builds upon the DBLP++ dataset, an enhancement of DBLP, originally stored in a MySQL database. DBLP mapping rules are originally defined using D2RQ [10] and were converted to RML using D2RQ-to-R2RML[15] to be processable by our workflow. DBLP is a *medium-sized* dataset of very good quality according to our evaluation. Nonetheless, the workflow resulted in improvements.

Contact Details of Flemish Local Governments Dataset (CDFLG) In the scope of the EWI Open Data[16] project, the CDFLG dataset[17] was generated using our workflow [11]. This is a real case of contact details for local governments in Flanders. CDFLG is annotated using the OSLO ontology[18], defined by the Open Standards for Linking Governments Working Group (V-ICT-OR, OSLO) under the OSLO (Open Standards for Local Administrations) Programme. Two subsequent versions of its RML mapping rules were used to generate this Linked Data set and each one was assessed for its quality. The decrease of mapping violations over the mapping rules evolution indicates that our methodology can correctly identify errors.

iLastic Our methodology was used in the use case for iLastic[19] (details in Section 6.2), a research institute founded by the Flemish Government, which publishes its own data regarding researchers, publications, projects, external partners, etc. The mapping rules that specify how Linked Data is generated are stated using RML. After the primary mapping rules were stated, they were fed to our proposed implementation and were refined twice, once based on mapping quality assessment results and once based on data quality assessment results, leading to their final version which is free of violations.

CEUR-WS Our methodology was also applied to the CEUR-WS.org use case (Section 6.5). The ESWC2015 Semantic Publishing Challenge (SPC)[20] is focused on refining and enriching the CEUR-WS.org[21] Linked Data set originally generated at the ESWC2014 edition[22]. It contains Linked Data about workshops, their publications, and their authors. The workflow was well aligned with the requirements of the 2015 challenge and was used to evaluate the previous year's, i.e., 2014, submission based on RML [1] and refine it to produce the base for this year's submission [18].

[14] DBpedia mapping quality assessment report, `http://goo.gl/KcSu3E`
[15] D2RQ-to-R2RML, `https://github.com/RMLio/D2RQ_to_R2RML.git`
[16] EWI Open Data, `http://ewi.mmlab.be`
[17] CDFLG, `http://ewi.mmlab.be/cd/all`
[18] OSLO ontology, `http://purl.org/oslo/ns/localgov#`
[19] iLastic, `http://explore.ilastic.be/`
[20] SPC, `https://github.com/ceurws/lod/wiki/SemPub2015`
[21] CEUR-WS.org, `http://ceur-ws.org`
[22] SemPub2014, `http://challenges.2014.eswc-conferences.org/index.php/SemPub`

4.6.2 Results

In this section, the results are described and certain observations are discussed in more detail for each dataset and overall. The results are aggregated in Table 4.3. For our evaluation, we used a moderate working station which has an 8-core Intel i7 machine with 8GB RAM and 256 SSD HD.

Overall, it is clear that the *computational complexity and time are significantly reduced* when assessing the mapping rules compared to the complete Linked Data set (cf. Table 4.3). It takes 11 seconds to assess the approximately 700 mapping rules of English DBpedia, compared to assessing the whole DBpedia Linked Data set that takes several hours. In the latter case, the assessment requires examining each triple separately to identify, for instance, that 12M triples violated the range of foaf:primaryTopic, whereas with our proposed approach, only 1 triple needs to be assessed. It is indisputable the workflow's *effectiveness* as, in all cases that the Linked Data set generation fully relies on its mapping rules, the majority of violations is addressed. Moreover, if a set of RML mapping rules is assessed for its quality, for every other new data source also mapped using these mapping rules, the quality assessment does not need to be repeated for that part.

Below, we discuss the results for each use case in detail:

DBpedia Most violations in DBpedia have a *range-level* origin. When RDF is generated from the wikitext, the object type is not known and may result in wrong statements, as the DBpedia extraction framework automatically adjusts the predicate/object extraction according to the DBpedia ontology definitions. *Domain-level* violations occur as well, because users manually provide the class a Wikipedia infobox is mapped to and the ontology properties each infobox property will use. Our framework can, in this case, identify mismatches between the user-defined class and the rdfs:domain of each provided property. We observe that 8% of the errors in DBpedia in English and 13% of DBpedia in Dutch can be fixed directly at *mapping-level*. Not only are the errors as such directly pinpointed, but it also takes negligible time to have the refinements of the violations accomplished. The evaluation of all mapping rules for all 27 supported language editions resulted in a total of 1316 *domain-level* violations.

DBLP It has 7 individual violations, mainly related to domain-level violations, leading to 8.1M violated triples. The `swrc:editor` predicate defined in a Predicate Map expects a resource of `swrc:Person` type for its `domain` instead of `foaf:Agent` as defined in the corresponding Subject Map causing 21k errors. Similarly, approximately 3M errors occurred because the `dcterms:bibliographicCitation` exists in a Predicate Map whose domain is `bibo:BibliographicResource`. However, the types of the corresponding Subject Map(s) are `dcmitype:Text`, `foaf:Document` or `swrc:Book` but definitely not the expected one, thus agents should remain warned for potential contradictions. Moreover, the missing range of `foaf:page` and `foaf:homepage` can be fixed by refining the mapping rules but, for links to external resources, it is common practice not to define their type. Except for 12k inverse functional violations for `foaf:homepage` that can not be addressed directly from the mapping rules, all remaining violations (98%) could be refined.

CDFLG In the first version of the CDFLG Linked Data set, we found four violations: One

caused by Predicate Object Maps that all have predicates that expect oslo:Address as their domain. However, the Subject Map is defined to be of type oslo:BasicAddress. In the same context, an incorrect range violation was identified for oslo:availableAt property. In general, violations related to Referencing Object Maps are among the most frequently encountered. Last, the object property schema:nationality was mapped as literal. The second version of CDFLG is a result of manually refining the mapping rules according to the first mapping assessment's results. Besides the *domain level* violation, only 7% of the range violations remained.

iLastic This is particularly interesting because the assessment methodology was used from the primary version of the mapping rules, until they became free of violations. The first version was assessed and, besides the violations with respect to its semantic schema, it even contained R2RML schema violations. For instance, `rr:constant` had a string-valued object instead of a resource. If these mapping rules were used, almost one fourth (25%) of its triples would have been prone to errors. Every violation was fixed after a couple of iterations assessing and refining the mapping rules. For example, `cerif:isClassifiedBy` expects a `cerif:Classification` and not a `skos:Concept`, while `bibo:uri` expects a literal and not a resource as range. Similarly, `dcterms:issued` expects `xsd:date` and not `xsd:gYear`. A violation that occurred repeatedly was associated with `cerif:internalidentifier` that requires a string-valued object, whereas it was associated with an Object Map that generated `xsd:positiveInteger` objects.

CEUR-WS 12 violations were identified in the Linked Data set generated using RML for the ESWC 2014 challenge and 10 out of them could already be detected at the mapping rules. Most of the violations (7) were related to the *domain*, annotating, for instance, resources of type `bibo:Volume` with properties for `bibo:Document` or for `madsrdf:Address`, e.g., for specifying the city, implying unwittingly that resources are both *Documents* and *Addresses*. The rest of the detected violations were related to contradicted datatypes, for instance, incorrectly specifying the datatype as `xsd:gYear`, while it is expected to be a string. The mapping rules for ESWC 2015 submission were produced using our workflow, were assessed, and do not contain violations any more.

Conclusions

In this chapter, we present a methodology for assessing Linked Data quality for (semi-)structured data, originally stemming in heterogeneous data sources. Our methodology focuses on assessing mapping rules, rather than the Linked Data per se.

The quality assessment indicates the root causes of the violations that degrade a Linked Data set's quality and can be actively used to refine the corresponding mapping rules. The automation of refinements or suggestions is facilitated based on a comprehensive analysis of different cases, and encountered violations are addressed at the origin. This essentially allows publishers to correct violations before they even occur. Moreover, fixing violations early avoids propagation where one flawed mapping rule leads to many faulty triples.

Table 4.3: *Evaluation results summary. In the Dataset Assessment part, we provide the number of triples (**Size**) and test-cases (**TC**), evaluation Time (**Time**), Failed test-cases (**Fail.**) and total individual Violations (**Viol.**). In the Mapping Assessment part, we provide the mapping document number of triples (**Size**), evaluation Time (**Time**), Failed TCs (**Fail.**) and Violation instances (**Viol.**). Finally, we provide the number of Linked Data set violations that can be addressed by refining the mapping rules (**Ref.**) and estimated corresponding Linked Data set violations that are resolved (**Affect. triples**).*

	Dataset Assessment					***Mapping Assessment***					**Affect.**
Dataset	**Size**	**TC**	**Time**	**Fail.**	**Viol.**	**Size**	**Time**	**Fail.**	**Viol.**	**Ref.**	**triples**
DBpEn	62M	9,458	16.h	1,128	3.2M	115k	11s	1	160	–	255k
DBpNL	21M	10,491	1.5h	683	815k	53k	6s	1	124	–	106k
DBpAll	–	–	–	–	–	511k	32s	1	1,316	–	–
DBLP	12M	462	12h	7	8.1M	368	12s	2	8	6	8M
iLastic	150k	690	12s	23	37k	825	15s	3	26	23	37k
CDFLG	0.6k	2068	7s	15	678	558k	13s	4	16	13	631
CEUR-WS	2.4k	414	6s	7	783	702	5s	3	12	7	783

The evaluation shows that our methodology is applicable to (i) Linked Data sets without native [R2]RML mapping rules, such as DBLP, (ii) large Linked Data sets, such as DBpedia, and (iii) Linked Data sets in the whole process of defining their mapping rules, such as iLastic. Assessing the quality of mapping rules is more efficient in terms of computational complexity, and requires significantly less time to be executed compared to assessing the entire Linked Data set. As our evaluation indicates, it takes only a few seconds to assess the mapping rules, whereas it can be time-consuming and performance-intensive when this happens at Linked Data set level. Especially with large Linked Data sets, such as DBpedia, this can take up to several hours. Our methodology was adopted by both the community of significant public Linked Data sets, such as DBpedia, and several projects, resulting in published Linked Data of higher quality.

References

[1] **Anastasia Dimou**, Miel Vander Sande, Pieter Colpaert, Laurens De Vocht, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. Extraction and Semantic Annotation of Workshop Proceedings in HTML using RML. In Valentina Presutti, Milan Stankovic, Erik Cambria, Iván Cantador, Angelo Di Iorio, Tommaso Di Noia, Christoph Lange, Diego Reforgiato Recupero, and Anna Tordai, editors, *Semantic Web Evaluation Challenge: SemWebEval 2014 at ESWC 2014, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers*, pages 114–119, Cham, 2014. Springer International Publishing. URL http://dx.doi.org/10.1007/978-3-319-12024-9_15.

[2] **Anastasia Dimou**, Dimitris Kontokostas, Markus Freudenberg, Ruben Verborgh, Jens Lehmann, Erik Mannens, Sebastian Hellmann, and Rik Van de Walle. Assessing and Refining Mappings to RDF to Improve Dataset Quality. In Marcelo Arenas, Oscar Corcho, Elena Simperl, Markus Strohmaier, Mathieu d'Aquin, Kavitha Srinivas, Paul Groth, Michel Dumontier, Jeff Heflin, Krishnaprasad Thirunarayan, and

Steffen Staab, editors, *The Semantic Web – ISWC 2015*, volume 9367 of *Lecture Notes in Computer Science*, pages 133–149. Springer International Publishing, Oct 2015. doi: 10.1007/978-3-319-25010-6_8. URL http://link.springer.com/chapter/10.1007/978-3-319-25010-6_8.

[3] **Anastasia Dimou**, Dimitris Kontokostas, Markus Freudenberg, Ruben Verborgh, Jens Lehmann, Erik Mannens, Sebastian Hellmann, and Rik Van de Walle. Test-driven Assessment of [R2]RML Mappings to Improve Dataset Quality. In *Proceedings of the 14th International Semantic Web Conference: Posters and Demos*, volume 1486 of *CEUR Workshop Proceedings*, October 2015. URL http://ceur-ws.org/Vol-1486/paper_108.pdf.

[4] **Anastasia Dimou**, Dimitris Kontokostas, Markus Freudenberg, Ruben Verborgh, Jens Lehmann, Erik Mannens, and Sebastian Hellmann. DBpedia Mappings Quality Assessment. In Takahiro Kawamura and Heiko Paulheim, editors, *Proceedings of the 15th International Semantic Web Conference: Posters and Demos*, volume 1690 of *CEUR Workshop Proceedings*, October 2016. URL http://ceur-ws.org/Vol-1690/paper97.pdf.

[5] Maribel Acosta, Amrapali Zaveri, Elena Simperl, Dimitris Kontokostas, Sören Auer, and Jens Lehmann. Crowdsourcing linked data quality assessment. In Harith Alani, Lalana Kagal, Achille Fokoue, Paul Groth, Chris Biemann, Josiane Xavier Parreira, Lora Aroyo, Natasha Noy, Chris Welty, and Krzysztof Janowicz, editors, *The Semantic Web – ISWC 2013: 12th International Semantic Web Conference, Sydney, NSW, Australia, October 21-25, 2013, Proceedings, Part II*, pages 260–276, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN 978-3-642-41338-4. doi: 10.1007/978-3-642-41338-4_17. URL http://dx.doi.org/10.1007/978-3-642-41338-4_17.

[6] Riccardo Albertoni and Antoine Isaac. Data Quality Vocabulary. W3C Working Group Note, W3C, December 2016. https://www.w3.org/TR/dwbp/.

[7] Christian Bizer and Richard Cyganiak. Quality-driven information filtering using the {WIQA} policy framework. *Web Semantics: Science, Services and Agents on the World Wide Web*, 7(1):1 – 10, 2009. ISSN 1570-8268. doi: https://doi.org/10.1016/j.websem.2008.02.005. URL http://www.sciencedirect.com/science/article/pii/S157082680800019X. The Semantic Web and Policy.

[8] Barry W. Boehm. *Software Engineering Economics*. Prentice Hall PTR, 1981. ISBN 0138221227.

[9] Thomas Bosch and Kai Eckert. Towards description set profiles for RDF using SPARQL as intermediate language. In *Proceedings of the DCMI International Conference on Dublin Core and Metadata Applications (DC 2014)*. Dublin Core Metadata Initiative, 2014. URL http://dcpapers.dublincore.org/pubs/article/view/3708/1931.

[10] Richard Cyganiak, Chris Bizer, Jörg Garbers, Oliver Maresch, and Christian Becker. The D2RQ Mapping Language. Technical report, W3C, March 2012. http://d2rq.org/d2rq-language.

[11] 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. In *Proceedings of the 5th Workshop on Semantics for Smarter Cities*, volume 1280 of *CEUR Workshop Proceedings*, pages 47–52, October 2014. URL `http://ceur-ws.org/Vol-1280/paper4.pdf`.

[12] Jeremy Debattista, Christoph Lange, and Sören Auer. Representing Dataset Quality Metadata Using Multi-dimensional Views. In *Proceedings of the 10th International Conference on Semantic Systems*, SEM '14, pages 92–99, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2927-9. doi: 10.1145/2660517.2660525. URL `http://doi.acm.org/10.1145/2660517.2660525`.

[13] Bernadette Farias Lóscio, Caroline Burle, and Newton Calegari. Data on the Web Best Practices. W3C Recommendation, W3C, January 2017. `https://www.w3.org/TR/dwbp/`.

[14] Annika Flemming. Quality characteristics of linked data publishing datasources. Master's thesis, Humboldt-Universität of Berlin, 2010.

[15] Christian Fürber and Martin Hepp. Using SPARQL and SPIN for Data Quality Management on the Semantic Web. In Witold Abramowicz and Robert Tolksdorf, editors, *Business Information Systems: 13th International Conference, BIS 2010, Berlin, Germany, May 3-5, 2010. Proceedings*, pages 35–46, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. ISBN 978-3-642-12814-1. doi: 10.1007/978-3-642-12814-1_4. URL `http://dx.doi.org/10.1007/978-3-642-12814-1_4`.

[16] Christian Fürber and Martin Hepp. Using Semantic Web Resources for Data Quality Management. In Philipp Cimiano and H. Sofia Pinto, editors, *Knowledge Engineering and Management by the Masses: 17th International Conference, EKAW 2010, Lisbon, Portugal, October 11-15, 2010. Proceedings*, pages 211–225, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. ISBN 978-3-642-16438-5. doi: 10.1007/978-3-642-16438-5_15. URL `http://dx.doi.org/10.1007/978-3-642-16438-5_15`.

[17] Christophe Guéret, Paul Groth, Claus Stadler, and Jens Lehmann. Assessing linked data mappings using network measures. In Elena Simperl, Philipp Cimiano, Axel Polleres, Oscar Corcho, and Valentina Presutti, editors, *The Semantic Web: Research and Applications: 9th Extended Semantic Web Conference, ESWC 2012, Heraklion, Crete, Greece, May 27-31, 2012. Proceedings*, pages 87–102, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. ISBN 978-3-642-30284-8. doi: 10.1007/978-3-642-30284-8_13. URL `http://dx.doi.org/10.1007/978-3-642-30284-8_13`.

[18] Pieter Heyvaert, **Anastasia Dimou**, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. Semantically Annotating CEUR-WS Workshop Proceedings with RML. In Fabien Gandon, Elena Cabrio, Milan Stankovic, and Antoine Zimmermann, editors, *Semantic Web Evaluation Challenges: Second SemWebEval Challenge at ESWC 2015, Portorož, Slovenia, May 31 - June 4, 2015, Revised Selected Papers*, pages 165–176, Cham, 2015. Springer International Publishing. ISBN 978-3-319-25518-7.

doi: 10.1007/978-3-319-25518-7_14. URL http://dx.doi.org/10.1007/978-3-319-25518-7_14.

[19] Aidan Hogan, Andreas Harth, Alexandre Passant, Stefan Decker, and Axel Polleres. Weaving the Pedantic Web. In *Proceedings of the 3rd Workshop on Linked Data on the Web*, volume 628 of *CEUR Workshop Proceedings*, 2010.

[20] Bernadette Hyland, Ghislain Atemezing, and Boris Villazón-Terrazas. Best Practices for Publishing Linked Data. W3C Working Group Note, W3C, January 2014. http://www.w3.org/TR/ld-bp/.

[21] J.M. Juran and F.M. Gryna. *Juran's Quality Control Handbook*. Industrial engineering series. McGraw-Hill, 1988.

[22] Holger Knublauch and Dimitris Kontokostas. Shapes Constraint Language (SHACL). W3C candidate recommendation, W3C, April 2017.

[23] Dimitris Kontokostas, Martin Brümmer, Sebastian Hellmann, Jens Lehmann, and Lazaros Ioannidis. NLP Data Cleansing Based on Linguistic Ontology Constraints. In Valentina Presutti, Claudia d'Amato, Fabien Gandon, Mathieu d'Aquin, Steffen Staab, and Anna Tordai, editors, *The Semantic Web: Trends and Challenges: 11th International Conference, ESWC 2014, Anissaras, Crete, Greece, May 25-29, 2014. Proceedings*, pages 224–239, Cham, 2014. Springer International Publishing. ISBN 978-3-319-07443-6. doi: 10.1007/978-3-319-07443-6_16. URL http://dx.doi.org/10.1007/978-3-319-07443-6_16.

[24] Dimitris Kontokostas, Patrick Westphal, Sören Auer, Sebastian Hellmann, Jens Lehmann, Roland Cornelissen, and Amrapali Zaveri. Test-driven evaluation of linked data quality. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14, pages 747–758, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2744-2. doi: 10.1145/2566486.2568002. URL http://doi.acm.org/10.1145/2566486.2568002.

[25] Jens Lehmann. Dl-learner: Learning concepts in description logics. *Journal of Machine Learning Research*, 10:2639–2642, December 2009. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=1577069.1755874.

[26] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Kleef, Sören Auer, and Christian Bizer. DBpedia - a Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. *Semantic Web Journal*, 2014. URL http://jens-lehmann.org/files/2014/swj_dbpedia.pdf.

[27] Pablo N. Mendes, Hannes Mühleisen, and Christian Bizer. Sieve: Linked Data Quality Assessment and Fusion. In *Proceedings of the 2012 Joint EDBT/ICDT Workshops*, EDBT-ICDT '12, pages 116–123, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1143-4. doi: 10.1145/2320765.2320803. URL http://doi.acm.org/10.1145/2320765.2320803.

[28] Eric Prud'hommeaux, Jose Emilio Labra Gayo, and Harold Solbrig. Shape Expressions: An RDF Validation and Transformation Language. In *Proceedings of the*

10th International Conference on Semantic Systems, SEMANTICS '14, pages 32–40, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2927-9. doi: 10.1145/2660517.2660523. URL http://doi.acm.org/10.1145/2660517.2660523.

[29] Arthur Ryman. Resource Shape 2.0. Technical report, W3C, February 2014. URL https://www.w3.org/submission/shapes/.

[30] Max Schmachtenberg, Christian Bizer, and Heiko Paulheim. Adoption of the Linked Data Best Practices in Different Topical Domains. In Peter Mika, Tania Tudorache, Abraham Bernstein, Chris Welty, Craig Knoblock, Denny Vrandečić, Paul Groth, Natasha Noy, Krzysztof Janowicz, and Carole Goble, editors, *The Semantic Web – ISWC 2014: 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I*, pages 245–260, Cham, 2014. Springer International Publishing. ISBN 978-3-319-11964-9. doi: 10.1007/978-3-319-11964-9_16. URL http://dx.doi.org/10.1007/978-3-319-11964-9_16.

[31] Amrapali Zaveri, Anisa Rula, Andrea Maurino, Ricardo Pietrobon, Jens Lehmann, and Sören Auer. Quality Assessment for Linked Data: A Survey. *Semantic Web Journal*, 7(1):63–93, 2016. URL http://www.semantic-web-journal.net/content/quality-assessment-linked-data-survey.

Τη επιμελεία πάντα δούλα γίγνεται.
όλα υποτάσσονται με την επιμέλεια.

Αντιφάνης

Everything is subject to diligence

Antifanis

5 Workflow

Complete workflow composition for Linked Data generation with provenance and refinements tracing

In this chapter, we look into workflows which altogether smoothen the generation of high quality Linked Data. To be more precise, we firstly present in detail (i) the complete workflow for Linked Data generation; then we shed more light with respect to (ii) the workflow for refining mapping rules based on the mapping rules and Linked Data quality assessment results, as well as (iii) the workflow for tracking metdata and provenance information when Linked Data generation is executed.

chapter's outline In this chapter, we present the workflow composition that enables smooth Linked Data generation. In the next section, we introduce the Linked Data generation workflow (Section 5.1). In Section 5.2 we present the state of the art with respect to workflow composition for Linked Data generation. In Section 5.3 we present the Linked Data generation workflow, in Section 5.4 the mapping rules refinement workflow, and in Section 5.5 the automated metadata and provenance generation workflow. Last, in Section 5.6 we present the RMLWorkbench that provides a graphical interface to facilitate human agents to administrate their input data and Linked Data generation workflow.

chapter's sources

1. **Anastasia Dimou**, Ruben Verborgh, Miel Vander Sande, Erik Mannens, and Rik Van de Walle. Machine-interpretable Dataset and Service Descriptions for Heterogeneous Data Access and Retrieval. In *Proceedings of the 11th International Conference on Semantic Systems*, pages 145–152, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3462-4. doi: 10.1145/2814864.2814873. URL `http://doi.acm.org/10.1145/2814864.2814873`

2. **Anastasia Dimou**, Dimitris Kontokostas, Markus Freudenberg, Ruben Verborgh, Jens Lehmann, Erik Mannens, Sebastian Hellmann, and Rik Van de Walle. Assessing and Refining Mappings to RDF to Improve Dataset Quality. In Marcelo Arenas, Oscar Corcho, Elena Simperl, Markus Strohmaier, Mathieu d'Aquin, Kavitha Srinivas, Paul Groth, Michel Dumontier, Jeff Heflin, Krishnaprasad Thirunarayan, and Steffen Staab, editors, *The Semantic Web – ISWC 2015*, volume 9367 of *Lecture Notes in Computer Science*, pages 133–149. Springer International Publishing, Oct 2015. doi: 10.1007/978-3-319-25010-6_8. URL `http://link.springer.com/chapter/10.1007/978-3-319-25010-6_8`

3. **Anastasia Dimou**, Tom De Nies, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. Automated Metadata Generation for Linked Data Generation and Publishing Workflows. In Sören Auer, Tim Berners-Lee, Christian Bizer, and Tom Heath, editors, *Proceedings of the 9th Workshop on Linked Data on the Web*, volume 1593 of *CEUR Workshop Proceedings*, April 2016

4. **Anastasia Dimou**, Dimitris Kontokostas, Markus Freudenberg, Ruben Verborgh, Jens Lehmann, Erik Mannens, and Sebastian Hellmann. DBpedia Mappings Quality Assessment. In Takahiro Kawamura and Heiko Paulheim, editors, *Proceedings of the 15th International Semantic Web Conference: Posters and Demos*, volume 1690 of *CEUR Workshop Proceedings*, October 2016. URL `http://ceur-ws.org/Vol-1690/paper97.pdf`

5.1 Introduction

Generating Linked Data from heterogeneous data sources requires dealing with data which can originally (i) reside on *diverse, distributed locations*; (ii) be approached using *different access interfaces*; and (iii) have *heterogeneous structures and formats*:

Diverse, distributed locations
: Data can reside locally, e.g., in files or in a database at the local network, or can be published on the Web.

Different access interfaces
: Data can be approached using diverse interfaces. For instance, it can be as straightforward to access data as raw files. There might be metadata that describe how to access data, as in case of data catalogues. However, a dedicated access interface is required to retrieve data from a repository, such as database connectivity for databases, or different interfaces from the Web, such as Web APIs.

Heterogeneous structures and formats
: Data can be stored and/or retrieved in different structures and formats. For instance, data can originally have a *tabular structure* (e.g., databases or CSV files), be *tree-structured* (e.g., XML or JSON format), or be *semi-structured* (e.g., in HTML).

Even though uniform mapping rules to generate Linked Data from heterogeneous data have been addressed [1], more generic applications still cannot be built because data access and retrieval remains hard-coded. As it was mentioned earlier (Chapter 2), uniform, machine-interpretable mapping rules indicate how triples should be generated in a generic way for all possible different input sources. Those mapping rules contain references to an input data source, which are case-specific and, thus, defined using formulations relevant to the corresponding data format, e.g., XPath for data in XML format. However, as data access and retrieval remains out of mapping rules' scope, it ends up being hard-coded in the corresponding implementations. While this is not a major problem when local, custom, or input-specific data is considered, the situation aggravates when data from multiple heterogeneous data sources, accessed via different interfaces, are required to be retrieved.

Vocabularies which are originally used to advertise datasets or services (e.g., DCAT[1] or Hydra[2]) and to enable applications to easily consume the underlying data exist. These vocabularies can be used to specify how to access and, subsequently, retrieve data, available on the Web or not and generate their semantically enhanced representation. This way, the description that specifies how to access the data becomes machine-interpretable, as the mapping rules also are. However, access descriptions with such vocabularies are not aligned with the vocabularies used to describe the mapping rules so far.

Moreover, data nowadays are published at an increasing rate, and for more and more of them its corresponding semantically enhanced representation is also published and interlinked. However, even though provenance and other metadata become increasingly

[1] DCAT, http://www.w3.org/TR/vocab-dcat/
[2] hydra, http://www.w3.org/ns/hydra/spec/latest/core/

important, most Linked Data sets published in the Linked Data cloud provide no or seldom little metadata. To be more precise, only 37% of the published Linked Data sets provide provenance information or any other metadata [32]. In these rare cases that such metadata is available, it is only manually defined by *human-agents*, rather than produced by *software-agents* involved in the Linked Data generation cycle. Most current solutions which generate or publish Linked Data, do not consider automatically generating the corresponding metadata information, despite the well-defined and W3C recommended vocabularies, e.g., PROV-O [25] or VOID [6], that clearly specify the expected metadata output.

As a consequence, the lack of available metadata information neither allow being aware of the Linked Data origin, nor reproducing its generation outside the context of the application that originally generated it. This occurs because most tools that generate Linked Data derived from heterogeneous data, put the focus on independently providing the corresponding Linked Data, dissociating it from its original source. In the same context, provenance and metadata information regarding the actual mapping rules which specify how the Linked Data is generated from raw data, are not captured at all. Nevertheless, such information might equally influence the assessment of the generated Linked Data trustworthiness.

Similarly, data publishing infrastructures do not automatically publish any provenance or other metadata regarding the Linked Data they host. Instead they would have been expected to enrich the metadata produced while the Linked Data was generated with metadata associated with the publishing activity. Moreover, the Linked Data generation and its publication are considered as interrelated activities that occur together. Although, this is not always the case. Thus, the generated Linked Data and the one subsequently published are not always one and the same. For instance, Linked Data might be generated in subgraphs and published altogether, or generated as a single RDF dataset but published in different RDF graphs. Consequently, their provenance and metadata is not identical.

In a nutshell, capturing (i) original data access and retrieval, (ii) mapping rules definition and refinement, and (iii) provenance and metadata information on every step of the Linked Data publication workflow is not addressed in a systematic and incremental way so far. In this section, we describe our approach for each one of these parts of the Linked Data publication workflow. Then, we present how we align all steps via a user interface for human agents to administrate the entire Linked Data generation and publication.

5.2 State of the Art

In Section 5.2.1, we describe the state of the art wit respect to dataset and service descriptions. In Section 5.2.2, we describe the current Linked Data publishing cycles and the extend to which provenance and other metadata is tracked. In Section 5.2.3, we describe W3C-recommended vocabularies for tracking provenance and other metadata. Last, in Section 5.2.4, we outline alternative approaches for tracing provenance and metadata.

5.2.1 Dataset and Service Descriptions

Different dataset and service descriptions exist, which describe how to access data. Taking advantage of them, we aim to reuse existing vocabularies, which we summarize and discuss in the following paragraphs.

Dataset descriptions may refer to data catalogues or to Linked Data sets. In the former case, the W3C-recommended vocabulary, DCAT [26], is defined. In the later case, the VOID vocabulary [6] is considered, which defines how to describe metadata for Linked Data sets to improve their discoverability. Among the metadata which can be specified with the VOID vocabulary, it is also the *access metadata*. The VOID vocabulary allows to specify as access interface (i) SPARQL endpoints, (ii) RDF data dumps, (ii) root resources, (iv) URI lookup endpoints, and (v) OpenSearch description documents. Dataset descriptions could also refer to a specific type of data, e.g., tabular data. In this context, the CSV on the Web Working Group[3] aims to define a case-specific metadata vocabulary for Tabular data on the Web [34] which, at its current state, only allows data dumps as an access interface.

As far as service descriptions is concerned, and in respect to accessing data in RDF syntax, besides the VOID vocabulary, there is the W3C recommended SPARQL-SD [35]. Regarding database connectivity, there are no dedicated vocabularies. Descriptions in the frame of mapping languages, e.g., D2RQ prevailed so far. However, regarding Web APIs and Services, different vocabularies were defined. Nevertheless, in the case of Web APIs, there is no W3C-recommended vocabulary yet.

The Web Service Description Language (WSDL) [9] describes the possible interactions, messages and the abstract functionality provided by Web services. [20] describes its representation in RDF and in OWL, as well as a mapping procedure for transforming WSDL descriptions into RDF. Semantic Annotations for WSDL (SAWSDL) [21] was one of the first attempts to offer semantic annotations for Web services. Later on, an adaptation for generic HTTP interfaces was proposed [27].

The OWL for Services (OWL-S) [28], the Web Service Modeling Ontology (WSMO) [13] and the WSMO-Lite [37] are alternative ontologies, defined for modelling Web services. The OWL-S ontology also focuses on input and output parameters, as SAWSDL. The WSMO ontology is an alternative to OWL-S, although there are substantial differences between the two approaches [24]. The WSMO ontology employs a single family of layered logic languages [14]. However, when expressed in RDF syntax, WSMO expressions become

[3] CSVW, URL http://www.w3.org/2013/csvw/wiki/Main_Page

similarly unintegrated and hence not self-descriptive as OWL-S expressions. The WSMO-Lite ontology extends SAWSDL with conditions and effects. hRESTS [22] uses microformats to add machine-processable information to human-readable documentation, while its ontology[4] extends the WSMO-Lite ontology[5].

Last, MicroWSMO extends hRESTS and adopts the WSMO-Lite service ontology for expressing concrete semantics. For our purposes, we mostly need the interface description part. as our goal is access to the services rather than, for instance, composition.

5.2.2 Linked Data publishing cycle

In the Linked Data publishing workflow there are different activities taking place. Among them, the definition of the rules to generate Linked Data from raw data, its actual generation, its publishing and its interlinking are few of the most essential steps. However, the majority of the tools developed to address these tasks do not generate automatically any provenance or metadata information as the corresponding tasks are accomplished, let alone enriching metadata defined in prior steps of the Linked Data publishing workflow.

Hartig and Zhao [16] argued the need of integrating provenance information publication in the Linked Data publishing workflow. However, they only focused on its last step, namely the Linked Data publication, outlining metadata publication approaches and showcasing on well-know Linked Data publishing tools, such as Pubby[6] and Triplify[7].

None of the well-known systems that generate Linked Data from any type of (semi-)structured data provide any provenance or metadata information in conjunction with the generated Linked Data, to the best of our knowledge. The main obstacle, at least with respect to provenance, is that it is hard to specify where the data originally resides. That occurs because most of these tools, consider a file as data input. However, where the data of this file is derived from is not known and, therefore, the corresponding provenance annotations can not be accurately defined in an automated fashion.

The D2R server[8] and the CSV2RDF4LOD[9] are the only tools that generate provenance and metadata information in conjunction with the Linked Data. However, the D2R server refers only to data in relational databases, it supports a custom provenance vocabulary, not the W3C-recommended PROV-O [25], and is limited to dataset high level metadata information. The CSV2RDF4LOD refers only to CSV files and it achieves capturing provenance using custom bash scripts that aim to keep track of the commands used. The situation aggravates in the case of custom solutions for generating RDF data which neglect to include in its development cycle mechanisms to generate provenance and metadata information.

With the advent of mapping languages, such as the D2RQ[10], SML[11], or the W3C recom-

[4] http://www.wsmo.org/ns/hrests/
[5] http://www.wsmo.org/ns/wsmo-lite/
[6] Pubby, http://wifo5-03.informatik.uni-mannheim.de/pubby/
[7] Triplify, http://triplify.org/
[8] D2R, http://d2rq.org/
[9] CSV2RDF4LOD, https://github.com/timrdf/csv2rdf4lod-automation/wiki
[10] D2RQ, http://d2rq.org/d2rq-language
[11] SML, http://sml.aksw.org/

mended R2RML [12], the mapping rules that specify how Linked Data is generated from raw data, were decoupled from the source code of the corresponding tools that execute them. However, mapping languages are explicitly focused on specifying the mapping rules, neglecting to provide the means to specify the data source too. Whereas, for instance the D2RQ language allows to specify the relational database where the data is derived from, other languages, including R2RML, do not, considering it out of the language's scope.

In the same context, tools were developed to support human agents to semantically annotate their data. However, those tools still generate both the mapping rules and the corresponding Linked Data after the rules execution, without providing any provenance or metadata information. None of the tools that automatically generate mapping rules of relational databases to Linked Data, such as BootOx [19], IncMap [31], or Mirror [15], or supporting users in defining mapping rules, e.g., FluidOps editor [33], supports automated provenance and metadata information generation, neither for the mapping rules, nor for the generated Linked Data. Specifying metadata for the mapping rules or considering the mapping rules to determine the provenance and metadata becomes even more cumbersome, in particular in the case of a mapping language whose representation is not in RDF, e.g., SML, SPARQL, or XQuery.

Similarly, among the RDF data publishing infrastructures, only Triple Pattern Fragments[12] (TPF) [36] provide some metadata information, mainly regarding dataset level statistics, and access. Virtuso[13], 4store[14] and other pioneer publishing infrastructures do not provide *out-of-the-box* metadata information, e.g., provenance, dataset-level statics etc. of the published Linked Data. LODLaundromat[15] is the only Linked Data publishing infrastructure that provides automatically generated metadata. However, it uses its own custom ontology[16] which partially relies on the PROV-O ontology to provide metadata information.

5.2.3 Provenance and Metadata Vocabularies

W3C recommended vocabularies were defined for Linked Data provenance and metadata:

5.2.3.1 PROV Ontology

The PROV ontology (PROV-O) [25] is recommended by W3C to express the PROV Data Model [29] using the OWL2 Web Ontology Language (OWL2) [18]. PROV-O can be used to represent provenance information generated in different systems and under different contexts. According to the PROV ontology, a *prov:Entity* is a physical, digital, conceptual, or other kind of thing. A *prov:Activity* occurs over a period of time and acts upon or with entities; it may include consuming, processing, transforming, modifying, relocating, using, or generating entities. A *prov:Agent* bears some form of responsibility for an activity taking place, for the existence of an entity, or for another agent's activity.

[12] TPF, URL http://linkeddatafragments.org/
[13] Virtuoso, URL http://virtuoso.openlinksw.com/dataspace/doc/dav/wiki/Main/
[14] 4store, URL http://4store.org/
[15] LODLaundromat, URL http://lodlaundromat.org/
[16] `http://lodlaundromat.org/ontology/`

5.2.3.2 VoID Vocabulary

The Vocabulary of Interlinked Datasets (VOID) [6] is a vocabulary for expressing metadata about RDF datasets with applications ranging from data discovery to cataloging and archiving. VOID expresses (i) *general*, (ii) *access* and (iii) *structural* metadata, as well as links between datasets. *General metadata* is based on Dublin Core. *Access metadata* describes how the RDF data can be accessed using different protocols. *Structural metadata* describes the structure and schema of the RDF data. According to the VOID vocabulary, a *void:Dataset* is a set of RDF triples maintained or aggregated by a single provider. A *void:Dataset* is a meaningful collection of triples, that deal with a certain topic, originate from a certain source or process, and contains a sufficient number of triples that there is benefit in providing a concise summary. The concrete RDF triples contained in a *void:Dataset* are established through access information, such as the address of a SPARQL endpoint. Last, a *void:Linkset* is a collection of RDF links whose subject and object are described in different RDF datasets.

5.2.3.3 DCAT Vocabulary

The Data Catalog Vocabulary (DCAT) [26] is designed to facilitate interoperability between data catalogs published on the Web. It aims to (i) increase data discoverability, (ii) enable applications to easily consume metadata from multiple catalogs, (iii) enable decentralized catalogs publishing, and (iv) facilitate federated dataset search. According to the DCAT vocabulary, a *dcat:Catalog* represents a dataset catalog, a *dcat:Dataset* represents a dataset in the catalog, whereas a *dcat:Distribution* represents an accessible form of a dataset, for instance a downloadable file, an RSS feed, or a Web service that provides the data. DCAT considers as a dataset a collection of data, published or curated by a single agent, and available for access or download in one or more formats. This data is considered for generating an RDF dataset. Thus, the generated RDF dataset forms a *dcat:Distribution* of a certain *dcat:Dataset*.

5.2.4 Approaches for tracing PROV & metadata

We outline methods for capturing provenance and other metadata information. We identify two approaches that capture provenance and other metadata information inline with the rest RDF data –*Explicit Graphs* (Section 5.2.4.3) and *Singleton Properties* (Section 5.2.4.2)– and two that trace them independently of the RDF data –RDF *Reification* (Section 5.2.4.1) and *Implicit Graphs*(Section 5.2.4.4). In the following subsections, we discuss in more detail alternative approaches for defining the provenance of the following RDF triple:

```
1  ex:item10245 ex:weight "2.4"^^xsd:decimal .
```

5.2.4.1 RDF Reification

The RDF framework considers a vocabulary for describing RDF statements and providing additional information. RDF reification is intended for expressing properties such as dates of composition and source information, applied to specific instances of RDF triples. The conventional use involves describing an RDF triple using four statements. A description of a statement is called a *reification* of the statement. The RDF reification vocabulary consists of the type `rdf:Statement`, and the properties `rdf:subject`, `rdf:predicate` and `rdf:object`. RDF reification is the W3C recommended approach for representing provenance and metadata information.

```
1  _:ex12345  rdf:type        rdf:Statement .
2  _:ex12345  rdf:subject     ex:item10245 .
3  _:ex12345  rdf:predicate   ex:weight .
4  _:ex12345  rdf:object      "2.4"^^xsd:decimal .
5  _:ex12345  prov:wasDerivedFrom _:src123 .
```

The major disadvantage of RDF reification is the number of RDF triples required to represent a reified statement. For each generated RDF triple, at least four additional statements is required to be generated. So, for an RDF dataset of N triples, the metadata graph will be equal to four times the number of the RDF dataset triples in the best case where only the RDF reification statements are generated and no additional.

5.2.4.2 Singleton Properties

Singleton properties [30] is an alternative approach for representing statements about statements using RDF. This approach relies on the intuition that the nature of every relationship is universally unique and can be a key for any statement using a singleton property. A *singleton property* represents one specific relationship between two entities under a certain context. It is assigned a URI, as any other property, and can be considered as a subproperty or an instance of a generic property. Singleton properties and their generic property are associated with each other using the `singletonPropertyOf` property, subproperty of `rdf:type`.

```
1  ex:item10245   ex:weight#1              "2.4"^^xsd:decimal .
2  ex:weight#1    sp:singletonPropertyOf   ex:weigh .
3  ex:weight#1    prov:wasDerivedFrom      _:src123 .
```

5.2.4.3 Explicit Graphs

The *Explicit Graphs* approach relies on *named graphs*. *Named Graphs* is a set of RDF triples named by a URI and can be represented using *TriG* [7], *N-Quads* [8], or *JSON-LD* [10], but it is not compatible with all RDF serialisations. This approach is similar to *Singleton Properties*. Instead of annotating the common predicate of the triples, the context of the triple is annotated. This way, introducing one triple per predicate is avoided. However, the *Explicit Graphs* approach has two drawbacks: (i) they are not supported by all RDF serializations; and (ii) they might be in conflict with the named graph defined as part of the RDF dataset and whose intent is different than tracing provenance information.

```
1  ex:item10245   ex:weight  "2.4"^^xsd:decimal    ex:graph .
2  ex:graph       prov:wasDerivedFrom    _:src123 .
```

5.2.4.4 Implicit Graphs

Implicit graphs are URIs assigned implicitly to a dataset, graph triple, or term. An *Implicit Graph* is aware of what it represents but the represented entity is not directly linked to its implicit graph. Implicit graphs might be used to identify a dataset or a graph, but also triples. In the later case, as Triple Pattern Fragments (TPF) introduced [36], each triple can be found by using the elements of itself, thus, each triple has a URI and, thereby, its implicit graph. For example, the triple `x y z` for a certain dataset could be identified by the TPF URI `http://example.org/dataset?subject=x&predicate=y&object=z`.

```
1  <http://example.org/dataset?
2  subject=ex:item10245&predicate=ex:weight&object="2.4">
3      prov:wasDerivedFrom _:src123 .
```

5.2.5 Discussion

Based on the aforementioned, despite the existence of dedicated vocabularies and alternative approaches for tracing provenance and metadata, complete Linked Data generation workflows cannot be built because the different components of the workflow remain fragmented and generate or publish Linked Data neglecting to trace provenance and other metadata, let alone validating the quality of the Linked Data they generate or publish.

5.3 Generation Workflow

In this section, we describe the workflow which we proposed to access and retrieve the data, whose semantically enhanced representation is desired. Figure 5.1 illustrates how data is accessed and retrieved from their original repositories and how further data fractions are extracted to be used to finally obtain the desired Linked Data.

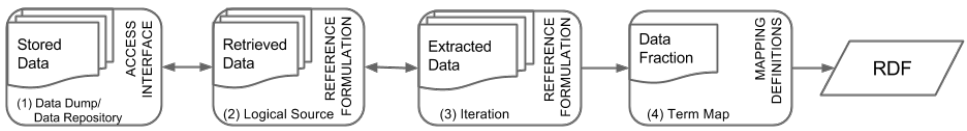


Figure 5.1: *Data retrieval and Linked Data generation. A triple consists of RDF Terms which are generated by Term Maps (4). Those Term Maps are defined with mapping rules and are instantiated with data fractions referred to using a reference formulation relevant to the corresponding data format. Those fractions are derived from data extracted at a certain iteration (3) from a logical source (2). Such a logical source is formed by data retrieved from a repository (1) which is accessed as defined using the corresponding dataset or service description vocabulary.*

Data is stored in different repositories often residing in different locations. Those repositories can be found, e.g., locally, on a network, or on the Web. For instance, data can be available as raw files, databases, or Web sources, or files listed in catalogues. To retrieve data from a repository, an *access interface* is required (Step 1). Data can be approached using diverse interfaces, for instance, database connectivity, such as Open DataBase Connectivity (ODBC) to access data residing in a database. However, data on the Web can also be retrieved using different interfaces, such as Web APIs.

Once the *retrieved data* is obtained (Step 2), from one or more repositories, one or more *Logical Sources* are formed. Such a *Logical Source* contains data in a certain structure and format, e.g., CSV, XML, or JSON. Further data fragmentation and extraction requires references relevant to the data format (i.e., its corresponding *Reference Formulation*).

As *mapping rules* are meant to be applied recursively to data fragments extracted from a *Logical Source*, an *iterator* is required. The iteration pattern is also defined in a formulation relevant to the *Logical Source*. The *iterator* runs over the *Logical Source*, extracting data fragments (Step 3). For instance, an *iterator* running over a CSV file extracts a row of the CSV at each iteration. In case the iteration pattern applies to the complete *Logical Source*, the *Iteration* fragment coincides with the *Logical Source*.

For each *Iteration* further data fragmentation occurs (Step 4) to extract the exact *Data fraction(s)* used to instantiate a *Term Map* which, in its turn, generates the corresponding RDF term. For the aforementioned CSV example, such a data fraction is the value of a column from a given row extracted at a certain *Iteration*. At the end, the corresponding RDF representation of the *Logical Source* is obtained (Step 5).

RML is focused on the rules that define how to generate Linked Data from raw data, but not where this data originally reside or how this data is accessed and retrieved, in the same way as it remains out of scope for the R2RML specification how the SQL connection is established [12]. Hence, even though uniform, machine-interpretable mapping rules and case specific references to the input data source, depending on its format, are addressed with RML, data access and retrieval remains hard-coded in the implementation.

However, obtaining a raw data semantic representation requires dealing with data which can originally (i) reside on diverse, distributed locations (data can reside locally, e.g., in files or in a database at the local network, or can be published on the Web); and (ii) be approached using different access interfaces. For instance, it can be straightforward to access the data as raw files. However, it might also be required to have a dedicated access interface to retrieve the data from a repository, such as database connectivity for databases, or different interfaces from the Web, such as Web APIs.

Therefore, in this section, we define how heterogeneous data and service descriptions can be considered to access and retrieve data and how such descriptions are aligned with RML. Detailed documentation regarding access interfaces aligned with RML and supported by the RMLMapper is available at `http://rml.io/RMLdataRetrieval`.

5.3.1 Access Interfaces

Accessing data requires different dedicated interfaces. In any case, corresponding vocabularies can describe how to access the underlying data. Such vocabularies refer to:

Database connectivity For the description of database connectivity, corresponding descriptions from the D2RQ mapping language [11] can be considered for accessing databases with the JDBC framework. The D2RQ vocabulary contains the following term: `d2rq:Database` to represent a JDBC connection to a local or remote relational database. Instances of `d2rq:Database`, annotated with its properties to specify the JDBC connection properties, can be used to describe the access to a database (Listing 5.1, Line 3).

```
1  @PREFIX d2rq: <http://www.wiwiss.fu-berlin.de/suhl/bizer/D2RQ/0.1#>.
2
3  <#DB_source>
4    a d2rq:Database;
5    d2rq:jdbcDSN "jdbc:mysql://localhost/example";
6    d2rq:jdbcDriver "com.mysql.jdbc.Driver";
7    d2rq:username "user";
8    d2rq:password "password".
```

Listing 5.1: *D2RQ database connectivity description*

The D2RQ database connectivity description is focused on databases with JDBC-based connectivity and serves the needs of an exemplary case of this work. Other vocabularies describing the JDBC framework or vocabularies for other interfaces of database connectivity can be considered as well.

Metadata for the Access Interface Data can be published either independently, as data dumps, or in the frame of a data catalogue. The DCAT vocabulary[17] [26] provides machine-readable metadata that enables applications to easily consume them, without making any assumptions about the dataset's format. The DCAT namespace is `http://www.w3.org/ns/dcat#` and the preferred prefix is `dcat`.

The DCAT vocabulary defines the following terms: (i) `dcat:Catalog` to represent a dataset catalog; (ii) `dcat:Dataset` to represent a dataset in the catalog, i.e., a collection of data, published or curated by a single agent, and available for access or download in one or more formats; and (iii) `dcat:Distribution` to represent an accessible form of a dataset, e.g., a downloadable file, an RSS feed, or a Web service that provides the data. Thus, a certain distribution is the minimum information that a mapping processor requires to retrieve some data. Directly downloadable distributions contain a `dcat:downloadURL` reference (Listing 5.2, Line 7).

```
1  @prefix dcat: <http://www.w3.org/ns/dcat#>.
2
3  <#DCAT_source>
4    a dcat:Dataset;
5    dcat:distribution [
6      a dcat:Distribution;
7      dcat:downloadURL <http://example.org/file.xml>].
```

Listing 5.2: *DCAT access metadata description*

[17] DCAT, `http://www.w3.org/ns/dcat#`

A dcat:Distribution might not be directly downloadable though. For instance, a dataset might be available via an API which, on its own turn, can be defined as an instance of a dcat:Distribution. In this case, it is recommended to use dcat:accessURL, instead of dcat:downloadURL. However, access-specific properties, e.g., for API descriptions, are not defined by DCAT itself. Thus, a client does not know how to interact with the specified interface, the API in this case. Due to its shortcoming to entirely describe indirectly accessed Web sources, other vocabularies focused on describing specific interfaces could be considered instead.

Hypermedia-driven Web APIs For the description of hypermedia-driven Web APIs, the Hydra Core Vocabulary [23] is published by the Hydra W3C Community Group[18]. It is a lightweight vocabulary to specify concepts commonly used in Web APIs, which enable a server to advertise valid state transitions to a client. The server is decoupled from the client which can use this information to construct valid HTTP requests to retrieve data. The Hydra namespace is http://www.w3.org/ns/hydra/core# and the preferred prefix hydra. The Hydra vocabulary includes the following terms:

(i) hydra:ApiDocumentation to represent a Web API, by providing its title, short description, main entry point and additional information about status codes that might be returned. The Hydra vocabulary enables the API's main entry point to be discovered automatically, when it is not known or specified, if the API publisher marks the response with a special HTTP link header. A client looks for a link header with a relation type hydra:apiDocumentation and, this way, obtains the API's main entry point using a hydra:ApiDocumentation.

(ii) hydra:IriTemplateMapping to represent an instance whose values depend on information only known by the client, a template-valued IRI with variables. The Hydra vocabulary can be used to describe both static and dynamically (Listing 5.3, Line 4) generated IRIs. The dcat:accessUrl of a dcat:Distribution instance can point to a resource described with the Hydra vocabulary, to inform potential agents how to perform valid HTTP requests.

```
1  @prefix hydra: <http://www.w3.org/ns/hydra/core#>.
2
3  <#API_source> a hydra:IriTemplate
4    hydra:template "https://api.twitter.com/1.1/followers/ids.json?screen_name={name}";
5    hydra:mapping [
6      a hydra:IriTemplateMapping;
7      hydra:variable "name";
8      hydra:required "true" ].
```

Listing 5.3: *Template-valued Web API description*

(iii) hydra:PagedCollection to represent information regarding a collection of data into multiple pages, e.g., total number of items, number of items per page, and first, next and last page.

```
1  @prefix hydra: <http://www.w3.org/ns/hydra/core#> .
2  <#APIpaged_source> a hydra:PagedCollection;
3    hydra:apiDocumentation <#HydraDocumentation>;
4    hydra:itemsPerPage "100";
5    hydra:firstPage "/comments?page=1";
6    hydra:lastPage "/comments?page=10" .
```

Listing 5.4: *Hydra Paged Collection description*

[18] Hydra, http://www.w3.org/community/hydra/

SPARQL Services For SPARQL endpoints description, the W3C recommends the SPARQL Service Description vocabulary (SPARQL-SD) [35]. SPARQL-SD provides a list of features of a SPARQL service and their descriptions, made available via the SPARQL 1.1 protocol for RDF. The SPARQL-SD namespace is `http://www.w3.org/ns/sparql-service-description#` and the preferred prefix is `sd`.

The SPARQL-SD vocabulary includes the following terms: (i) `sd:Service` to represent a SPARQL service which is made available via the SPARQL protocol (Listing 5.5, Line 3); (ii) `sd:Dataset` to represent the default Linked Data set that a `sd:Service` refers to; (iii) `sd:Graph` to represent an RDF dataset's default graph; (iv) `sd:NamedGraph` to represent a named graph (each RDF dataset might have zero or more); (v) `sd:GraphCollection` to represent a collection of graphs; and (vi) `sd:Language` to represent the SPARQL language (e.g., `sd:SPARQL11Query` or `sd:SPARQL11Update`, Listing 5.5, Line 6).

```
1  @prefix sd: <http://www.w3.org/ns/sparql-service-description#>.
2
3  <#SPARQL_source>
4    a sd:Service;
5    sd:endpoint <http://dbpedia.org/sparql/>;
6    sd:supportedLanguage sd:SPARQL11Query;
7    sd:resultFormat <http://www.w3.org/ns/formats/SPARQL_Results_XML>.
```

Listing 5.5: *SPARQL Service Description*

Similarly to `hydra:IriTemplate`, a `sd:Service` instance could be used to clarify how to access a dataset via the SPARQL-SD interface (`dcat:accessUrl`), allowing potential agents to know how to perform the corresponding HTTP requests. In the same way, `void:endpoint` could be considered in the case of datasets published with metadata described using the W3C-recommended VOID vocabulary [6].

5.3.2 Data Source

The input data is specified within the Logical Source, but not how to access and retrieve this data. The Logical Source consists of some data without further specifying how to retrieve this data. Nevertheless, the access descriptions, that advertise services or datasets, can be the Triples Map's Source (*rml:source*). For instance, the Logical Source, specified at Listing 2.6 for the <#Person> Triples Map, could have been published on a data catalogue and, thus, it is an instance of *dcat:Distribution*. The corresponding description then is:

```
1  @prefix rml:    <http://semweb.mmlab.be/ns/rml#>.
2  @prefix dcat:   <http://www.w3.org/ns/dcat#> .
3
4  <#InputX>
5    rml:source [ a dcat:Distribution;
6      dcat:downloadURL <http://ex.com/file.csv> ];
7    rml:referenceFormulation ql:CSV .
```

Listing 5.6: *Data dump in catalogue as Input Source*

For the other Triples Map, <#TwitterAccount>, the data might be derived from a user's twitter account, and could have been stored locally in a file retrieved at some point from the Twitter API, or the request could have been hard-coded in the implementation. Nevertheless, the required request could have just been described using the Hydra vocabulary

or could have been provided using directly the resource advertising the API. In the aforementioned example of Listing 2.6, the Logical Source for the <#TwitterAccount> Triples Map could have been described as follows:

```
1  @prefix hydra: <http://www.w3.org/ns/hydra/core#>.
2
3  <#InputY> a hydra:IriTemplate
4    hydra:template "https://api.twitter.com/1.1/followers/ids.json?screen_name={name}";
5    hydra:mapping [
6      hydra:variable "name";
7          hydra:required "true" ].
```

Listing 5.7: *Web API as Input Source*

5.3.2.1 RML Referencing Object Map and heterogeneous data retrieval

Using data formed by instantiating an access description, is straightforward in most cases. Dataset and service descriptions either are derived from agents who are *data owners/publishers* or explicitly defined by agents who are *data consumers*. If the service provides access metadata, an agent can just point to the resource that describes them. If not, the minimum required information for each access interface should be defined. Linked Data generation processors consider data access and service descriptions to retrieve data and instantiate *Logical Sources*. The access description might be static or dynamic. If dynamically created, it is often required to instantiate a template, e.g., a URI template or a SQL/SPARQL query template. The values to instantiate the template might be provided by an agent who generates Linked Data or the variables might be replaced with values derived from another input source, as it occurs in the case of *Referencing Object Maps*.

In case the access interface is described by a *data consumer*, its description prevails over the one provided by the *data owner/publisher*. For instance, if data is retrieved from a *hydra:PagedCollection*, the *data consumer* might define that two hundred items per page should be returned and six pages should be taken into consideration. If ten pages of data are retrieved, only the six of them should be considered for Linked Data generation. If the *data consumer* does not specify the pages, all of them will be considered.

A Logical Source extends R2RML's Logical Table which determines a database's table, using the Table Name. Nevertheless, how the connection to the database is achieved is not specified at the R2RML specification, since it remains out of its scope. Moreover, R2RML is specific for relational databases (SQL), while a D2RQ description may refer to other databases using the JDBC framework too, or in general, any other database with its own access description. In case of an SQL query against the table *DEPT* of a database for instance, the R2RML *Logical Table* (Listing 5.8).

```
1  [ ] rr:logicalTable [ rr:sqlQuery """
2    SELECT DEPTNO, DNAME, LOC,
3    (SELECT COUNT(*) FROM EMP WHERE EMP.DEPTNO=DEPT.DEPTNO)
4    AS STAFF FROM DEPT; """ ].
```

Listing 5.8: *R2RML Logical Table*

However, if a database is specified, the *Logical Table* should be superseded by its broader *Logical Source* and the corresponding database connectivity description should be provided (Listing 5.9).

```
1  [ ] rml:logicalSource [
2    rml:query """
3      SELECT DEPTNO, DNAME, LOC,
4      (SELECT COUNT(*) FROM EMP WHERE EMP.DEPTNO=DEPT.DEPTNO)
5      AS STAFF FROM DEPT; """ ;
6    rml:source <#DB_source> ].
```

Listing 5.9: *RML Logical Source for Database Input*

5.3.3 Binding Condition

A Referencing Object Map (see Listing 5.10,line 8) might have a *template-valued* data source (line 15) as input that requires one or more values to be filled with. The Binding Condition then specifies how the Logical Source of the Referencing Object Map is instantiated either with value(s) retrieved from the input source, specified with an rml:reference or a rr:template, or with a constant value, specified with a rr:constant. In the first case, a reference (line 10) that exists in the Logical Source of the Triples Map that contains the Referencing Object Map is provided. In the later case, a constant value can be provided. The value is bind to a variable that is specified with crml:variable. If the Referencing Object Map's Logical Source has more than one variables required, an equal number of Binding Conditions is expected. Detailed documentation regarding the *RML Binding Condition* is available at `http://rml.io/RMLconditions`.

```
1  @prefix rr:     <http://www.w3.org/ns/r2rml#>.
2  @prefix rml:    <http://semweb.mmlab.be/ns/rml#>.
3  @prefix crml:   <http://semweb.mmlab.be/ns/crml#>.
4
5  <#Person>
6    rr:predicateObjectMap [ rr:predicate foaf:account;
7      rr:objectMap [
8        rr:parentTriplesMap <#TwitterAcount> ;
9        crml:binding [
10         rml:reference "id" ;
11         crml:variable "name" ]]].
12
13 <#InputY> rml:source [
14   a hydra:IriTemplate
15   hydra:template "https://api.twitter.com/1.1/followers/ids.json?screen_name={name}";
16   hydra:mapping [
17     hydra:variable "name";
18     hydra:required "true" ] ];
19   rml:referenceFormulation ql:JSONPath.
```

Listing 5.10: *RML Binding Condition*

5.4 Refinement Workflow

In the previous sections, we covered the part of the broader Linked Data generation workflow which is related to data access and retrieval. Now we look into more detail with respect to the workflow for mapping rules refinement.

A *uniform, iterative, incremental assessment and refinement workflow* (Fig. 5.2) allows refining the mapping rules and, thus, produces high quality Linked Data in the end.

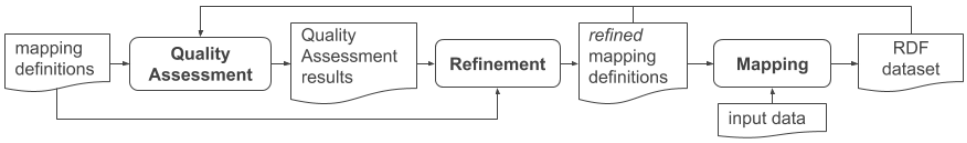


Figure 5.2: *Quality Assessment enabled Linked Data mapping and publishing workflow*

Its steps are explained below and presented in Fig. 5.2:

1. The semantic schema, as the mapping rules state, is assessed against certain quality assessment metrics, as it would have been done if it was the actual Linked Data.
2. The Quality Assessment report lists each violation which was identified.
3. The Mapping Quality Assessment (MQA) results are used to refine the mapping rules. This step can be repeated until a set of mapping rules without violations is generated, or the mapping rules cannot be further refined.
4. A refined version of the mapping rules is generated and used to execute the mapping of data, or a sample of the data.
5. The generated output (Linked Data) is assessed, using the same quality assessment framework. The Linked Data set –and optionally the Mapping– Quality Assessment (DQA) can be repeated until an ameliorated set of mapping rules is generated.
6. When the mapping rules are finalized, the actual mapping is performed and the Linked Data set is generated exempt of violations to the greatest possible extent.

5.5 Metadata and Provenance

In the previous sections, we covered the details on how data is accessed and retrieved, and how mapping rules are refined. In this section, we look into the details of the complete Linked data generation and publication workflow and in particular on how keeping track of provenance and other metadata. Provenance and other metadata can be captured at different steps of the Linked Data generation workflow. Keeping track of metadata derived from different steps of Linked Data generation, results in more complete information regarding how a Linked Data set was generated and formed in the end.

We consider each workflow step as an activity (`prov:Activity`) whose properties are needed to be traced. In Table 5.1, we summarize those activities and the information that needs to be defined each time. The provenance and how the different steps are associated with each other are shown at Figure 5.3. In Table 5.2, we summarize the metadata which is captured for each step of the Linked Data generation workflow.

		Same Dataset		Different Dataset		
	Map.	**Gen.**	**Pub.**	**Gen.**	**Pub.**	**Link.**
prov:Entity						
prov:wasGeneratedBy	●	●		●	●	●
prov:wasDerivedFrom	●	●		●	●	●
prov:wasAttributedTo		●	●	●	●	●
prov:Agent	●	◐	◐	●	●	●
prov:actedOnBehalfOf	●	●	●	●	●	●

A filled circle (●) indicates that the property should be (re-)assessed in *each* marked step. A half-filled circle (◐) indicates that that property can be assessed in *any* of marked step.

Table 5.1: *Table of properties required for each entity*

		Same Dataset		Different Dataset		
	Map.	**Gen.**	**Pub.**	**Gen.**	**Pub.**	**Link.**
prov:Entity						
void:Dataset – General						
dcterms:creator	●	●		●		●
dcterms:contributor	●	●	●	●	●	●
dcterms:publisher					●	
dcterms:source		●			●	
dcterms:created	●	●		●	●	
dcterms:modified	●	●		●	●	
dcterms:issued	●	◐	◐	●	●	
dcterms:license	●	◐	◐	●	●	●
void:feature	●	◐	◐	●	●	●
void:Dataset – Access			●			●
void:sparqlEndpoint			●			●
void:dataDump			●			●
void:uriLookupEndpoint			●			●
void:openSearchDescription			●			●
void:Dataset – Structural	●	◐	◐	●	●	●
void:exampleResource		◐	◐			
void:uriSpace		◐	◐			
void:vocabulary		◐	◐			
void:subset		◐	◐			
void:classPartition		◐	◐			
void:propertyPartition		◐	◐			
void:Dataset – Statistics	●	◐	◐	●	●	●
void:triples		◐	◐	●	●	●
void:entities		◐	◐	●	●	●
void:classes		◐	◐	●	●	●
void:properties		◐	◐	●	●	●
void:distinctSubjects		◐	◐	●	●	●
void:distinctObjects		◐	◐	●	●	●
void:documents		◐	◐	●	●	●
void:Linkset	●	◐	◐	●	●	●
void:target		◐	◐	●	●	●
void:linkPredicate		◐	◐	●	●	●
void:inDataset		◐	◐	●	●	●

A filled circle (●) indicates that the property should be (re-)assessed in *each* marked step.
A half-filled circle (◐) indicates that that property can be assessed in *any* of marked step.

Table 5.2: *Table of properties required for each entity*

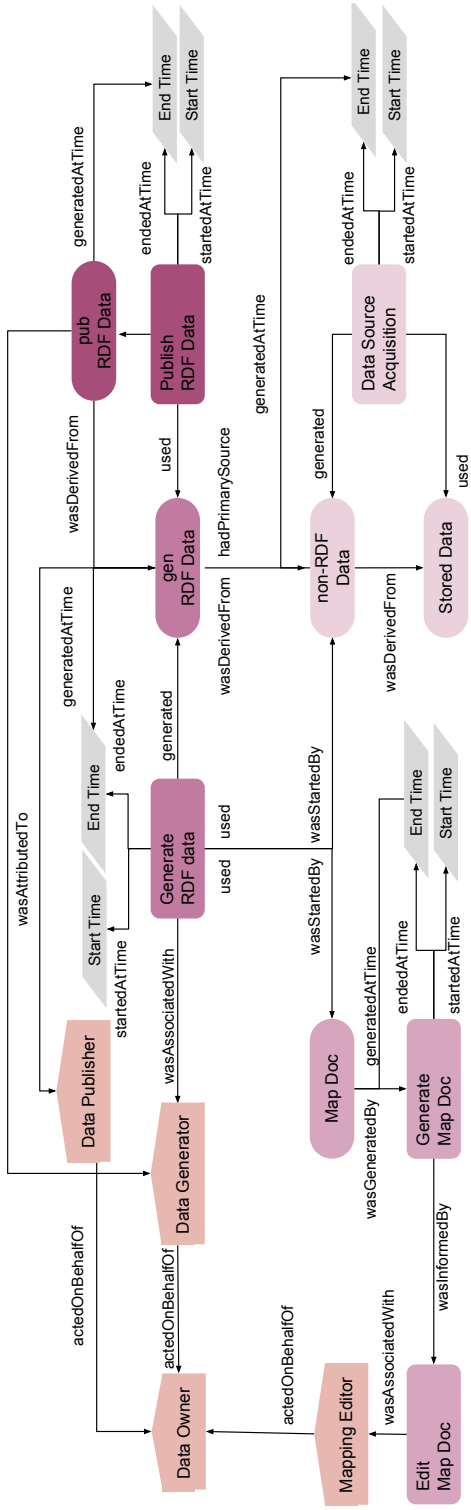


Figure 5.3: *A coordinated view of Linked Data publishing workflow activities.*

5.5.1 High Level Workflow

We identify the following primary steps: mapping rules generation, data source retrieval, Linked Data generation, and Linked Data publication.

mapping rules definition Provenance and metadata is required to be captured when mapping rules are defined (Figure 5.3, Edit Map Doc). In this case, it is important to track *when* the mapping rules were edited or modified and by *whom*. A Linked Data set might have been generated using multiple mapping rules whose definition occurred at different moments and by different agents. Consequently, the generation of certain mapping rules (Figure 5.3, *Generate Map Doc*) is an activity (`prov:Activity`) which is informed by all prior editing activities (Figure 5.3, *Edit Map Doc*). For instance, a mapping rule might have been generated by a mapping generator or edited by a human-agent using a mapping editor or even manually. However, such mapping rules might have been modified or used in conjunction with other mapping rules which were generated, on their own turn, by another human-agent at a different time.

Agents who define mapping rules (Figure 5.3, *Mapping Editor*) might differ from the ones who generate (Figure 5.3, *Data Generator*) or publish the data (Figure 5.3, *Data Publisher*), or even the owner of the data (Figure 5.3, *Data Owner*). Being aware of who defined the mapping rules is of crucial importance to assess the trustworthiness of the generated Linked Data. For instance, Linked Data generated using mapping rules from an automated generator might be considered less trustworthy compared to Linked Data whose mapping rules were defined by a data specialist.

data sources retrieval Linked Data might be derived from one or more heterogeneous data sources (Figure 5.3, *Data Source Acquisition*). Each data source, on its own turn, might be derived from certain data, as explained in Section 5.3. For instance, a table might be derived from a database or some JSON data might be derived from a Web API. Such a data source might be turned into Linked Data partially or in its entirety. This means that not the entirely stored data is retrieved but a selection is only used to generate the desired Linked Data. For instance, only data that fulfills an SQL query could be retrieved, instead of the entire table or database.

For this activity, it is important to keep track of metadata regarding the data sources which were used and their retrieval, as this indicates the original data sources of the generated Linked Data. As the original data might change over time, it is important to be able to update the corresponding Linked Data. For instance, in case of an API, some data is retrieved at a certain time, but different data might be retrieved at a subsequent time. Thus, it is crucial to know when the data is accessed to assess its timeliness with the original data. For instance, comparing the last modified date of the original data and the generation date of the Linked Data, indicates if the available semantic representation is aligned with the current version of the original data.

Linked Data generation As soon as the mapping rules and the data source are available, the Linked Data is generated (Figure 5.3, *Generate RDF Data*). For this activity, it is important to keep track of (i) *how* the Linked Data generation was triggered, i.e., *data-driven* or *mapping-driven*, from raw data (Linked Data generation) or from

Linked Data (Linked Data interlinking); (ii) *when* the Linked Data set was generated, and (iii) *how*, e.g., in a single graph or in subgraphs. Besides the aforementioned, this activity is crucial for capturing the origin of the Linked Data, as only at this step that information is known (in combination with the data description and acquisition).

Linked Data publication The published Linked Data is not always identical to the generated one (Figure 5.3, *genRDF Vs. pubRDF*). For instance, it might be the result of merging multiple Linked Data sets which are generated from different data sources at the same or different moments. Moreover, the published Linked Data set might be published in a different way compared to how the Linked Data was generated. For instance, it could be split in different graphs to facilitate its consumption. This might lead to different metadata for the generation and publication activities, and these metadata sets might have different purposes.

For instance, VOID access information metadata is more meaningful and possible to be generated during the RDF data publication, whereas provenance information in respect to the original data can only be defined during the RDF data generation activity. To the contrary, VOID structural or statistical metadata might be generated both during RDF data generation and publication. However, the generated RDF data is not always identical to the one published. If the generated RDF data differs from the one published, then such metadata should be defined for both cases (see Table 5.1).

5.5.2 Metadata Details Levels

There are different detail levels for capturing provenance and metadata information. However, in most cases so far, the provenance and metadata information is delivered on dataset level. This mainly occurs because the metadata information are only defined after the RDF data is generated and/or published. However, different applications and data consumption cases require different levels of provenance and metadata information. Overall, the goal is to achieve the best trade-off between details level and number of additional triples generated for balancing information overhead and acceptable information loss in an automated metadata generation occasion. For instance, considering RDF reification for capturing all provenance and metadata information for each triple, means that metadata referring to the entire RDF dataset is captured repeatedly for each individual triple. To the contrary, considering an implicit graph on dataset level results in information loss in respect to the origin of each triple, if multiple data sources are used to generate the RDF dataset, because it is not explicitly defined where each triple is derived from.

Automating the provenance and metadata information generation, allows exploiting hybridic approaches which can contribute in optimizing the metadata information balance. In this section, we outline the different detail levels for capturing metadata that we identified: *dataset level*, *named graph level*, *partition level*, *triple level* and *term level*. For each level, we describe what type of metadata is captured and we discuss the advantages and disadvantages when used in combination with different representation approaches.

Dataset Level *Dataset level* provenance and metadata provide high-level information for the complete Linked Data set. This level of detail is meaningful for metadata that refer to the whole dataset, i.e., a `void:Dataset` and are the same for each triple. Thus, among the alternative representation approaches, considering an *explicit* or *implicit graph* for the Linked Data set to represent provenance and metadata annotations is sufficient on *dataset level* and it requires the least number of additional triples. The alternative approaches, in principle, assign same metadata information to each triple, causing unnecessary overhead due to replication.

Provenance on *dataset level* is sufficient if all triples are derived from the same original data source and are generated at the same time, as a result of a single activity. The same occurs if the original source is sufficient to assess the Linked Data set trustworthiness. On the contrary, if being aware of the exact data source is required, for instance to align the semantically annotated representation with the original data values, more detailed provenance information is desired, because capturing it on high level is not as complete and accurate to accomplish the desired task.

Named Graph Level A Linked Data set might consist of one or more named graphs. Named graph based subsets of a Linked Data set provide conceptual partitions semanticfully distinguished in graphs. *Named graph level* provenance and metadata refer to all Linked Data annotations which are related to a certain named graph and contain information for each one of the named graphs. Each named graph is a `void:Dataset` and consists a subset of the whole Linked Data set.

In the case of *named graphs*, it is not possible to represent metadata and provenance using explicit graphs, because the RDF statements are already *quads* and the named graph has different semantics than providing metadata information. As in the case of *dataset level*, *implicit graphs* for each named graph and for the complete dataset generate the minimum number of additional RDF triples. Moreover, the *named graph level* metadata information are sufficient if all triples of a certain named graph are derived from the same data source. Otherwise, there is information loss which can be addressed at a narrower detail level.

Partitioned Dataset Metadata Level A dataset might be partitioned based on different aspects. The most frequent partitions are related to (i) the underlying data source or the triple's (ii) subject, (iii) predicate, or (iv) object. Besides the aforementioned partitions, any other custom partition can be equally considered. A *source-based* partitioned RDF dataset is an RDF dataset whose subsets are formed with respect to their derivation source. To be more precise, all RDF terms and triples are derived from the same original data source. *Source-based* partitioned RDF datasets derived from a single data source are not considered because they coincide with the actual RDF dataset. A *subject-based* partitioned RDF dataset is the part of an RDF graph whose triples share the same subject. Consequently, *subject-level* metadata provides information for all triples which share the same subject. It similarly applies to the case of *predicate-based* or *object-based* partitions.

Partitioned datasets might be treated in the same way as named graphs, but it is also possible to use *explicit graphs* to define the subsets metadata. An *implicit graph* for each subset of the RDF dataset which resembles a partition achieves generating the minimum number of additional triples for the metadata information. In the particular

case of *predicate-based* partition, representing the provenance and metadata information using *singleton properties* would cause generating almost the same number of additional triples as in case of defining an *explicit* or *implicit graph* per partition.

Triple Level If metadata is captured on *triple level*, it becomes possible to keep track of the data source each triple was derived from. However, that causes the generation of RDF annotations for metadata whose number of triples is larger than the actual dataset. In the simplest case, the number of additional triples for the metadata information depends on the number of data sources. The more data sources, the more metadata information to be defined. *Triple level* metadata become meaningful also in the case of big data or streamed data where the time one triple was generated might significantly differ compared to the rest triples of the RDF dataset.

In case of *triple level* metadata, singleton properties become meaningful when statements about all RDF triples sharing the same property share the same metadata information. For instance, if all RDF triples whose RDF terms are associated using a certain predicate, share the same metadata, for instance they are all derived from the same data source.

RDF Term Level Even RDF terms that are part of a certain RDF triple can derive from different data sources. For instance, an RDF term is generated considering some data value derived from a source A. This RDF term might constitute the subject of an RDF triple whose object though is an RDF term derived from a source B. In this case, even more detailed metadata information is required to keep track of provenance information. Among the alternative approaches for representing metadata, the RDF reification becomes meaningful at this level of detail. To be more precise, it is meaningful in cases that the RDF terms that form an RDF triple derive from different data sources and/or are generated at a different time.

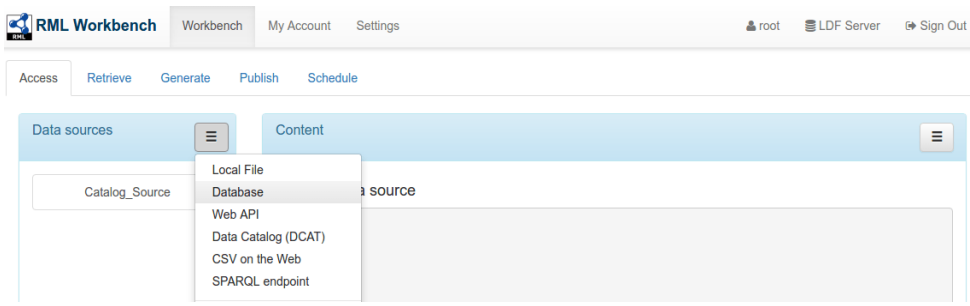
5.6 RML Workbench

To minimize the effort and knowledge that agents need to administrate the overall Linked Data generation workflow, a user-friendly application is required. To this end, we developed the multi-user browser RMLWorkbench. A screencast of the RMLWorkbench is available at `https://youtu.be/8UkI01nQNxc`.

Depending on their assigned roles, data owners can view and manage different sources for retrieving raw data and the corresponding mappings to generate Linked Data, in contrast to the RMLEditor [17] which only focuses on editing mapping rules.

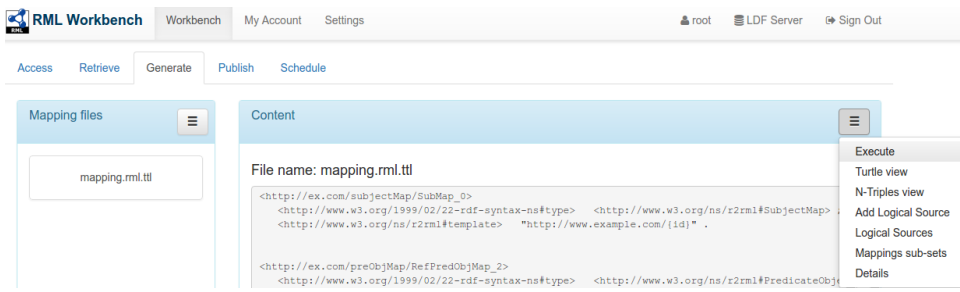
The RMLWorkbench design principles follow the classical multitier client-server architecture. All mapping rules, including the aligned data sources description, may be described using the RMLWorkbench, and exported and re-used by other tools to replicate certain Linked Data generation. The RMLWorkbench consists of five panels: *Access*, *Retrieve*, *Generate*, *Publish*, and *Schedule*:

Access panel. Agents can manage their own sources, which can be accessed through interfaces for local files, databases, or Web sources. The descriptions are annotated using different vocabularies, e.g., DCAT, CSVW, or Hydra. For instance, a user specifies a database accessed via a certain JDBC, and labels it “DB Source” and a dataset published on a DCAT catalog, which he labels “Catalog Source”.



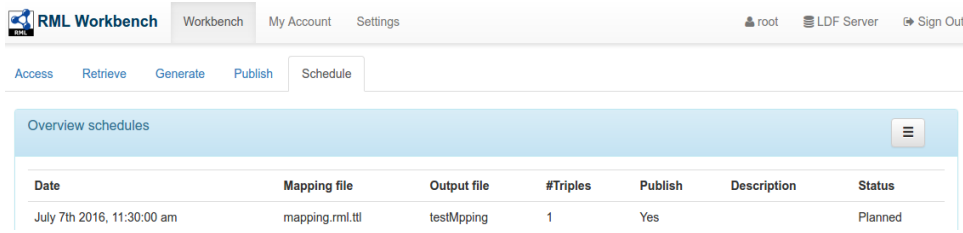
Retrieve panel Not all data which appears in a data source is required to generate the desired Linked Data. Distinct subsets may be considered separately for generating different Linked Data sets. The *Retrieve* panel allows agents to specify which exact data is retrieved for each selection. For instance, via the *Retrieve panel*, an agent specifies the exact tables, which are eventually considered to generate certain Linked Data. The agent specifies the “Singers” and “Albums” tables of the “DB source” and labels them as “Singer data” and “Album data” respectively. The same agent may specify the label “Performance data”, for the dataset about performances derived from the “Catalog source” and precisely its XML distribution.

Generate panel To generate the desired Linked Data, the agents need to specify sets of mapping rules. The RMLWorkbench allows agents to (i) upload a mapping document, (ii) specify a Web source with mapping rules, or (iii) directly edit them via its interface. Different sets of mapping rules may be associated with the same data, generating thus different Linked Data *views*. Once the set of mapping rules is associated with some raw data, the agents can execute the mapping and generate the desired Linked Data („Execute” button, as shown in the following figure). The dataset is then listed among the datasets available for publishing, or the agents are notified if the generation was not successful. The agents can specify mapping rules, for instance the sets of “Singer mappings” and “Performance mappings”. Once the mapping rules are listed among the available sets, the agents can associate them with the corresponding data (“Add Logical Source” button), in our example the “Singer data” and “Performance data”. Furthermore, the agents may desire to generate another Linked Data set with the same data. In that case, another set of mapping rules is added, for instance the “Person mappings”, and the user associates it with the “Singer data” as well.



Publish panel A frequent activity after generating Linked Data is its publication. The RMLWorkbench supports users to easily accomplish this activity. In our example, the Linked Data is published via an LDF server[19]. Nevertheless, the administrator can easily configure other interfaces, for instance SPARQL endpoints. The users can then choose one or more of them to publish their Linked Data.

Schedule panel. In most cases, the Linked Data generation and publication is a recurring activity. Agents periodically regenerate their Linked Data set to keep it up-to-date with the original data. The RMLWorkbench allows to schedule the Linked Data generation and publication activities.



Conclusions

In this chapter, we introduce an approach that exploits ontologies and vocabularies originally used to advertise services or datasets, to define how to access data whose semantically enhanced representation is desired. While, machine-interpretable descriptions of data access remain independent, their alignment with uniform machine-interpretable mapping rules leads to a granular but robust solution which further automates and facilitates the generation of semantically enhanced representations. Moreover, a uniform, iterative, incremental assessment and refinement workflow was presented that enables refining the mapping rules and thus produces high quality Linked Data in the end.

We also show how metadata of the fundamental activities for the Linked Data generation and publication can be automatically generated. Based on the provided metadata infor-

[19]Linked Data Fragments server, `https://github.com/LinkedDataFragments/Server.js`

mation, it is expected that Linked Data publication infrastructures will re-determine certain properties regarding the metadata, if the Linked Data set is reformed before it gets published. Moreover, the metadata can be enriched with additional information derived from other activities involved in the Linked Data publication workflow. Provenance and metadata can be multidimensional and its consumption diverges across different systems. Different applications require different levels of metadata to fulfill their tasks, whereas diverse metadata might be desired. The presented workflow was focused on the essential parts of the Linked Data publication workflow. However, any metadata information might be considered as they accrue from other activities involved in the Linked Data publishing workflow.

References

[1] **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. URL http://ceur-ws.org/Vol-1184/ldow2014_paper_01.pdf.

[2] **Anastasia Dimou**, Dimitris Kontokostas, Markus Freudenberg, Ruben Verborgh, Jens Lehmann, Erik Mannens, Sebastian Hellmann, and Rik Van de Walle. Assessing and Refining Mappings to RDF to Improve Dataset Quality. In Marcelo Arenas, Oscar Corcho, Elena Simperl, Markus Strohmaier, Mathieu d'Aquin, Kavitha Srinivas, Paul Groth, Michel Dumontier, Jeff Heflin, Krishnaprasad Thirunarayan, and Steffen Staab, editors, *The Semantic Web – ISWC 2015*, volume 9367 of *Lecture Notes in Computer Science*, pages 133–149. Springer International Publishing, Oct 2015. doi: 10.1007/978-3-319-25010-6_8. URL http://link.springer.com/chapter/10.1007/978-3-319-25010-6_8.

[3] **Anastasia Dimou**, Ruben Verborgh, Miel Vander Sande, Erik Mannens, and Rik Van de Walle. Machine-interpretable Dataset and Service Descriptions for Heterogeneous Data Access and Retrieval. In *Proceedings of the 11th International Conference on Semantic Systems*, pages 145–152, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3462-4. doi: 10.1145/2814864.2814873. URL http://doi.acm.org/10.1145/2814864.2814873.

[4] **Anastasia Dimou**, Tom De Nies, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. Automated Metadata Generation for Linked Data Generation and Publishing Workflows. In Sören Auer, Tim Berners-Lee, Christian Bizer, and Tom Heath, editors, *Proceedings of the 9th Workshop on Linked Data on the Web*, volume 1593 of *CEUR Workshop Proceedings*, April 2016.

[5] **Anastasia Dimou**, Dimitris Kontokostas, Markus Freudenberg, Ruben Verborgh, Jens Lehmann, Erik Mannens, and Sebastian Hellmann. DBpedia Mappings Quality Assessment. In Takahiro Kawamura and Heiko Paulheim, editors, *Proceedings of the 15th International Semantic Web Conference: Posters and Demos*, volume 1690 of *CEUR Workshop Proceedings*, October 2016. URL http://ceur-ws.org/Vol-1690/paper97.pdf.

[6] Keith Alexander, Richard Cyganiak, Michael Hausenblas, and Jun Zhao. Describing Linked Datasets with the VoID Vocabulary. Interest Group Note, W3C, March 2011. http://www.w3.org/TR/void/.

[7] Gavin Carothers and Andy Seaborne. RDF 1.1 TriG. W3C Recommendation, W3C, February 2014. URL http://www.w3.org/TR/trig/.

[8] Gavy Carothers. RDF 1.1 N-Quads. W3C Recommendation, W3C, February 2014. http://www.w3.org/TR/n-quads/.

[9] Erik Christensen, Francisco Curbera, Greg Meredith, and Sanjiva Weerawarana. Web Services Description Language (WSDL) 1.1. W3C Note, March 2001. http://www.w3.org/TR/wsdl.

[10] Dan Connolly. JSON-LD 1.0. W3C Recommendation, W3C, January 2014. http://www.w3.org/TR/json-ld/.

[11] Richard Cyganiak, Chris Bizer, Jörg Garbers, Oliver Maresch, and Christian Becker. The D2RQ Mapping Language. Technical report, W3C, March 2012. http://d2rq.org/d2rq-language.

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

[13] Jos de Bruijn, Christoph Bussler, John Domingue, Dieter Fensel, Martin Hepp, Uwe Keller, Michael Kifer, Birgitta König-Ries, Jacek Kopecky, Rubén Lara, Holger Lausen, Eyal Oren, Axel Polleres, Dumitru Roman, James Scicluna, and Michael Stollberg. Web Service Modeling Ontology (WSMO). W3C Member Submission, June 2005. http://www.w3.org/Submission/WSMO/.

[14] Jos de Bruijn, Dieter Fensel, Uwe Keller, Michael Kifer, Holger Lausen, Reto Krummenacher, Axel Polleres, and Livia Predoiu. Web Service Modeling Language (WSML). W3C Member Submission, June 2005. http://www.w3.org/Submission/WSML/.

[15] LucianoFrontino de Medeiros, Freddy Priyatna, and Oscar Corcho. MIRROR: Automatic R2RML Mapping Generation from Relational Databases. In *Engineering the Web in the Big Data Era*. 2015.

[16] Olaf Hartig and Jun Zhao. *Provenance and Annotation of Data and Processes: Third International Provenance and Annotation Workshop, IPAW 2010*, chapter Publishing and Consuming Provenance Metadata on the Web of Linked Data. 2010.

[17] 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 Harald Sack, Eva Blomqvist, Mathieu d'Aquin, Chiara Ghidini, Paolo Simone Ponzetto, and Christoph Lange, editors, *The Semantic Web – Latest Advances and New Domains (ESWC 2016)*, volume 9678 of *Lecture Notes in Computer Science*, pages 709–723. Springer, 2016. ISBN 978-3-319-34129-3. doi: 10.1007/978-3-319-34129-3_43. URL http://dx.doi.org/10.1007/978-3-319-34129-3_43.

[18] Pascal Hitzler, Markus Krötzsch, Bijan Parsia, Peter Patel-Schneider, and Sebastian Rudolph. OWL 2 Web Ontology Language. W3C Recommendation, W3C, December 2012. http://www.w3.org/TR/owl-primer.

[19] Ernesto Jiménez-Ruiz, Evgeny Kharlamov, Dmitriy Zheleznyakov, Ian Horrocks, Christoph Pinkel, MartinG. Skjæveland, Evgenij Thorstensen, and Jose Mora. BootOX: Practical Mapping of RDBs to OWL 2. In *The Semantic Web - ISWC 2015*. 2015.

[20] Jacek Kopecký. Web Services Description Language (WSDL) Version 2.0: RDF Mapping. W3C Working Group Note, June 2007. http://www.w3.org/TR/wsdl20-rdf/.

[21] Jacek Kopecký, Tomas Vitvar, Carine Bournez, and Joel Farrell. SAWSDL: Semantic Annotations for WSDL and XML Schema. *IEEE Internet Computing*, 11, 2007.

[22] Jacek Kopecký, Karthik Gomadam, and Tomas Vitvar. hrests: An HTML Microformat for Describing RESTful Web Services. In *Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 01*. IEEE Computer Society, 2008.

[23] Markus Lanthaler. Hydra Core Vocabulary. Unofficial Draft, W3C, June 2014. http://www.hydra-cg.com/spec/latest/core/.

[24] Rubén Lara, Dumitru Roman, Axel Polleres, and Dieter Fensel. A Conceptual Comparison of WSMO and OWL-S. In *Web Services*, volume 3250 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2004.

[25] Timothy Lebo, Satya Sahoo, and Deborah McGuinness. PROV-O: The PROV Ontology. W3C Recommendation, W3C, April 2013. https://www.w3.org/TR/prov-o/.

[26] Fadi Maali and John Erickson. Data Catalog Vocabulary (DCAT). W3C Recommendation, W3C, January 2014. http://www.w3.org/TR/vocab-dcat/.

[27] Maria Maleshkova, Jacek Kopecký, and Carlos Pedrinaci. Adapting SAWSDL for Semantic Annotations of RESTful Services. In *On the Move to Meaningful Internet Systems: OTM 2009 Workshops*, volume 5872 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2009.

[28] David Martin, Mark Burstein, Jerry Hobbs, Ora Lassila, Drew McDermott, Sheila McIlraith, Srini Narayanan, Massimo Paolucci, Bijan Parsia, Terry Payne, Evren Sirin, Naveen Srinivasan, and Katia Sycara. OWL-S: Semantic Markup for Web Services. W3C Member Submission, November 2004. http://www.w3.org/Submission/OWL-S/.

[29] Luc Moreau and Paolo Missier. PROV-DM: The PROV Data Model. Working Group Recommendation, W3C, April 2013. http://www.w3.org/TR/prov-dm/ .

[30] Vinh Nguyen, Olivier Bodenreider, and Amit Sheth. Don't Like RDF Reification?: Making Statements About Statements Using Singleton Property. In *Proceedings of the 23rd International Conference on World Wide Web*, 2014.

[31] 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, 2013.

[32] Max Schmachtenberg, Christian Bizer, and Heiko Paulheim. Adoption of the Linked Data Best Practices in Different Topical Domains. In Peter Mika, Tania Tudorache, Abraham Bernstein, Chris Welty, Craig Knoblock, Denny Vrandečić, Paul Groth, Natasha Noy, Krzysztof Janowicz, and Carole Goble, editors, *The Semantic Web – ISWC 2014: 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I*, pages 245–260, Cham, 2014. Springer International Publishing. ISBN 978-3-319-11964-9. doi: 10.1007/978-3-319-11964-9_16. URL http://dx.doi.org/10.1007/978-3-319-11964-9_16.

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

[34] Jeni Tennison, Gregg Kellogg, and Ivan Herman. Model for Tabular Data and Metadata on the Web. W3C Working Draft, W3C, April 2015. http://www.w3.org/TR/2015/WD-tabular-data-model-20150416/.

[35] Gregory Todd Williams. SPARQL 1.1 Service Description. W3C Recommendation, W3C, March 2013. http://www.w3.org/TR/sparql11-service-description/.

[36] Ruben Verborgh, Miel Vander Sande, Olaf Hartig, Joachim Van Herwegen, Laurens De Vocht, Ben De Meester, Gerald Haesendonck, and Pieter Colpaert. Triple Pattern Fragments: a Low-cost Knowledge Graph Interface for the Web. *Journal of Web Semantics*, 37–38:184–206, March 2016. ISSN 1570-8268. doi: doi:10.1016/j.websem.2016.03.003. URL http://linkeddatafragments.org/publications/jws2016.pdf.

[37] Tomas Vitvar, Jacek Kopecký, Jana Viskova, and Dieter Fensel. WSMO-Lite Annotations for Web Services. In *The Semantic Web: Research and Applications*, volume 5021 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2008.

Πάντ' εστίν εξευρείν, αν μη τον πόνον φεύγη τις.
όλα γίνονται, αν δεν αποφεύγει κανείς να κοπιάσει.

Δημοσθένης

Everything can be done, if you do not avoid labor.

Demosthenes

6

Use Cases

We describe five use cases where our mapping language, RML (Chapter 2), the corresponding implementations, namely the RMLEditor (Section 2.6), RMLMapper (Section 3.4), RMLValidator (Section 4.5) and RMLWorkbench (Section 5.6), and our workflows for Linked Data generation are progressively introduced and used.

chapter's outline We describe five use cases which are also summarized in Table 6.1.

The first use case is related to *open data* (Section 6.1). An RML-based solution was used to specify the mapping rules to semantically annotate data derived from governmental institutions in the frame of the EWI Open Data project[1], and lasted from 2013 to 2015.

The second use case is related to *semantic publishing* (Section 6.2). An RML-based solution was used to specify the mapping rules to semantically annotate data residing at iMinds data warehouse and was aligned with data derived from universities repositories. This use case was accomplished in the frame of the iLastic project[2], and lasted from 2015 to 2016.

The third use case is related to *business-to-business data transactions*, based on the Data-as-a-Service (DaaS) principle, (Section 6.3). An RML-based solution was used to semantically annotate data residing in different partners' data repositories. This use case was accomplished in the frame of the COMBUST project[3], and lasted from 2014 to 2017.

The fourth use case is related to DBpedia (Section 6.4). The rules for generating the DBpedia Linked Data set were translated into RML and the RMLProcessor was integrated in the DBpedia EF. This use case was accomplished in the frame of our contribution to the broader DBpedia community and its corresponding association, as well as the Google Summer of Code program (GSoC), and lasted from 2015 to 2017.

The last use case is related to *semantic publishing* too (Section 6.5). RML was used to specify rules to semantically annotate data derived from the CEUR-WS.org workshops proceedings volumes, which are published as Web pages. The RML-based solution was compared to other solutions fulfilling the same task in the frame of the Semantic Publishing Challenge. This use case was accomplished by participating and organizing the Semantic Publishing Challenge, and lasted from 2014 to 2016.

Each use case is described in detail below. The Linked Data generation workflow is presented for all use cases, as well as particularities related to each use case specifically.

chapter's sources This chapter is based on the following papers:

1. **Anastasia Dimou**, Sahar Vahdati, Angelo Di Iorio, Christoph Lange, Ruben Verborgh, and Erik Mannens. Challenges as Enablers for High Quality Linked Data: Insights from the Semantic Publishing Challenge. *PeerJ Computer Science*, 3: e105, January 2017. ISSN 2376-5992. doi: 10.7717/peerj-cs.105. URL `https://peerj.com/articles/cs-105/`

2. **Anastasia Dimou**, Gerald Haesendonck, Martin Vanbrabant, Laurens De Vocht, Ruben Verborgh, Steven Latré, and Erik Mannens. iLastic: Linked Data Generation Workflow and User Interface for iMinds Scholarly Data. In *Proceedings of the Workshop on Semantics, Analytics, Visualisation: Enhancing Scholarly Data*, April 2017. (to appear)

[1] EWI Open Data, `http://ewi.mmlab.be/`
[2] iLastic, `http://explore.ilastic.be/`
[3] COMBUST, `https://www.iminds.be/en/projects/combust`

3. **Anastasia Dimou**, Dimitris Kontokostas, Markus Freudenberg, Ruben Verborgh, Jens Lehmann, Erik Mannens, and Sebastian Hellmann. DBpedia Mappings Quality Assessment. In Takahiro Kawamura and Heiko Paulheim, editors, *Proceedings of the 15th International Semantic Web Conference: Posters and Demos*, volume 1690 of *CEUR Workshop Proceedings*, October 2016. URL http://ceur-ws.org/Vol-1690/paper97.pdf

4. **Anastasia Dimou**, Angelo Di Iorio, Christoph Lange, and Sahar Vahdati. Semantic Publishing Challenge – Assessing the Quality of Scientific Output in Its Ecosystem". In Harald Sack, Stefan Dietze, Anna Tordai, and Christoph Lange, editors, *Semantic Web Challenges: Third SemWebEval Challenge at ESWC 2016, Heraklion, Crete, Greece, May 29 - June 2, 2016, Revised Selected Papers*, pages 243–254, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46565-4. doi: 10.1007/978-3-319-46565-4_19. URL http://dx.doi.org/10.1007/978-3-319-46565-4_19

5. **Anastasia Dimou**, Miel Vander Sande, Pieter Colpaert, Laurens De Vocht, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. Extraction and Semantic Annotation of Workshop Proceedings in HTML using RML. In Valentina Presutti, Milan Stankovic, Erik Cambria, Iván Cantador, Angelo Di Iorio, Tommaso Di Noia, Christoph Lange, Diego Reforgiato Recupero, and Anna Tordai, editors, *Semantic Web Evaluation Challenge: SemWebEval 2014 at ESWC 2014, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers*, pages 114–119, Cham, 2014. Springer International Publishing. URL http://dx.doi.org/10.1007/978-3-319-12024-9_15

6. **Anastasia Dimou**, Laurens De Vocht, Geert Van Grootel, Leen Van Campe, Jeroen Latour, Erik Mannens, and Rik Van de Walle. Visualizing the Information of a Linked Open Data Enabled Research Information System. *Procedia Computer Science*, 33:245 – 252, 2014. ISSN 1877-0509. doi: http://dx.doi.org/10.1016/j.procs.2014.06.039. URL http://www.sciencedirect.com/science/article/pii/S1877050914008291

7. Wouter Maroy, **Anastasia Dimou**, Dimitris Kontokostas, Ben De Meester, Ruben Verborgh, Jens Lehmann, Erik Mannens, and Sebastian Hellmann. Sustainable Linked Data Generation: The Case of DBpedia. In Claudia d'Amato, Miriam Fernandez, Valentina Tamma, Freddy Lecue, Philippe Cudré-Mauroux, Juan Sequeda, Christoph Lange, and Jeff Heflin, editors, *The Semantic Web – ISWC 2017: 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part II*, pages 297–313, Cham, 2017. Springer International Publishing. ISBN 978-3-319-68204-4. doi: 10.1007/978-3-319-68204-4_28. URL https://doi.org/10.1007/978-3-319-68204-4_28

6.1 Open Data

The open data movement is widely spreading, while its governmental data perspective is driven by the fundamental idea that governments have the right and duty to publish public data. Linked Data may act as an enabler for such open data ecosystems. In this context, this use case investigates Linked Data potential as a promising technology that enables the Open Data Ecosystem's requirements and become the basis to validate the Flemish Open Data Portal across various sectors and in different contexts.

This use case serves as a guide for civil servants and other data publishers, who are interested in publishing contact information about services and persons within Flemish local governments and central administrations. It enables data consumers (civil servants or beyond) to consult personal contact details of civil servants with the same function, job activity, or policy field. Going beyond administrative borders and cross-organizational interoperability issues, Linked Data enables integrating and publishing information, such as generic and personal phone numbers or e-mail addresses.

In the rest of this section, the data model (Section 6.1.1), vocabularies (Section 6.1.2), and Linked Data generation and publishing workflow (Section 6.1.3) are described in detail.

6.1.1 EWI Open Data Data Model

The input data for this use case is derived from different municipalities of Flanders: Beveren[4], Destelbergen[5], Ingelmunster[6], Knokke-Heist[7], Kortrijk[8], and Roeselare[9]. The input data is in different structures –tabular, hierarchical and loose-coupled– and formats –CSV, XML, and HTML– as shown in Table 6.2, and describe information about each municipality's governmental organizations. An interesting remark is that the municipalities whose data is originally in CSV format, provided more complete input data with respect to the data model compared to the municipalities whose data was in other formats.

[4] Beveren, `http://www.beveren.be/`
[5] Destelbergen, `http://www.destelbergen.be/`
[6] Ingelmunster, `http://www.ingelmunster.be/`
[7] Knokke-Heist, `http://www.knokke-heist.be/`
[8] Kortrijk, `https://www.kortrijk.be/`
[9] Roeselare, `https://www.roeselare.be/`

6.1.2 EWI Open Data Vocabularies

The following vocabularies were used to semantically annotate the data: ADMS[10], DCMI[11], LOCN[12], ORG[13], OSLO[14], person[15], schema[16], SKOS[17], VCARD[18], and ROV[19].

The Asset Description Metadata Schema (ADMS, with `adms` as its prefix), is a profile of DCAT [17] (used to describe semantic assets) that is highly reusable metadata, and reference data that are used for eGovernment system development. The DCMI Metadata Terms (DCMI with `dcterms` as its prefix) is a general purpose, high level vocabulary which includes metadata terms maintained by the Dublin Core Metadata Initiative (DCMI) [20]. Location core vocabulary (LOCN, with `locn` as its prefix), is a vocabulary for describing any place in terms of its name, address, or geometry. The Organization vocabulary (ORG, with `org` as its prefix) describes information on organizations and organizational structures, including governmental organizations. The Open Standards for LOcal administration vocabulary (OSLO, with `oslo` as its prefix), is used to describe information for local administrations in Flanders which is related to contact information, localisation, and public services and is exchanged by public administration. The Person Core Vocabulary (person, with `person` as its prefix) is used to describe a natural person. The schema vocabulary (schema, with `schema` as its prefix) is launched by big search engines operators to have a common set of annotations for structured data markup on Web pages. The Simple Knowledge Organization System (SKOS, with `skos` as its prefix) is a common data model for knowledge organization systems, such as thesauri, classification schemes, subject heading systems, and taxonomies. The VCARD ontology is used to describe people and organizations. Last, the Registered Organization Vocabulary (ROV, with `rov` as its prefix) is a profile of the Organization Ontology[13] for describing organizations that gained legal entity status via a formal registration process, typically in a national or regional register.

6.1.3 EWI Open Data Workflow

The EWI Open Data use case was the first use case which was explored, so the generation workflow consisted of only three basic steps:

Mapping rules definition The mapping rules statements were written by hand. The Semantic Web experts consulted the data owners and defined the data model accordingly. The Semantic Web experts were also responsible for specifying the RML statements which indicate how the input data should be semantically annotated with

[10] ADMS, `http://www.w3.org/ns/adms#`
[11] DCMI, `http://purl.org/dc/terms/`
[12] LOCN, `http://www.w3.org/ns/locn#`
[13] ORG, `http://www.w3.org/ns/org#`
[14] OSLO, `http://purl.org/oslo/ns/localgov#`
[15] person, `http://www.w3.org/ns/person#`
[16] schema, `http://schema.org/`
[17] skos, `http://www.w3.org/2004/02/skos/core#`
[18] VCARD, `http://www.w3.org/2006/vcard/ns#`
[19] ROV, `http://www.w3.org/ns/regorg#`
[20] DCMI, `http://dublincore.org/`

the desired data model. The whole procedure required a few iterations where the Semantic Web experts defined and executed the mapping rules to generate Linked Data and the data owners provided feedback. As the RMLValidator or any other corresponding tool, was not available yet, the mapping rules validation against the semantic schema was manually accomplished by the Semantic Web experts.

Linked Data generation The input data was either provided in files in the first place or they were exported in files, as the first version of the RMLProcessor supported only data residing in files. Data transformations were performed in advanced (*pre-processing*), as it is recommended by R2RML. Once the mapping rules were defined, the EWI Open Data Linked Data sets were generated.

Linked Data publication Each data owner published its own Linked Data set. To achieve that a Virtuoso SPARQL endpoint[21] is used by each data owner. The Virtuoso SPARQL query service implements the SPARQL protocol for RDF and enables access via querying to the Linked Data sets which are published through its access interface.

6.2 iLastic

The iLastic project was launched in 2015 by the iMinds research institute[22] –nowadays imec[23]– to publish scholar data associated with any of the iMinds labs. The iMinds labs are spread across Flanders universities, thus researchers affiliated with iMinds are also affiliated with a university and their publications are archived by both iMinds and the corresponding university. This section's content is based on **Anastasia Dimou** et al. [6].

The iLastic use case aims to align Linked Data which is generated from (semi-)structured data with Linked Data generated from plain text enrichment. The (semi-)structured data is in hierarchical structure, JSON format and derived from a Web API, while some more data is derived from a database directly. The plain text is extracted from PDF files and enriched by applying Named Entity Recognition (NER). The use case involves generating Linked Data derived from data in the iMinds data warehouse which gets enriched with knowledge extracted from the publications content. To achieve that, Flemish universities' digital repositories were considered, as they provide full content of open accessed publications.

The use case relies on (i) the RML-*based workflow*, which was used for semantically annotating data derived from the iMinds data warehouse, aligned with (ii) the *iLastic enricher*, an in-house mechanism for publications' retrieval and enrichment; and (iii) an extensible and adjustable *user interface* that aims to facilitate non-Semantic Web experts to search and explore the semantically enriched data. This use case was conducted in two phases:

First Phase A *proof-of-concept* aiming to assess the solution's feasibility and potential was carried out during the first phase. We relied on selected data retrieved from the iMinds data warehouse to form the first version of the iLastic Linked Data set.

[21] Virtuoso open-source edition, `https://virtuoso.openlinksw.com/dataspace/doc/dav/wiki/Main/`

[22] iMinds, `https://www.iminds.be/`

[23] imec, `http://imec.be/`

Second Phase In the second phase, two goals were set: (i) enrich the first version of the iLastic Linked Data with Linked Data generated from the publications' content; and (ii) automate the Linked Data generation workflow to systematically generate Linked Data from the iMinds data warehouse, enrich them and publish them altogether. In this phase, the solution was packaged, so other agents only need to configure their own rules for their data and repositories to generate their own Linked Data.

6.2.1 iLastic Model

The iLastic dataset consists of data which describe (i) people, (ii) publications, (iii) projects, and (iv) organizations derived from the iMinds data warehouse. More details about each type of entity follows:

People The iLastic dataset includes data regarding people who work for iMinds, but not exclusively. They might be researchers, who belong to one of the iMinds labs and were authors of publications. Besides researchers affiliated with iMinds, many more people might appear in the iMinds data warehouse, even though they do not belong in any of the iMinds labs, thus they are not iMinds personnel, but they co-authored one or more papers with one or more iMinds researchers. People are associated with their publications, organizations, and, rarely, projects they are involved in, if their role is known, for instance if they are project or research leads or contact persons.

Publications The iLastic dataset includes information regarding publications where at least one of the co-authors is an iMinds researcher. Each publication that is registered in the iMinds data warehouse is assigned a unique identifier. Nevertheless, even though the iMinds data warehouse includes information regarding publications, it refers mainly to metadata, such as the title, authors publication date or category. There is no information regarding the actual content of publications. To enrich the information regarding publications, we integrated data from complementary repositories, namely universities' repositories, such as Ghent University Academic Bibliography digital repository[24] or the digital repository for KU Leuven Association research[25]. These repositories also provide the PDF file of open accessed publication which can be parsed and analyzed to derive more information.

Organizations The iMinds research institute is a multi-part organization consisting of several labs associated with different universities in Flanders. Information about each of the labs was required to be semantically annotated. Persons, publications, and projects are linked to the different iMinds labs.

Projects Information related to projects the different iMinds labs are involved in is also included. The projects are associated with people who work on them, but only the information regarding the projects' research and management leads, and contact person was considered.

[24] UGent biblio, `https://biblio.ugent.be/`
[25] KU Leuven lirias, `https://lirias.kuleuven.be/`

Modeling Challenges iMinds personnel and publications are uniquely identified with identifiers which the CRM system assigns. However, researchers who are co-authors in publications and do not belong to any of the iMinds labs, thus are not iMinds personnel, are not assigned such a unique identifier.

On the one hand, there were three major challenges to be addressed with respect to persons: (i) *distinguish iMinds* and *non-iMinds researchers*; and (ii) among the non-iMinds researchers, *identify the same person* appearing in the dataset multiple times, being only aware of the researchers name (and on certain occasions their affiliation). Besides data from the iMinds data warehouse, integrating information extracted from the papers' content required dealing with one more challenge: (iii) *associate authors extracted from the publications' content* with the people which appear in the iMinds data warehouse.

On the other hand, there were two challenges encountered with respect to publications' semantic annotation: (i) *aligning* publications as they appear in iMinds data warehouse with corresponding publications in universities repositories; and (ii) *enriching* the structured data annotation derived from the iMinds data warehouse with plain text enrichment derived from the publications' actual content. In the former case, proper algorithms and heuristics were required to identify the publications' content by comparing the publications' titles, as they appear in the iMinds data warehouse, with the titles as extracted from the publications' PDF. In the latter case, once the PDF of a certain publication was identified, the extraction of meaningful keywords, the recognition of well-known entities among those keywords, and the enrichment of the publications with this additional knowledge was required.

6.2.2 iLastic Vocabulary

The following vocabularies are used to semantically annotate iMinds scholarly data: BIBO[26], bibTex[27], CERIF[28], DCMI[29], and FOAF[30] (a high level list of classes and properties is available at Table 6.6). The Bibliographic Ontology (BIBO) provides basic concepts and properties to describe citations and bibliographic references. The bibTeX is used to describe bibTeX entries. The Common European Research Information Format (CERIF) ontology provides basic concepts and properties for semantically annotating research information. The DCMI Metadata Terms (DCMI) includes metadata terms maintained by the Dublin Core Metadata Initiative, and the Friend Of A Friend (FOAF) ontology is used to describe people.

CERIF, DCMI and FOAF vocabularies were used to annotate data regarding people. The more generic DCMI and FOAF vocabularies were used to annotate information regarding, for instance, the name and surname of the author, whereas CERIF was used to define and associate with its organization and publications. BIBO, BibTex, CERIF, and DCMI vocabularies were used to annotate publications, FOAF, CERIF, and DC to annotate organizational units and CERIF to annotate projects. We used custom properties to cover cases where the aforementioned or other vocabularies did not have properties to annotate particular inter-

[26] BIBO, `http://purl.org/ontology/bibo/`
[27] bibTex, `http://purl.org/net/nknouf/ns/bibtex#`
[28] CERIF, `http://spi-fm.uca.es/neologism/cerif#`
[29] DCMI, `http://dublincore.org/documents/dcmi-terms/`
[30] FOAF, `http://xmlns.com/foaf/0.1/`

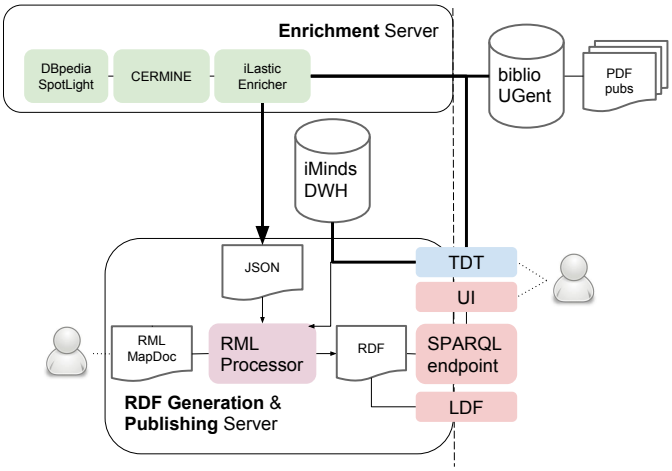


Figure 6.1: *Linked Data set Generation and Publishing Workflow for the iLastic project.*

nal concepts of iMinds data. For instance, iMinds tracks if a certain publication is indexed by Web Of Science[31]. Thus, a custom property (`im:webOfScience`) was introduced to represent this knowledge. Moreover, iMinds classifies publications in different categories. A custom property (`im:publicationCategory`) was introduced for this purpose.

6.2.3 iLastic Linked Data set

The iLastic dataset contains information about 59,462 entities: 12,472 researchers (both people affiliated with iMinds and externals), 22,728 publications, 81 organizational units, and 3,295 projects. It consists of 765,603 triples in total and is available for querying at `http://explore.ilastic.be/sparql`.

6.2.4 iLastic Workflow

The iLastic workflow consists of two pipelines: (i) enriching the research metadata derived from the iMinds data warehouse, and (ii) enriching the publications' content. The two pipelines aim to deal with the peculiarities of the different nature that the original data has, namely the structured data and plain text data, while they merge when the final Linked Data set is generated and published. A uniform interface is built on top of the iLastic dataset offering searching and navigation functions to the users.

iLastic Mapping Rules Definition and Validation A sample of data derived from the iMinds data warehouse was used to define the mapping rules that specify how the iLastic

[31] Web of science, `http://www.webofknowledge.com/`

Linked Data is generated. To facilitate the mapping rules editing, the RMLEditor was incorporated [11] (Section 2.6) to define the RML mapping rules (Section 2.4). The exported mapping document is validated for its consistency using the RMLValidator [3] (Section 4.5) to make sure that the defined mapping rules are consistent and no violations occur because multiple vocabularies are (re)used and combined. Any violations are addressed and the final mapping document is produced. The mapping documents for the iLastic project are available at `http://rml.io/data/iLastic/PubAuthGroup_Mapping.rml.ttl`.

iLastic Input Data Access and Retrieval The iLastic workflow consists of two input pipelines for publishing: (i) structured data derived from the iMinds data warehouse, and (ii) results of the plain text enrichment. The two input pipelines are merged at the time of the Linked Data generation. Each input pipeline requires accessing different parts of the data stored in the iMinds data warehouse (as explained in Section 5.3). To achieve that, the corresponding SQL queries were published on a DataTank[32] instance that acts as the interface for accessing the underlying data for both pipelines. The DataTank offers a generic way to publish data sources and provide an HTTPAPI on top of them. The results of the SQL queries executed against the iMinds data warehouse and of the publications' enrichment are proxied by the DataTank and are returned in (paged) JSON format. The original raw data as retrieved from the iMinds data warehouse and made available as Open Data can be found at `http://explore.ilastic.be/iminds`.

iLastic Linked Data Generation The iLastic workflow consists of two input pipelines. The *structured-data pipeline* semantically annotates data derived from the iMinds data warehouse. This pipeline considers input data as retrieved from the DataTank and semantically annotates them with the aforementioned vocabularies and ontologies. To access the iMinds data warehouse, the RMLProcessor relies on the DataTank API's description which is defined using the Hydra vocabulary[33].

The *plain-text-enrichment pipeline* enriches the publications metadata with information derived from the publications' actual content. Thus, retrieving, extracting and processing each publication's text is required. This occurs in coordination with the university repositories. For each publication, the university affiliated with the authors which is also part of iMinds is considered to retrieve the publication from its repository. The same publications that appear both in the iMinds data warehouse and the university's repository are identified applying fuzzy matching over their title and author(s), if the latter is available. More details about this pipeline is available at [6]. The output is summarized in a JSON file, where all identified terms are summarized. The JSON file is passed to the RMLMapper (Section 3.4) where, together with the rest data retrieved from the iMinds data warehouse and the mapping document defined using the RMLEditor, are turned into triples. This way, the corresponding publications information is enriched with data from its own text. For our exemplary case, the API of Ghent University is considered[34].

[32] The DataTank, `http://thedatatank.com/`

[33] Hydra, `http://www.hydra-cg.com/spec/latest/core/`

[34] UGent biblio API, `https://biblio.ugent.be/doc/api`

iLastic Linked Data Publication Once the iLastic Linked Data set is generated, it is stored and published to a Virtuoso instance[35] which is installed on the same server for this purpose. Virtuoso is a cross-platform server that provides a triplestore and a SPARQL endpoint for querying the underlying Linked Data. This endpoint may be used by clients which desire to access the iLastic dataset, as it is used by the DataTank to provide data to the iLastic user interface –described in the next section (Section 6.2.4).

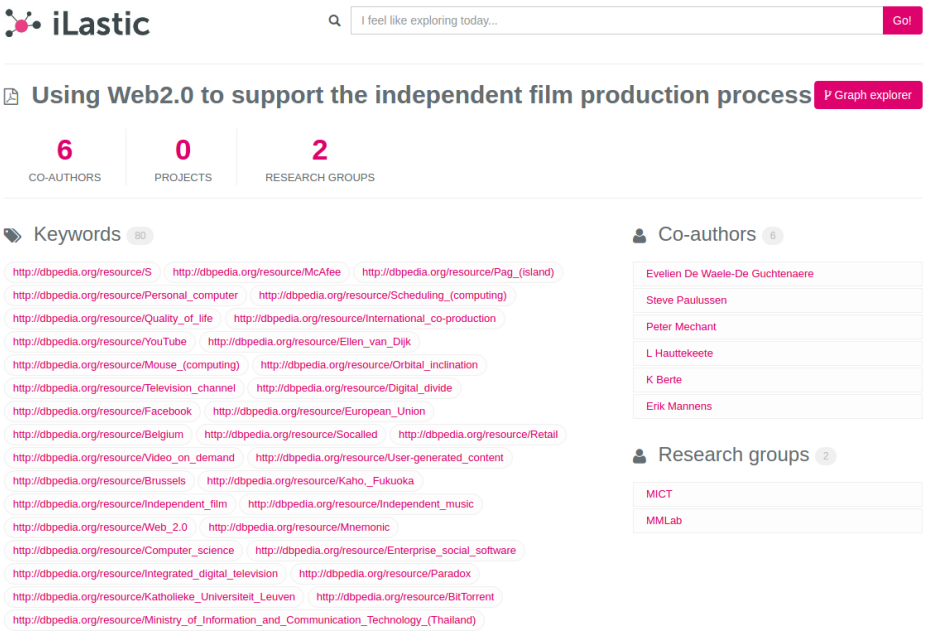


Figure 6.2: *A publication as presented at iLastic user interface with information derived both from iMinds data warehouse and the analyzed and enriched publication's content.*

iLastic User Interface The iLastic user interface was included in the second phase of the project to make the iLastic dataset accessible to non-Semantic Web experts who do not have the knowledge to query it via its endpoint and, hence, showcase its potential. The iLastic user interface relies on LodLive[36] [8], a demonstration of Linked Data standards' use to browse different resources of a dataset. The iLastic user interface can be accessed at `http://explore.iLastic.be` and a sreencast showcasing its functionality is available at `https://youtu.be/ZxGrHnOuSvw`.

The iLastic user interface allows users to explore the underlying dataset either using its *regular interface* or its *graph explorer*, or search within the Linked Data set which is supported by the *iLastic sitemap*. While the users use the *regualar interface*, their requests are translated into SPARQL queries which, on their own turn, are parameterized and published at the DataTank. The *iLastic sitemap* supports searching. It has a tree-like structure

[35] Virtuoso, `https://github.com/openlink/virtuoso-opensource`
[36] LodLive, `http://en.lodlive.it/`

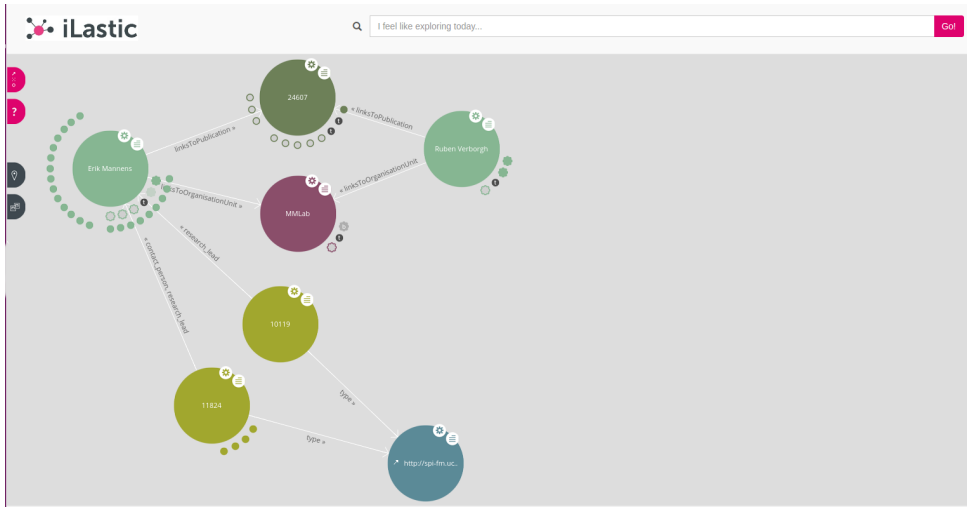


Figure 6.3: *The graph explorer view for Erik Mannens.*

including the different entities of the iLastic project which is indexed and serves as a search API, whose results are then used by the user interface's search application. The iLastic search application builds a front-end around the search API results. Moreover, a user may access the user interface of iLastic to explore the integrated information on publications, as shown in Figure 6.2. Besides the regular user interface, the users may take advantage of the graph explorer. For each one of the iLastic Linked Data set's entities, the user may switch from the regular interface to the graph explorer and vice versa. For instance, the graph explorer for 'Erik Mannens' is shown in Figure 6.3.

6.3 COMBUST

Over the past years, and as the Web evolves in an integrated and interlinked knowledge space, companies are overwhelmed with an increasing amount of data. Agents own data and have different purpose of use, but they are often eager to incorporate complementary data. Nevertheless, to achieve that, it is required to efficiently handle huge volumes of dispersed information which, on its own turn, is hampered by challenges related to data integration, e.g., matches validation or conflicting data aggregation.

The problem of identifying and aligning multiple but possibly (slightly) different representations of the same real-world entities, appearing within multiple distributed heterogeneous sources, aggravates when these sources are of big size, and especially when efficiency and accuracy are very critical. In Semantic Web, those entities are ideally identified by the same IRI (acting as the entity's unique identifier). However, aligning the same entities to automatically integrate these data sources is a challenging and meticulous procedure. Existing technologies would uniquely identify and semantically annotate the different instances separately, generating multiple IRIs for each distinct instance of the entity.

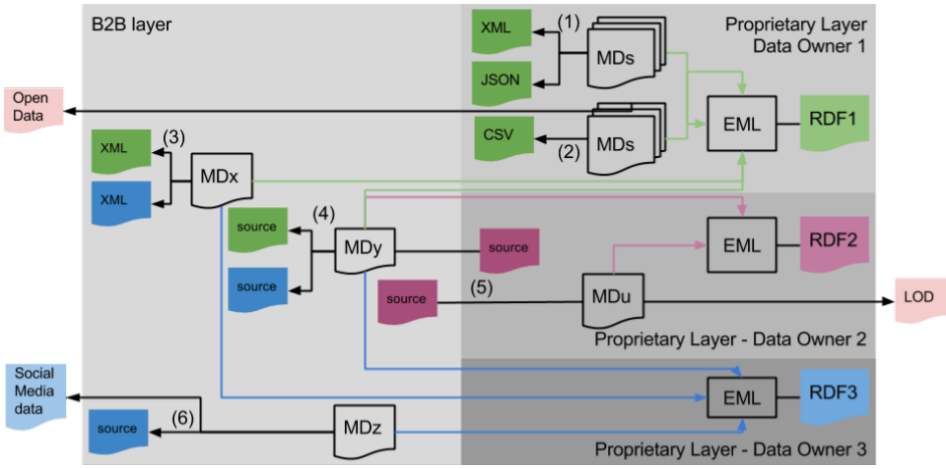


Figure 6.4: COMBUST *integration.*

This use case focuses on sharing identifiers by taking advantage of a unified knowledge framework. This way, data veracity and conformity are ensured, which contributes, on its own turn, to efficient data integration and management. This is achieved by adjusting the mapping rules applied to the original data to incrementally (i) integrate different data owner's heterogeneous data sources; (ii) enrich them with information derived from different data sources available on the Web, e.g., open data or social media; (iii) incorporate information from other data owners by sharing data over a B2B layer; and (iv) resolve mismatches, ambiguous alignments, and conflicting information derived from different data owners taking advantage of data provenance or crowdsourcing results.

The Combining Multi-trUst BusinesS daTa (COMBUST) project[37] was launched at the end of 2014 and aimed to create such a knowledge framework focused on data integration and veracity, exploring the foundations of Data-as-a-Service (DaaS) solutions. The COMBUST workflow was employed by different industrial companies participating in the project, namely GIM[38], Truvo Belgium[39], and Roularta Business Information[40]. The business partners collaborate on a common platform for data manipulation, and bring together the complementary domain knowledge. The companies' use cases focus on aggregation and disaggregation of high variety, multi-source, heterogeneous data. The mismatches and conflicts of entities alignments are validated by determining the data provenance and by amending data considering crowdsourcing results that form a dataset on their own.

6.3.1 COMBUST Model

The COMBUST data model is generated dynamically. To be more precise, each data owner defines or considers mapping rules which refer to the same (or not) data sources or may complement them with third-party proprietary. However, each one of the data owners specifies its own mapping rules which reflect on its own data model, and may publish (part of) the mapping rules and Linked Data set. For each heterogeneous data source, its data owner can define the mapping rules uniformly, i.e., using the same language, to generate the corresponding Linked Data set independently of the source's type or format. All possible alternatives are depicted in Figure 6.4 and explained in details below.

A data owner, e.g., *Data Owner 1* in Figure 6.4, generates its own Linked Data set, *RDF 1*, from data originally in different formats. To be more precise, the first set of mapping rules, Figure 6.4 ①, are defined taking into consideration only certain data sources whose data is in XML and JSON format and these mapping rules are not shared with other partners. As the mapping rules are defined for both data sources, cross-references are established within the mapping rules and unique IRIs are assigned for equivalent instances of the same real-world entities. This happens because the pattern that generates an IRI is uniquely defined and this mapping rule is referred every other time the same resource appears. Thus, modifications to the patterns, or data values appearing in the patterns that generate the IRIs, are propagated to every other reference, keeping the different views synchronized. This way duplicates generation is avoided, and data is modeled integrally and annotated with the same schema (vocabularies and ontologies).

The *RDF 1* is enriched with data derived from a data source whose data is in CSV format, Figure 6.4 ②. This RDF representation is generated in combination with open data and even though the mapping rules, MD_{χ}, were defined subsequently, they can be executed altogether with the previous ones.

The *RDF 1* is further enriched with data derived from a data source whose data is in XML format, too. The latter's RDF representation is generated in alignment with data derived from a data source owned by another data owner, *Data Owner 3*, Figure 6.4 ③. Thus, this data is made available to *Data Owner 1* via the mapping rules, instead of referring to the already generated Linked Data set, *RDF 3*, that includes them. This way, the *Data Owner 1* may generate its own RDF representation of the *Data Owner 3*'s data. The latter mapping rules, and thus the corresponding Linked Data that these mapping rules generate, are made available to other data owners who participate in the same B2B layer.

In the same context, another data owner, *Data Owner 2* in Figure 6.4, may define its own mapping rules, Figure 6.4 ④, to semantically annotate one of its own data sources and generate its own Linked Data set, *RDF 2*. In this case, the data is aligned with data derived from data sources which belong to *Data Owner 1* and *Data Owner 3*. The mapping rules, MD_{y}, are made available to the other data owners of the B2B layer, and, thus, the generated Linked Data, may also be incorporated in the other two data owners Linked Data sets, namely *RDF 1* and *RDF 3* respectively. Moreover, the *RDF 2* is further enriched with

[37] COMBUST, https://www.imec-int.com/en/what-we-offer/research-portfolio/combust
[38] GIM, http://www.gim.be/
[39] Truvo,https://goudengids.prezly.com/
[40] Roularta Business Information, http://www.roularta-business-information.be/en/

data derived from another data source, Figure 6.4 ⑤, and whose RDF representation is obtained taking also into consideration Linked Data which appear in the Linked Open Data cloud, i.e., reusing existing identifiers wherever possible and generate new ones if same entities are not identified.

Last, another data owner, *Data Owner 3*, generates its own Linked Data set combining data derived from a certain data source that he owns with data derived from social media, Figure 6.4 ⑥, e.g., twitter. In the end, each data owner executes only relevant mapping rules, periodically or when the data is updated, to acquire a new version of its Linked Data set.

6.3.2 COMBUST Vocabulary

The following vocabularies are used to indicatively semantically annotate samples of COMBUST partners' data: ADMS, DCMI, FOAF, gr, OSLO, schema, SKOS, VCARD, LOCN, and DBO. All the ontologies apart from the Good Relations[41] (gr) were already introduced. Good Relations is a vocabulary for product, price, store, and company data. All the properties which were used are summarized in Table 6.5.

6.3.3 COMBUST Linked Data set

Each partner generates and publishes its own Linked Data which is proprietary and, thus, not entirely and freely available for sharing with other partners of the B2B layer, let alone with the broader public. Thus, there is no detailed information regarding each Linked Data set that was generated. Nevertheless, we share below the most frequent (validation) cases which were performed by the different partners.

Validation Cases We identified that data owners involved in a B2B layer are mainly in need of three validation cases: verification, enrichment, and completion. Each validation case is explained in details:

verification (`co:Verification`) This validation case occurs if there is an RDF triple which is required to be verified, or a certain resource has multiple values for a certain property which are considered to be in conflict and it is required to be verified the most accurate value. For instance, in the former case, the following RDF triple, i.e., the resource's label, is required to be verified:

```
_:triple1 a rdf:Statement ;
          rdf:subject ex:address1 ;
          rdf:predicate rdfs:label ;
          rdf:object "IBBT" ;
          co:requires co:Verification .
```

[41] Good Relations, `http://www.heppnetz.de/ontologies/goodrelations/v1#`

To the contrary, in the latter case, the resource has two labels whose values are different and it is required to be verified whether both of them are valid or only one of the two should remain.

```
ex:address1 rdfs:label "IBBT", "iMinds" ;
            co:requires co:Verification .
```

enrichment (`co:Enrichment`) This validation case occurs if there is a resource which is required to be enriched. The complementary data sources might have been identified and it is only required to be verified if the complementary data should be integrated, or it is required complementary data sources to be discovered. For instance, in the latter case, a resource, e.g., `ex:address1`, is required to be enriched (`ex:address1 co:requires co:Enrichment.`) but no candidate RDF triples exist, whereas, in the former case, the same resource may be enriched with the following RDF triples: `ex:address1 geo:lat "51.036402"` and `ex:address1 geo:lon "3.734794"` and it is required to be verified if they are accepted by the data owner.

completion (`co:Completion`) This validation case occurs if there are missing values for a certain resource's property. For instance, a resource, like the aforementioned, might not have a label at all and it is desired to complete the value, as below:

```
_:triple1 a rdf:Statement ;
          rdf:subject ex:address1 ;
          rdf:predicate rdfs:label ;
          co:requires co:Completion .
```

In the cases that verification is required when conflicting values occur for certain properties, common identifiers are not used in the first place. To the contrary,

```
_:triple1 a prov:Entity ;
          prov:wasDerivedFrom ex:document1.

_:triple2 a prov:Entity ;
          prov:wasDerivedFrom ex:document2.
```

but

```
ex:address1 rdfs:label "IBBT" .
ex:address2 rdfs:label "iMinds" .
```

Ideally, `ex:address1 owl:sameAs ex:address2`. But, given that the label's values are in conflict, an abstract IRI is assigned for all candidates resources that might specialize it. For instance,

```
ex:address1 co:specializationOf ex:AddressA .
ex:address2 co:specializationOf ex:AddressA .

co:specializationOf rdfs:subClassOf prov:specializationOf
```

6.3.4 COMBUST Workflow

The COMBUST platform is a trusted-peer network for data sharing. Every partner is a peer, acting both as data provider (server) and data consumer (client). The partners relied on the RMLEditor (Section 2.6) to define their own mapping rules, followed by the RMLValidator (Section 4.5) to validate them, while the RMLMapper (Section 3.4) was used to execute them and generate the corresponding Linked Data, its metadata and provenance (Section 5.5). Last, the generated Linked Data is published relying on a Liked Data Fragments server per partner. The partners used the RMLEditor as a service and each one of them installed the other parts of the workflow locally.

6.4 DBpedia

Even though DBpedia is a central interlinking hub of the Linked Open Data (LOD) cloud [24], its quality can still be improved both on schema and data level. In the past, it was determined that its errors and inconsistencies can be addressed by amending the (i) *DBpedia Extraction Framework* (DBpedia EF) [16], (ii) mapping rules, or (iii) Wikipedia itself [27]. The DBpedia EF generates Linked Data from Wikipedia, mainly from the *Wikipedia infobox templates*, after annotating them using the *DBpedia ontology*[42] [16]. The generated Linked Data is made available in the form of the well-known DBpedia Linked Data set, while the *mapping rules*, which specify how the DBpedia Linked Data set is generated from Wikipedia, and are executed by the DBpedia EF, are crowd-sourced and maintained at the *DBpedia mappings wiki*[43].

Several changes were applied to both the DBpedia EF and mapping rules so far [16], although, quality issues still persist in DBpedia [20, 21]. To this end, we applied the RML-based solutions to both DBpedia mapping rules, by validating them (RMLValidator, as described in Section 4.5), and DBpedia EF. The latter is achieved by applying fundamental changes to its core modules and incorporating the RMLMapper (as described in Section 3.4), to enable developing solutions that can address its quality issues, e.g., [3, 20]. The transition to the new solution was fulfilled in four iterative steps:

mapping rules translation The DBpedia mapping rules were translated from the original custom for DBpedia wikitext-based syntax into RML statements;

mapping rules validation The DBpedia mapping rules were translated in RML statements and validated against the DBpedia ontology and their quality was assessed;

transformations decoupling The data transformations and mapping rules which were embedded in the DBpedia EF were decoupled and the transformation functions formed a distinct and independent module, the DBpedia Parsing Functions[44].

mapping rules execution The RMLProcessor was integrated in the DBpedia EF to execute the RML statements of the DBpedia mapping rules.

[42] DBpedia ontology, `http://dbpedia.org/ontology/`
[43] DBpedia mappings, `http://mappings.dbpedia.org/index.php/Main_Page`
[44] `https://github.com/FnOio/dbpedia-parsing-functions-scala`

6.4.1 DBpedia Generation Limitations

The coupling among extraction, transformations, and mapping rules execution contributes to DBpedia EF's inadequacy to cope with the increasing demands for high quality Linked Data [17]. *Extraction*, is the activity of retrieving data from Wikipedia; *transformation*, of processing the extracted data values; and *mapping rules execution*, of applying semantic annotations to the retrieved and transformed data values. Due to this coupling, the current custom DBpedia EF has several limitations:

coupled and embedded mapping rules On the one hand, agents are limited to the set of mapping rules which are already defined and cannot easily introduce new ones, due to the 1-to-1 relationship between the mapping rules and their implementation. Adjusting them requires amending both the custom DBpedia mapping language and DBpedia EF. Some rules, such as transformation of values are, in some cases, more flexible to change. For instance, changing the `language` value from 'el' to 'nl', or converting a string to a URI by appending a namespace. Nevertheless, extending the language is not straightforward and is performed in an ad-hoc manner, while corresponding implementations are developed as custom solutions, hampering the DBpedia EF maintenance. On the other hand, there are still many mapping rules not supported at all, while not all RDF triples forming DBpedia can be adjusted, because not all of them are generated based on mapping rules. The DBpedia EF adds certain RDF triples automatically or generates a few of them based on hard-coded mapping rules whose execution is embedded in and triggered by the DBpedia EF.

coupled and embedded transformations Transformations over extracted data values are hard-coded in the DBpedia EF and executed at different places within the DBpedia EF, from *extraction* to *mapping* and RDF *generation*. For instance, the DBpedia EF, parses and extracts dates or units from certain formats and turns them to valid XSD date value or units respectively. If another date format or unit is encountered, its conversion is required to be implemented within the DBpedia EF. Overall, the DBpedia Linked Data set is restricted to certain transformations which cannot be easily amended, unless the DBpedia EF is amended, even if it is just desired to reuse one of the existing transformations, which on its own turn, is not so trivial, while incorporating new requires adjusting the DBpedia EF.

restriced and coupled ontology use Agents cannot use other schema(s), than DBpedia ontology, to annotate Wikipedia pages because the DBpedia EF selects the corresponding parser based on the (i) mapping template use and (ii) selected ontology term. For instance, certain properties trigger the *GeoCoordinate parser*, whereas others trigger the *Data parser*. If an ontology term is not added to the DBpedia ontology, it cannot be used. Thus, other vocabularies, apart from certain, such as `dcterms`[45] or `foaf`[46], cannot be used unless imported into the DBpedia ontology and adjustments are applied to the DBpedia EF to process them and generate Linked Data. The assigned data type is also dependent on the used ontology term.

[45] DCTerms, `http://purl.org/dc/terms/`
[46] FOAF, `http://xmlns.com/foaf/0.1/`

Therefore, we adjusted the current DBpedia EF and provided a general-purpose and more sustainable framework, based on the RMLMapper that replaces the current solution, decouples extraction, transformations and conditions from DBpedia EF, and enables generating high quality Linked Data for DBpedia. All DBpedia mapping rules and transformations, were decoupled from the DBpedia EF and declaratively described. All template functionalities were also extracted and gathered into an independent module[47] and were formally described relying on RML's extension for data processing and transformations.

6.4.2 DBpedia Workflow

DBpedia Mapping Rules Translation DBpedia stem originally from Wikipedia, while its schema is derived from the set of DBpedia classes and properties specified by the DBpedia mapping rules, the result of a *collaborative mapping approach*. Its mapping rules are maintained and edited through the DBpedia mappings wiki[48], using the same wikitext syntax as Wikipedia to define the mapping rules.

To take advantage of an RML-based solution, the DBpedia mapping rules are automatically translated to RML statements and subsequently assessed using the RMLValidator. As RML is highly scalable towards other structures and formalizations, we introduced *wikitext serialisation* as a new Reference Formulation (ql:wikitext). 674 distinct mapping documents for English, 463 for Dutch and a total of 4,468 for all languages supported in the DBpedia mappings wiki were translated to RML statements[49] [5]. An example DBpedia mapping rule follows for a part of the Infobox Person[50]

```
{{TemplateMapping
| mapToClass = Person
| mappings =
        {{PropertyMapping | templateProperty = name | ontologyProperty = foaf:name }}
        {{PropertyMapping | templateProperty = birth_date | ontologyProperty = birthDate }}
        {{PropertyMapping | templateProperty = birth_place | ontologyProperty = birthPlace }}}}
```

and its corresponding RML mapping rules, after being translated to RML statements are:

```
<http://mappings.dbpedia.org/server/mappings/en/Infobox_person>
    rr:subjectMap [ rr:class dbpedia:Person ; rr:termType rr:IRI ;
        rr:constant "http://dbpedia.org/resource/Template:Infobox_person" ] ;
    rr:predicateObjectMap [ rr:predicate dbpedia:birthPlace ;
        rr:objectMap [ a rr:ObjectMap ;  rml:reference "birth_place". ] ] .
```

The translation to RML statements was fulfilled in two steps:

mapping rules translations The DBpedia mapping rules in wikitext syntax were translated in RML statements and are available at `http://mappings.dbpedia.org/server/mappings/en/pages/rdf/`.

embedded rules translation The mapping rules which were embedded in the DBpedia EF were expressed as RML statements (as described in Section 2.4) and the data

[47] DBpedia Parsing Functions, `https://github.com/FnOio/dbpedia-parsing-functions-scala/`

[48] DBpedia mappings, `http://mappings.dbpedia.org`

[49] `https://github.com/dbpedia/extraction-framework/blob/master/server/src/main/scala/org/dbpedia/extraction/server/resources/RMLMapping.scala`

[50] `http://mappings.dbpedia.org/index.php?title=Mapping_en:Infobox_person&action=edit`:

transformations as FnO statements (as described in Section 2.5.2). Every mapping template was assigned its own function, and we also added basic functions [9]. The corresponding RML mapping rules are available at `http://mappings.dbpedia.org/rml_mappings-201705.zip`.

A temporary extension in the DBpedia EF was built to automate the translation[51] of the custom DBpedia mapping rules to RML. This extension builds on top of the RMLModel, and can be run both within the DBpedia EF and standalone. The DBpedia EF loads the original custom DBpedia mapping rules, represents them in the DBpedia EF data structures and based on these, RML mapping rules are automatically generated. Each template is translated into corresponding RML statements.

The original DBpedia mapping templates mainly build on top of a core template, although different templates emerged to cover specific cases, such as templates for geocoordinates and date intervals. Therefore, each extension and edge case should be covered and corresponding RML mapping rules should be automatically generated. Moreover, each parameter's underlying functionality, and the mapping rules and data transformations which were embedded in the DBpedia EF were all described declaratively. Nevertheless, a few mapping rules were not translated deliberately, because they (i) were deprecated, (ii) were introduced only for very rare or specific cases, or (iii) did not produce sustainable URIs.

Declaratively defining every mapping rule causes an overhead when editing. However, even though it is convenient that part of the DBpedia Linked Data set is generated automatically by the DBpedia EF, the complete lack of control over what is actually generated or how the data values are transformed, it might often be the cause for deteriorating the DBpedia Linked Data set quality [26, 27].

Search:

language	mapping	predicate	expected	existing
en	mapping	predicate	expected	existing
en	Infobox_Prime_Minister-elect (edit)	militaryCommand (edit)	MilitaryPerson (edit hierarchy)	PrimeMinister (edit hierarchy)
en	Infobox_Prime_Minister-elect (edit)	militaryBranch (edit)	MilitaryPerson (edit hierarchy)	PrimeMinister (edit hierarchy)
en	Infobox_President (edit)	president (edit)	Organisation (edit hierarchy)	President (edit hierarchy)
en	Infobox_President (edit)	otherParty (edit)	OfficeHolder (edit hierarchy)	President (edit hierarchy)
en	Infobox_President (edit)	militaryUnit (edit)	MilitaryPerson (edit hierarchy)	President (edit hierarchy)
en	Infobox_President (edit)	militaryRank (edit)	MilitaryPerson (edit hierarchy)	President (edit hierarchy)
en	Infobox_President (edit)	militaryCommand (edit)	MilitaryPerson (edit hierarchy)	President (edit hierarchy)
en	Infobox_President (edit)	militaryBranch (edit)	MilitaryPerson (edit hierarchy)	President (edit hierarchy)

Showing 1 to 15 of 301 entries (filtered from 2,287 total entries) Previous 1 2 3 4 5 … 21 Next

Figure 6.5: *Screenshot of a violations list presented to the DBpedia community. For every violating mapping rule, the predicate with the existing RDF term, according to the corresponding DBpedia mapping, and the expected value, according to the DBpedia ontology are presented.*

DBpedia Mapping Rules Validation The DBpedia mapping rules are applied repeatedly to Wikipedia pages to generate DBpedia. Consequently, if they contain inaccuracies, same violations are repeated in its Linked Data set [3, 5]. Therefore, we incorporated in the

[51] `https://github.com/dbpedia/extraction-framework/tree/rml/server/src/main/scala/org/dbpedia/extraction/server/resources/rml`

DBpedia validation workflow, our RML-based solution for applying schema validation to the RML mapping rules of DBpedia (as described in Section 4.3) and, thus, assessing the quality that a Linked Data set will eventually have. We used the RML mapping rules of DBpedia because the quality of wikitext-based mapping rules cannot be assessed directly, and certainly not in the same way as the resulting Linked Data. This occurs because the original mapping rules are not expressed in a machine-processable format (they do not form Linked Data).

To systematically validate the RML statements for the DBpedia mapping rules and have up-to-date reports, we created a script[52] to trigger the RMLValidator (Section 4.5) to validate all DBpedia mapping rules and export the results as a JSON file. The assessment and report generation is automated, streamlined, and frequently executed. The DBpedia community uses the violations list as feedback to correct violating mapping rules or enhance the DBpedia ontology and, thus, improves the dataset's quality. The subsequent DBpedia releases rely on these results[53] and other solutions already build on top of this work to improve DBpedia dataset quality, such as [20].

The validation results are presented to the DBpedia community via a user friendly interface[54] and visualized as shown in Figure 6.5. The community can directly contribute to improve the DBpedia mapping rules and, thus, the DBpedia Linked Data set itself. Once a DBpedia mapping rule or ontology term is updated, a new validation round can be triggered, assessing if new violations were introduced.

DBpedia Linked Data Generation The DBpedia EF generation procedure is *data-driven*. Namely, Wikipedia pages are retrieved and their data values are extracted and considered each time to perform the applicable mapping rules. When an infobox template is found in a Wikipedia page, the corresponding RML mapping rules are identified, among all RML mapping rules loaded in memory by the RML-MapDocHandler[55], since the generation process was initially triggered. The RMLMapper (Section 3.4) is triggered with the Wikipedia page and the relevant RML mapping rules as input and the generated Linked Data are eventually forwarded to an output sink.

6.4.3 DBpedia Linked Data subsets

We indicatively generated Linked Data for 16,244,162 English Wikipedia pages (version 20170501[56]), both with the current DBpedia EF and RMLProcessor. The former DBpedia EF yielded 62,474,123 RDF triples in total, while the new 52,887,156. In terms of entities, 4,747,597 are extracted with the former DBpedia EF, whereas 4,683,709 entities are extracted with RMLProcessor, reaching 98% coverage.

[52] https://github.com/AKSW/RDFUnit/blob/master/rdfunit-examples/src/main/java/org/aksw/rdfunit/examples/DBpediaMappingValidator.java

[53] https://github.com/dbpedia/mappings-tracker/issues/57

[54] DBpedia mapping rules validation interface, http://mappings.dbpedia.org/validation

[55] https://github.com/RMLio/RML-MapDocHandler

[56] https://dumps.wikimedia.org/enwiki/20170501/enwiki-20170501-pages-articles.xml.bz2

The datasets differ because certain mapping rules were *deliberately* omitted, due to the following reasons: (i) *unsustainable URIs*; the DBpedia EF itself generates intermediate entities whose URI is generated relying on an iterator, which, though, does not generate sustainable identifiers. Thus, those mapping rules were temporarily omitted. (ii) *custom DBpedia datatypes*; the DBpedia EF generates additional RDF triples with custom DBpedia datatypes for units of properties. We omitted them from the RML mapping rules because there is still discussion on whether these should be included or not. (iii) *RDF triples added by the DBpedia EF;* these are RDF triples generated from a value of an infobox, if the article's text contains a URI with this value. However, the RMLProcessor does not support referencing the articles text, thus generating those RDF triples is also temporarily omitted.

The mapping rules, being Linked Data, can be (automatically) updated and other semantic annotations can be applied or other Linked Data can be generated from Wikipedia. For instance, relying on the DBpedia mapping rules and the alignment of DBpedia with `schema.org`[57], we translated the RML mapping rules for DBpedia, and generated other Linked Data with `schema.org` annotations[58] by generating a new RML mapping file with *schema.org* annotations, based on the mapping rules with the DBpedia ontology annotations. In our exemplary case, we based on *Infobox_person* mapping template. An extraction was done over 1,6244,162 pages, 191,288 Infobox_persons were found and 1,026,143 RDF triples were generated: 179,037 RDF triples were generated with the property `schema:name`, 54,664 with `schema:jobTitle`, 23,751 with `schema:nationality`, 144,907 with `schema:birthPlace`, and 139,488 with `schema:birthDate`.

6.5 CEUR-WS

A promising means to foster and accelerate high quality Linked Data generation is the organization of *challenges*: competitions during which participants complete tasks with innovative solutions that are then ranked in an objective way to determine the winner. The Semantic Web Evaluation Challenges series[59], co-located with the ESWC Semantic Web Conference[60], co-ordinates such challenges which compare their solutions based on their output, namely the produced Linked Data set. The Semantic Publishing Challenge is one of these challenges. Its goal is to extract data from heterogeneous sources on scholarly publications (*semantic publishing*), and generate Linked Data to be further exploited.

Semantic publishing is defined by Shotton [25] as "*the enhancement of scholarly publications by the use of modern Web standards to improve interactivity, openness and usability, including the use of ontologies to encode rich semantics in the form of machine-readable* RDF *metadata*". According to [19], extracting, annotating, and sharing scientific data (namely standalone research datasets, data inside documents, as well as metadata about datasets and documents) and then building new research efforts on them, can lead to a data value chain producing value for the scholar and Semantic Web community. On

[57] Schema.org, `http://schema.org`

[58] DBpedia with schema.org annotations, `http://mappings.dbpedia.org/person_schema.dataset.ttl.bz2`

[59] CEUR-WS.org, `https://github.com/ceurws/lod/wiki`

[60] `http://eswc-conferences.org/events`

the one hand, the scholar community benefits from a challenge that produces data, as the challenge results in more data and in data of higher quality being available to the community to exploit. On the other hand, the Semantic Web community benefits: participants optimize their tools towards performance in this particular challenge, but such optimisations may also improve the tools in general. Once such tools are reused, any other dataset benefits from their advancements, because the processes producing them has been improved. However, bootstrapping and enabling such value chains is not easy.

For this use case, we (i) introduce the Semantic Publishing Challenge (Section 6.5.1) [4, 7, 12, 15], (ii) describe our proposed solution [1, 10] (Section 6.5.2), and (iii) the other solutions (Section 6.5.3), and (iv) analyze and compare all other solutions (Section 6.5.4) that participated in the same task [7]. Our RML-based solution participated in the first edition of the Semantic Publishing challenge. To address the challenge of semantically annotating the content of HTML pages, we exploited RML's extensibility to other data formats. Moreover, we thoroughly analyze and compare the other solutions that participated in the different Semantic Publishing Challenge's editions. Our study provides insights regarding different approaches that address the same task, namely it acts as a benchmark for different solutions against a common challenge. Both the scholarly and broader Linked Data community may benefit from looking into our results. Other data owners may find details on different approaches dealing with the same problem and the results they produce, and they can determine their own approach for a similar case or even launch a corresponding challenge to choose the best performing tool.

6.5.1 Semantic Publishing Challenge

The Semantic Publishing Challenge provides as input scholarly papers and queries in natural language. Participants semantically annotate information derived from these papers and publish them as a Linked Data set that could be used to answer queries. The best performing approach was identified automatically by comparing the output of the queries in the produced datasets against a gold standard, and by measuring precision and recall. The same motivation, structure and evaluation procedure have been maintained in each edition, with some improvements and extensions. Detailed reports for each edition were published separately by Lange and Di Iorio [15], Iorio et al. [12], and **Anastasia Dimou** et al. [4] and an overall analysis of all editions, including lessons learnt and best practices by **Anastasia Dimou** et al. [7].

All editions had two main tasks (Task 1 and Task 2). For Task 1, the participants were asked to extract information from selected CEUR-WS.org workshop proceedings volumes. The input files were HTML tables of content using different levels of semantic markup. For Task 2, the input dataset included XML-encoded research papers in 2014, derived from the PubMedCentral[61] and Pensoft Open Access archives[62], and PDF research papers in 2015 and 2016, derived from the CEUR-WS.org workshop proceedings volumes. The participants were asked to extract data about citations, affiliations, and fundings indicatively. For each task and year, two datasets were published: (i) a training dataset (TD) on which

[61] PubMedCentral, https://www.ncbi.nlm.nih.gov/pmc/
[62] http://pensoft.net/index.php

the participants could test and train their extraction tools and (ii) an evaluation dataset (ED) made available a few days before the final submission and used as input for the final evaluation. As our solution participated in Task 1 of the Semantic Publishing Challenge, only this task will be further presented and its submissions will be discussed and compared (Section 6.5.4).

Training and Evaluation dataset for Task 1 The CEUR-WS.org workshop proceedings volumes served as the data source for Task 1 training and evaluation datasets in all editions of the Semantic Publishing Challenge. They were represented in different formats and at different levels of encoding quality and semantics. An HTML 4 index page[63] links to all workshop proceedings volumes which have HTML tables of contents and contain PDF or PostScript full texts. A mixture of different HTML formats (no semantic markup at all, different versions of microformats, RDFa) were chosen for both training and evaluation datasets. The training dataset comprised all volumes of several workshop series, including, e.g., Linked Data on the Web workshop (LDOW) at the WWW conference, and all workshops of some conferences, e.g., of several editions of ESWC. In 2014 and 2015, the evaluation dataset was created by adding further workshops on top of the training dataset. To support the evolution of extraction tools, the 2015 and 2016 training datasets were based on the previous years training and evaluation datasets union.

6.5.2 RML-based Workflow

To participate in the Semantic Publishing Challenge, our RML-based solution was extended to support input data in HTML format. To achieve that both the set of reference formulations that the mapping language is aligned with and the processor that executes the mapping rules were extended. Another reference formulation was included, the CSS3 selectors [28] which are patterns that match against elements in a tree. CSS3 selectors are optimized for use not only with HTML, e.g., for cascading styles or jQuery[64], but also for data sources in XML format, besides XPath. Even though the latter was already among the aligned and supported reference formulations, it was not used for CEUR-WS.org workshop proceedings volumes, because it requires the HTML pages to be valid XML documents and this is not always the case with CEUR-WS.org Web pages. Moreover, the first version of the RMLProcessor implementation[65] was extended to generate Linked Data derived from data in HTML format.

Our solution participated in two editions of the Semantic Publishing Challenge: the 2014 and 2015 editions. The mapping rules were manually specified, namely RML statements were written by hand. The mapping rules were assessed for their quality, using the RMLValidator, but only for the 2015 edition. Our 2015 solution built on the 2014 submission. Nevertheless, (i) a number of shortcomings were addressed, and (ii) the generated Linked Data set was assessed for its quality, relying on the RMLValidator. This way, the generated Linked Data set had higher quality as reflected in the query results.

[63] `http://ceur-ws.org/`
[64] jQuery, `http://jquery.com`
[65] `https://github.com/mmlab/RMLProcessor`

The mapping rules and result Linked Data set for the 2014 submission can be found at `http://rml.io/spc/spc.html`, and for the 2015 submission can be found at `http://rml.io/data/SPC2015/`. Moreover, we experimented with incorporating cleansing elements in the mapping language, as it was necessary to further process the extracted values. In more details:

2014 edition The first version of the RMLMapper was used and data cleansing was performed relying on regular expressions defined within the reference to the input data.

2015 edition The second version of the RMLMapper was used, data cleansing was performed relying on regular expressions defined in experimental attributes that extended RML, and mapping rules and dataset quality assessment was performed to increase even more the final Linked Data set quality.

CEUR-WS.org Vocabulary The vocabularies used to describe the domain were selected to be aligned with the annotations provided in volumes that already included RDFa annotations and considering additional vocabularies relevant to the domain as listed at `http://linkeduniversities.org/lu/index.php/vocabularies/`.

To be more precise, the following vocabularies were used: BIBO[66], DCMI [20], FOAF[67], FaBiO[68], event[69], and SWRC[70]. The Bibliographic Ontology (BIBO with `bibo` as its prefix), can be used as a document classification ontology, or simply as a way to describe any kind of bibligraphic document. The Friend of a Friend (FOAF with `foaf` as its prefix), is broadly used to describe people. The FRBR-aligned Bibliographic Ontology (FaBiO with `fabio` as prefix) is an ontology for recording and publishing bibliographic records of scholarly endeavours. The event Ontology (with `event` as prefix) is centered around the notion of *event* and is used in a wide range of context from talks in a conference, to the description of a concert. Last, the Semantic Web for Research Communities (SWRC with `swrc` as its prefix) is used by the Semantic Web Dog Food[71] (SWDF) to annotate data related to papers presented in Semantic Web conferences.

CEUR-WS.org Workflow The Linked Data generation and publication workflow followed for this use case consisted of three steps in 2014 and four steps in 2015. In detail:

CEUR-WS.org Mapping Rules Definition Both in 2014 and 2015, mapping rules were manually defined, using RML statements, while the references to the input data were specified relying on CSS3 selectors. Data cleansing was performed relying on regular expressions whenever it is required to be more selective over the returned values. For instance, a reference to `h3 span.CEURLOCTIME` returns, for the aforementioned example, `Montpellier, France, May 26, 2013`, and, as there

[66] BIBO, `http://purl.org/ontology/bibo/`
[67] FOAF, `http://xmlns.com/foaf/0.1/`
[68] FaBiO, `http://purl.org/spar/fabio/`
[69] event, `http://purl.org/NET/c4dm/event.owl`
[70] SWRC, `http://swrc.ontoware.org/ontology#`
[71] SWDF, `http://data.semanticweb.org/`

is no further HTML annotation, regular expressions are used to select parts of the returned value to be mapped separately (e.g., city). Nevertheless, in 2014, the regular expressions were defined within the variables for the references to the input data, whereas, in 2015, they were specified in special attributes which were introduced experimentally. To be more precise, the following properties were defined to further process the extracted values from the input source: `rml:process`, `rml:replace` and `rml:split`. In the aforementioned example, `rml:process` specifies a regular expression, e.g., `([a-zA-Z]*), [a-zA-Z]*, [a-zA-Z]* [0-9]*, [0-9]*`, and `rml:replace` specifies the part of the data value that will be used for a certain mapping rule, e.g., $1, for the aforementioned case to get the city "Montpellier". In the end, none of the two approaches was adopted by RML, but cleansing is performed relying on FnO statements [9].

CEUR-WS.org Mapping Rules Validation The mapping rules were assessed using the RMLValidator, but only for the 2015 submission, and were refined [3]. 12 violations were identified in the 2014 edition and 10 of them could already be detected at the mapping rules assessment. Most of them, i.e., 7 out of the 12, were *domain-level* violations, annotating, for instance, resources of type `bibo:Volume` with properties for `bibo:Document` or `madsrdf:Address` for specifying the city, implying inadvertently that resources are both *Documents* and *Addresses*. The rest of the detected violations were related to contradicting datatypes. For instance, incorrectly specifying a datatype as `xsd:gYear`, while it is expected to be `xsd:string`. The 2015 submission mapping rules and the generated Linked Data set were assessed and do not contain any violations.

CEUR-WS.org Linked Data Generation The CEUR-WS.org Linked Data set was generated using the first version of the RMLProcessor which was extended to support the new reference formulation and data transformation properties;

CEUR-WS.org Linked Data Publication The result Linked Data set was published, using the DataTank (TDT). The DataTank is a RESTful data management system. It enables publishing data stored in text based files, such as XML, binary structures or relational databases into Web readable formats. Besides publishing data, the DataTank also allows to publish (templated) SPARQL queries whose variables' value is defined at runtime. Such queries fitted well in the challenge's needs.

6.5.3 Solutions Description

There were four distinct solutions for Task 1 in 2014 and 2015 editions of the Semantic Publishing Challenge. Both generic and ad-hoc solutions, as well as different methodologies and approaches were submitted. Nevertheless, solutions tend to converge. Each distinct solution is described in this subsection.

Solution 1.1 Kolchin and Kozlov [13], Kolchin et al. [14] submitted a case-specific crawling based approach for addressing Task 1. Their solution relies on an extensible template-dependent crawler that turns the system tolerant to invalid HTML, and uses sets of special

predefined templates based on XPath and regular expressions to extract the content from CEUR-WS.org workshop proceedings volumes and generate its semantically enhanced representation. The output is then processed to merge resources using fuzzy-matching. This solution improved its precision and richness of the data model in 2015.

Solution 1.2 **Anastasia Dimou** et al. [1] and Heyvaert et al. [10], exploited a generic approach for generating Linked Data from heterogeneous data based on RML language (Chapter 2) and RMLMapper (Chapter 3) which is described in more detail in Section 6.5.2. This solution also improved its precision and accuracy of the data model in 2015.

Solution 1.3 Ronzano et al. [22, 23] designed a case-specific solution that relies on chunk- and sentence-based Support Vector Machine (SVM) classifiers which are exploited to semantically characterize parts of CEUR-WS.org workshop proceedings volumes textual contents. Thanks to a pipeline of text analysis components based on GATE Text Engineering Framework[72], each HTML page is characterized by structural and linguistic features. These features are exploited to train the classifiers on the ground-truth provided by the subset of CEUR-WS.org proceedings with microformat annotations. A heuristic-based annotation sanitizer is applied to fix classifiers imperfections and interlink annotations. The generated Linked Data set is extended with information retrieved from external resources.

Solution 1.4 Milicka and Burget [18] relied on an application of the FITLayout framework[73]. It combines different page analysis methods: layout analysis and visual and textual feature classification to analyze the rendered pages, rather than their code. The solution is generic but requires case-specific actions in certain phases, e.g., model building.

6.5.4 Solutions Comparison

Table 6.7 provides details about the methodology, approach and implementation each solution followed. Table 6.8 summarizes the model and vocabularies, whereas Table 6.9 provides statistics regarding the dataset schema/entities and triples/size each solution produced. Last, Table 6.11 summarizes the data model and Table 6.10 the number of instances extracted and annotated per concept for each solution.

Approaches Kolchin et al. and Ronzano et al. relied on tools developed specifically for this task, whereas Dimou et al. and Milicka et al. relied on generic solutions which can be adapted to other domains by only adjusting task-specific templates or rules. In the latter case, Dimou et al. abstracts the extraction rules from the implementation, whereas Micka et al. keeps them inline with the implementation.

[72] GATE, https://gate.ac.uk/

[73] FITLayout, http://www.fit.vutbr.cz/~burgetr/FITLayout/

Even though the solutions were methodologically different, four approaches prevailed: (i) *structure-based* (relying on the HTML code/structure), (ii) *layout-based* (relying on the page layout), (iii) *linguistic-based*, and (iv) *presentation-based*. Three out of four solutions relied on structured-/layout-based approach. Only Ronzano et al. relied on a partially linguistic-based approach showing significantly lower precision and recall, compared to other tools, indicating that linguistic-based solutions are not enough, if not supported by a precise structure analysis. Even though the linguistic-based approach was considered a rather innovative way of dealing with Task 1, the evaluation showed that barely relying on a linguistic-based analysis might not be able to perform as well as a structure-based one.

Besides Ronzano et al., all other approaches considered rules to extract data values from HTML. Two considered CSS3 to define the rules, and one considered JAPE; the latter solution was based on crawling. All solutions used regular expressions at some stage.

Implementations As shown in Table 6.7, three implementations relied on Java-based implementations whereas the crawling-based one relied on Python. All relied on open-source tools, and MIT[74] was the most popular license, used by half of the approaches. Half of the implementations incorporated external services to accomplish the tasks.

Data models As shown in Table 6.11, all submissions converge with respect to the data model. They identify the same core concepts: *Conference*, *Workshop*, *Proceedings*, *Papers*, and *Person*. Nevertheless, a few submissions covered more details than others. For instance, Ronzano et al. captured different types of sessions: *Session*, *Keynote Session*, *Invited Session* and *Poster Session*. Moreover, it was observed that different submissions are influenced by each other. For instance, Milicka et al. model was inspired by Kolchin et al. model. This is a practice commonly observed in real Linked Data set modeling too. Nevertheless, the different submissions rarely agree upon the extracted information. For instance, some skip the extraction of wrong data. Overall, we observed significant differences with respect to the number of identified entities per category.

Vocabularies As shown in Table 6.8 there is a wide range of vocabularies and ontologies used to annotate scholarly data. Most submissions (re)used almost the same existing vocabularies. Three submissions used BIBO [66], and two used SWRC [70], and the Semantic Web Conference (*swc*) vocabulary[75]. Two submissions used one or more vocabularies of the Semantic Publishing and Referencing Ontologies[76] (*SPAR*).

Besides the domain-specific vocabularies and ontologies, all submissions used the Dublin Core vocabularies (*dc*[77] and *dcterms*[78]), eight the Friend of a Friend vocabulary[67]three

[74]MIT, `http://opensource.org/licenses/mit-license.html`
[75]SWC, `http://data.semanticweb.org/ns/swc/ontology#`
[76]SPAR, `http://www.sparontologies.net/`
[77]DC, `http://purl.org/dc/elements/1.1/`
[78]DCTerms, `http://purl.org/dc/terms/`

submissions used the DBpedia ontology[79] (*dbo*), two the VCard[80] (*vcard*) and *timeline*[81] ontologies, and one the *event*[82].

Even though all solutions used almost the same vocabularies, not all of them used the same vocabulary terms to annotate the same *entities*. To be more precise, all submissions converged on annotating *Persons* using the `foaf:Person` class and properly captured the *Conference* concept, even though often differently. For the other main concepts the situation was diverge, as shown in Table 6.10.

It is also interesting to note that, for the 2014 edition, most solutions used generic vocabulary terms, e.g., `swrc:Event`, `swc:Event` or `swc:OrganizedEvent` to annotate the data. However, in the 2015 edition, most solutions preferred more explicit vocabulary terms for the same concept, e.g., `swrc:Conference` and `bibo:Conference`, but they also maintained the more generic vocabulary terms for events. The same occurred with the *Paper* concept. The 2014 edition Linked Data sets were annotated using more generic vocabulary terms, e.g., `swrc:Publication` or even `foaf:Document`, whereas in 2015 more explicit vocabulary terms were preferred, such as `swrc:InProceedings` or `bibo:Article`; the former was adopted by three out of four submissions.

Overall, continuity helps submissions to improve with respect to both the implementation and the data model. In particular for the latter, the more familiar data owners get with (Linked) data, the more explicit they become with the annotations they use and the more they converge on the choices they make. Nevertheless, how different submissions extract particular properties reflects on the final data model.

Linked Data sets From the 2014 to the 2015 edition of the Semantic Publishing Challenge, we noticed that submissions improved the overall Linked Data set, not only the parts useful to answer the queries. For instance, all three solutions that participated in both 2014 and 2015 editions modified how they represented the data, and this resulted in corresponding improvements to the overall Linked Data set.

Indicatively, Dimou et al. addressed a number of shortcomings the previous tool's version had with respect to data transformations, which might have influenced their precision improvement. Dimou et al. also assessed their mappings' quality to verify the schema is valid with respect to the used vocabularies. To address the same issue and avoid inconsistencies in their dataset, Kolchin et al. preferred to align different ontologies' classes and properties, e.g., aligning BIBO to the SWRC ontologies.

As shown in Table 6.9, the submissions differ significantly with respect to the Linked Data set size. This happens for different reasons. For instance, the Linked Data set of Kolchin et al. consists of an extraordinary number of triples compared to other solutions due to two reasons: (i) each concept is annotated with at least two classes, turning one fourth of the Linked Data set to be type declarations; (ii) even annotations that indicate the resource type or property on a very low level are included, i.e., they include `rdfs:Class`,

[79] DBO, `http://dbpedia.org/ontology/`
[80] VCard, `http://www.w3.org/2006/vcard/ns#`
[81] timeline ontology, `http://purl.org/NET/c4dm/timeline.owl#`
[82] event ontology, `http://purl.org/NET/c4dm/event.owl#`

`rdfs:Property` annotations, which counts for almost 2,000 triples of the total Linked Data set. The Linked Data set of Milicka et al. also consists of a high number of triples again due to two reasons: (i) the Linked Data set contains triples describing the structure of the Web page, besides the actual content; (ii) a new URI is generated each time a concept appears in one of the CEUR-WS.org workshop proceedings volumes. For instance, *Ruben Verborgh* appears to have 9 URIs: `<http://ceur-ws.org/Vol-1034/#RubeniVerborgh>` for Vol-1034 or `<http://ceur-ws.org/Vol-1184/#RubeniVerborgh>` for Vol-1184.

Overall, we observed that the greatest variation with respect to the number of triples occurs because of different practices of assigning URIs: a few solutions reuse URIs across different proceedings volumes, others do not.

6.5.5 Discussion

The Open Data use case used the same vocabularies to semantically annotate the original data. Each data owner published its own Linked Data set, while the integration is based on the use of common vocabularies. The semantic annotations were defined by Semantic Web experts and they were validates manually.

The CEUR-WS.org use case uses the same vocabulary to semantically annotate multiple data sources of almost same structure and format. There is a single data owner, several data publishers and all data is aggregated in a single Linked Data set, while the integration occurs due to the same mapping rules being applied to all data. The mapping rules that determine the semantic annotations were defined by Semantic Web experts and manually checked in the first place whereas, in the next iteration,

The iLastic use case uses a vocabulary to semantically annotate both (semi-)structured and plain text data. There are multiple data owners but a single Linked Data set is generated as a certain data owner sees the data. The integration occurs by aligning the semantically enhanced representation of (semi-)structured and plain text which, on its own turn, occurs due to shared mapping rules.

The COMBUST use case uses multiple vocabularies to semantically annotate data. Its data owner considers its own data and complements them with data derived from data sources which belong to other data owners, as well as external data sources, such as open data or social media. Each data owner publishes its own data set.

Overall, the different use cases were explored from 2013 until 2017. Across the different use cases, various data structures and formats were involved either separately or in different combinations and the data sources were accessed using different interfaces, as shown in Table 6.1. The mapping rules were defined manually in the first use cases, relying on the RMLEditor later on or even automatically derived from existing mapping rules. In most use cases, apart from one, the mapping rules were validated with the RMLValidator and different versions of the RMLMapper were used to generate the corresponding Linked Data. Last, different publishing interfaces were used depending on the use case's needs.

References

[1] **Anastasia Dimou**, Miel Vander Sande, Pieter Colpaert, Laurens De Vocht, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. Extraction and Semantic Annotation of Workshop Proceedings in HTML using RML. In Valentina Presutti, Milan Stankovic, Erik Cambria, Iván Cantador, Angelo Di Iorio, Tommaso Di Noia, Christoph Lange, Diego Reforgiato Recupero, and Anna Tordai, editors, *Semantic Web Evaluation Challenge: SemWebEval 2014 at ESWC 2014, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers*, pages 114–119, Cham, 2014. Springer International Publishing. URL http://dx.doi.org/10.1007/978-3-319-12024-9_15.

[2] **Anastasia Dimou**, Laurens De Vocht, Geert Van Grootel, Leen Van Campe, Jeroen Latour, Erik Mannens, and Rik Van de Walle. Visualizing the Information of a Linked Open Data Enabled Research Information System. *Procedia Computer Science*, 33:245 – 252, 2014. ISSN 1877-0509. doi: http://dx.doi.org/10.1016/j.procs.2014.06.039. URL http://www.sciencedirect.com/science/article/pii/S1877050914008291.

[3] **Anastasia Dimou**, Dimitris Kontokostas, Markus Freudenberg, Ruben Verborgh, Jens Lehmann, Erik Mannens, Sebastian Hellmann, and Rik Van de Walle. Assessing and Refining Mappings to RDF to Improve Dataset Quality. In Marcelo Arenas, Oscar Corcho, Elena Simperl, Markus Strohmaier, Mathieu d'Aquin, Kavitha Srinivas, Paul Groth, Michel Dumontier, Jeff Heflin, Krishnaprasad Thirunarayan, and Steffen Staab, editors, *The Semantic Web – ISWC 2015*, volume 9367 of *Lecture Notes in Computer Science*, pages 133–149. Springer International Publishing, Oct 2015. doi: 10.1007/978-3-319-25010-6_8. URL http://link.springer.com/chapter/10.1007/978-3-319-25010-6_8.

[4] **Anastasia Dimou**, Angelo Di Iorio, Christoph Lange, and Sahar Vahdati. Semantic Publishing Challenge – Assessing the Quality of Scientific Output in Its Ecosystem". In Harald Sack, Stefan Dietze, Anna Tordai, and Christoph Lange, editors, *Semantic Web Challenges: Third SemWebEval Challenge at ESWC 2016, Heraklion, Crete, Greece, May 29 - June 2, 2016, Revised Selected Papers*, pages 243–254, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46565-4. doi: 10.1007/978-3-319-46565-4_19. URL http://dx.doi.org/10.1007/978-3-319-46565-4_19.

[5] **Anastasia Dimou**, Dimitris Kontokostas, Markus Freudenberg, Ruben Verborgh, Jens Lehmann, Erik Mannens, and Sebastian Hellmann. DBpedia Mappings Quality Assessment. In Takahiro Kawamura and Heiko Paulheim, editors, *Proceedings of the 15th International Semantic Web Conference: Posters and Demos*, volume 1690 of *CEUR Workshop Proceedings*, October 2016. URL http://ceur-ws.org/Vol-1690/paper97.pdf.

[6] **Anastasia Dimou**, Gerald Haesendonck, Martin Vanbrabant, Laurens De Vocht, Ruben Verborgh, Steven Latré, and Erik Mannens. iLastic: Linked Data Generation Workflow and User Interface for iMinds Scholarly Data. In *Proceedings of the Workshop on Semantics, Analytics, Visualisation: Enhancing Scholarly Data*, April 2017. (to appear).

[7] **Anastasia Dimou**, Sahar Vahdati, Angelo Di Iorio, Christoph Lange, Ruben Verborgh, and Erik Mannens. Challenges as Enablers for High Quality Linked Data: Insights from the Semantic Publishing Challenge. *PeerJ Computer Science*, 3: e105, January 2017. ISSN 2376-5992. doi: 10.7717/peerj-cs.105. URL https://peerj.com/articles/cs-105/.

[8] Diego Valerio Camarda, Silvia Mazzini, and Alessandro Antonuccio. Lodlive, exploring the web of data. In *Proceedings of the 8th International Conference on Semantic Systems*, I-SEMANTICS '12, pages 197–200, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1112-0. doi: 10.1145/2362499.2362532. URL http://doi.acm.org/10.1145/2362499.2362532.

[9] Ben De Meester, Wouter Maroy, **Anastasia Dimou**, Ruben Verborgh, and Erik Mannens. Declarative Data Transformations for Linked Data Generation: The Case of DBpedia. In Eva Blomqvist, Diana Maynard, Aldo Gangemi, Rinke Hoekstra, Pascal Hitzler, and Olaf Hartig, editors, *The Semantic Web: 14th International Conference, ESWC 2017, Portorož, Slovenia, May 28 – June 1, 2017, Proceedings, Part II*, pages 33–48, Cham, 2017. Springer International Publishing. ISBN 978-3-319-58451-5. doi: 10.1007/978-3-319-58451-5_3. URL http://dx.doi.org/10.1007/978-3-319-58451-5_3.

[10] Pieter Heyvaert, **Anastasia Dimou**, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. Semantically Annotating CEUR-WS Workshop Proceedings with RML. In Fabien Gandon, Elena Cabrio, Milan Stankovic, and Antoine Zimmermann, editors, *Semantic Web Evaluation Challenges: Second SemWebEval Challenge at ESWC 2015, Portorož, Slovenia, May 31 - June 4, 2015, Revised Selected Papers*, pages 165–176, Cham, 2015. Springer International Publishing. ISBN 978-3-319-25518-7. doi: 10.1007/978-3-319-25518-7_14. URL http://dx.doi.org/10.1007/978-3-319-25518-7_14.

[11] 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 Harald Sack, Eva Blomqvist, Mathieu d'Aquin, Chiara Ghidini, Paolo Simone Ponzetto, and Christoph Lange, editors, *The Semantic Web – Latest Advances and New Domains (ESWC 2016)*, volume 9678 of *Lecture Notes in Computer Science*, pages 709–723. Springer, 2016. ISBN 978-3-319-34129-3. doi: 10.1007/978-3-319-34129-3_43. URL http://dx.doi.org/10.1007/978-3-319-34129-3_43.

[12] Angelo Di Iorio, Christoph Lange, **Anastasia Dimou**, and Sahar Vahdati. Semantic Publishing Challenge – Assessing the Quality of Scientific Output by Information Extraction and Interlinking. In Fabien Gandon, Elena Cabrio, Milan Stankovic, and Antoine Zimmermann, editors, *Semantic Web Evaluation Challenges: Second SemWebEval Challenge at ESWC 2015, Portorož, Slovenia, May 31 - June 4, 2015, Revised Selected Papers*, pages 65–80, Cham, 2015. Springer International Publishing. ISBN 978-3-319-25518-7. doi: 10.1007/978-3-319-25518-7_6. URL https://doi.org/10.1007/978-3-319-25518-7_6.

[13] Maxim Kolchin and Fedor Kozlov. A Template-Based Information Extraction from Web Sites with Unstable Markup. In Valentina Presutti, Milan Stankovic, Erik Cam-

bria, Iván Cantador, Angelo Di Iorio, Tommaso Di Noia, Christoph Lange, Diego Reforgiato Recupero, and Anna Tordai, editors, *Semantic Web Evaluation Challenge: SemWebEval 2014 at ESWC 2014, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers*, pages 89–94, Cham, 2014. Springer International Publishing. doi: 10.1007/978-3-319-12024-9_11. URL https://doi.org/10.1007/978-3-319-12024-9_11.

[14] Maxim Kolchin, Eugene Cherny, Fedor Kozlov, Alexander Shipilo, and Liubov Kovriguina. CEUR-WS-LOD: Conversion of CEUR-WS Workshops to Linked Data. In Fabien Gandon, Elena Cabrio, Milan Stankovic, and Antoine Zimmermann, editors, *Semantic Web Evaluation Challenges: Second SemWebEval Challenge at ESWC 2015, Portorož, Slovenia, May 31 - June 4, 2015, Revised Selected Papers*, pages 142–152, Cham, 2015. Springer International Publishing. ISBN 978-3-319-25518-7. doi: 10.1007/978-3-319-25518-7_12. URL https://doi.org/10.1007/978-3-319-25518-7_12.

[15] Christoph Lange and Angelo Di Iorio. Semantic Publishing Challenge – Assessing the Quality of Scientific Output. In Valentina Presutti, Milan Stankovic, Erik Cambria, Iván Cantador, Angelo Di Iorio, Tommaso Di Noia, Christoph Lange, Diego Reforgiato Recupero, and Anna Tordai, editors, *Semantic Web Evaluation Challenge: SemWebEval 2014 at ESWC 2014, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers*, pages 61–76, Cham, 2014. Springer International Publishing. ISBN 978-3-319-12024-9. doi: 10.1007/978-3-319-12024-9_8. URL https://doi.org/10.1007/978-3-319-12024-9_8.

[16] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Kleef, Sören Auer, and Christian Bizer. DBpedia - a Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. *Semantic Web Journal*, 2014. URL http://jens-lehmann.org/files/2014/swj_dbpedia.pdf.

[17] Wouter Maroy, **Anastasia Dimou**, Dimitris Kontokostas, Ben De Meester, Ruben Verborgh, Jens Lehmann, Erik Mannens, and Sebastian Hellmann. Sustainable Linked Data Generation: The Case of DBpedia. In Claudia d'Amato, Miriam Fernandez, Valentina Tamma, Freddy Lecue, Philippe Cudré-Mauroux, Juan Sequeda, Christoph Lange, and Jeff Heflin, editors, *The Semantic Web – ISWC 2017: 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part II*, pages 297–313, Cham, 2017. Springer International Publishing. ISBN 978-3-319-68204-4. doi: 10.1007/978-3-319-68204-4_28. URL https://doi.org/10.1007/978-3-319-68204-4_28.

[18] Martin Milicka and Radek Burget. Information Extraction from Web Sources Based on Multi-aspect Content Analysis. In Fabien Gandon, Elena Cabrio, Milan Stankovic, and Antoine Zimmermann, editors, *Semantic Web Evaluation Challenges: Second SemWebEval Challenge at ESWC 2015, Portorož, Slovenia, May 31 - June 4, 2015, Revised Selected Papers*, pages 81–92, Cham, 2015. Springer International Publishing. ISBN 978-3-319-25518-7. doi: 10.1007/978-3-319-25518-7_7. URL https://doi.org/10.1007/978-3-319-25518-7_7.

[19] H. G. Miller and P. Mork. From Data to Decisions: A Value Chain for Big Data. *IT Professional*, 15(1):57–59, Jan 2013. ISSN 1520-9202. doi: 10.1109/MITP.2013.11.

[20] Heiko Paulheim. Data-Driven Joint Debugging of the DBpedia Mappings and Ontology. In Eva Blomqvist, Diana Maynard, Aldo Gangemi, Rinke Hoekstra, Pascal Hitzler, and Olaf Hartig, editors, *The Semantic Web: 14th International Conference, ESWC 2017, Portorož, Slovenia, May 28 – June 1, 2017, Proceedings, Part I*, pages 404–418, Cham, 2017. Springer International Publishing. ISBN 978-3-319-58068-5. doi: 10.1007/978-3-319-58068-5_25. URL http://dx.doi.org/10.1007/978-3-319-58068-5_25.

[21] Blake Regalia, Krzysztof Janowicz, and Song Gao. VOLT: A Provenance-Producing, Transparent SPARQL Proxy for the On-Demand Computation of Linked Data and its Application to Spatiotemporally Dependent Data. In Harald Sack, Eva Blomqvist, Mathieu d'Aquin, Chiara Ghidini, Simone Paolo Ponzetto, and Christoph Lange, editors, *The Semantic Web. Latest Advances and New Domains: 13th International Conference, ESWC 2016, Heraklion, Crete, Greece, May 29 – June 2, 2016, Proceedings*, pages 523–538, Cham, 2016. Springer International Publishing. ISBN 978-3-319-34129-3. doi: 10.1007/978-3-319-34129-3_32. URL https://doi.org/10.1007/978-3-319-34129-3_32.

[22] Francesco Ronzano, Gerard Casamayor del Bosque, and Horacio Saggion. Semantify CEUR-WS Proceedings: Towards the Automatic Generation of Highly Descriptive Scholarly Publishing Linked Datasets. In Valentina Presutti, Milan Stankovic, Erik Cambria, Iván Cantador, Angelo Di Iorio, Tommaso Di Noia, Christoph Lange, Diego Reforgiato Recupero, and Anna Tordai, editors, *Semantic Web Evaluation Challenge: SemWebEval 2014 at ESWC 2014, Anissaras, Crete, Greece, May 25-29, 2014, Revised Selected Papers*, pages 83–88, Cham, 2014. Springer International Publishing. ISBN 978-3-319-12024-9. doi: 10.1007/978-3-319-12024-9_10. URL https://doi.org/10.1007/978-3-319-12024-9_10.

[23] Francesco Ronzano, Beatriz Fisas, Gerard Casamayor del Bosque, and Horacio Saggion. On the automated generation of scholarly publishing linked datasets: The case of ceur-ws proceedings. In Fabien Gandon, Elena Cabrio, Milan Stankovic, and Antoine Zimmermann, editors, *Semantic Web Evaluation Challenges: Second SemWebEval Challenge at ESWC 2015, Portorož, Slovenia, May 31 - June 4, 2015, Revised Selected Papers*, pages 177–188, Cham, 2015. Springer International Publishing. ISBN 978-3-319-25518-7. doi: 10.1007/978-3-319-25518-7_15. URL https://doi.org/10.1007/978-3-319-25518-7_15.

[24] Max Schmachtenberg, Christian Bizer, and Heiko Paulheim. Adoption of the Linked Data Best Practices in Different Topical Domains. In Peter Mika, Tania Tudorache, Abraham Bernstein, Chris Welty, Craig Knoblock, Denny Vrandečić, Paul Groth, Natasha Noy, Krzysztof Janowicz, and Carole Goble, editors, *The Semantic Web – ISWC 2014: 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part I*, pages 245–260, Cham, 2014. Springer International Publishing. ISBN 978-3-319-11964-9. doi: 10.1007/978-3-319-11964-9_16. URL http://dx.doi.org/10.1007/978-3-319-11964-9_16.

[25] David Shotton. Semantic publishing: the coming revolution in scientific journal publishing. *Learned Publishing*, 22(2):85–94, 2009. ISSN 1741-4857. doi: 10.1087/2009202. URL http://dx.doi.org/10.1087/2009202.

[26] Dominik Wienand and Heiko Paulheim. Detecting Incorrect Numerical Data in DBpedia. In Valentina Presutti, Claudia d'Amato, Fabien Gandon, Mathieu d'Aquin, Steffen Staab, and Anna Tordai, editors, *The Semantic Web: Trends and Challenges: 11th International Conference, ESWC 2014, Anissaras, Crete, Greece, May 25-29, 2014. Proceedings*, pages 504–518, Cham, 2014. Springer International Publishing. ISBN 978-3-319-07443-6. doi: 10.1007/978-3-319-07443-6_34. URL https://doi.org/10.1007/978-3-319-07443-6_34.

[27] Amrapali Zaveri, Dimitris Kontokostas, Mohamed A. Sherif, Lorenz Bühmann, Mohamed Morsey, Sören Auer, and Jens Lehmann. User-driven Quality Evaluation of DBpedia. In *Proceedings of the 9th International Conference on Semantic Systems*, I-SEMANTICS '13, pages 97–104, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-1972-0. doi: 10.1145/2506182.2506195. URL http://doi.acm.org/10.1145/2506182.2506195.

[28] Tantek Çelik, Elika Etemad, Daniel Glazman, Ian Hickson, Peter Linss, and John Williams. Selectors Level 3. W3C Recommendation, W3C, September 2011. http://www.w3.org/TR/selectors/.

Table 6.1: *Use Cases summary*

	Open Data	**CEUR-Ws.org**	**iLastic**	**COMBUST**	**DBpedia**
Use Case					
theme	open data	semantic publishing	semantic publishing	business data (B2B)	general-purpose
duration	2013 - 2015	2014 - 2016	2015 - 2016	2014 - 2017	2016 - 2017
data format					
CSV	✓			✓	
HTML	✓	✓			
JSON	✓		✓		
XML	✓			✓	
wikitext					✓
data structure					
hierarchical	✓		✓	✓	
tabular	✓		✓	✓	
loose-coupled		✓			✓
data access					
file	✓	✓			
Web API		✓	✓	✓	
DB			✓	✓	
custom					DBpedia EF
Tooling editing					
manual	✓	✓			
RMLEditor			✓	✓	
automatic					✓
validation		✓	✓	✓	✓
processor					
v1.0	✓	✓			
v2.0		✓	✓	✓	
v3.0					✓
publishing					
data dumps	✓	✓			✓
LDF			✓	✓	
TDT		✓	✓		
Virtuoso	✓	✓	✓		
Modeling integration					
identifiers				✓	✓
interlinking			✓		
vocabulary	✓	✓			

Table 6.2: ***Input Data*** *used in the* EWI *Open* Data *use case*

	Beveren	Destelbergen	Ingelmunster	Knokke	Kortrijk	Roeselare
CSV		✓	✓		✓	✓
HTML				✓		
XML	✓	✓				

Table 6.3: ***Classes*** *&* ***properties*** *used to annotate the* EWI *Open* Data

	Beveren	Destelbergen	Ingelmunster	Knokke	Kortrijk	Roeselare
oslo:Activity		✓	✓		✓	✓
oslo:BasicAddress	✓	✓	✓		✓	✓
oslo:Channel	✓	✓	✓	✓	✓	✓
oslo:Code		✓	✓	✓	✓	✓
oslo:Membereship		✓	✓		✓	✓
oslo:Organization	✓	✓	✓	✓	✓	✓
oslo:Product	✓	✓	✓	✓	✓	✓
oslo:ResidenceObject		✓	✓		✓	✓
vcard:Cell		✓	✓		✓	✓
vcard:Contact	✓	✓	✓		✓	✓
vcard:Email		✓	✓		✓	✓
vcard:Fax		✓	✓		✓	✓
vcard:Work		✓	✓		✓	✓
adms:Identifier	✓	✓	✓	✓	✓	✓
dcterms:LPOJ	✓					
person:Person		✓	✓		✓	✓
skos:Concept	✓					

Table 6.4: ***Classes*** *&* ***properties*** *used to semantically annotate the iLastic data model*

Classes	**Properties - Bibo/bibTex/CERIF**	
cerif:Person	bibo:identifier	cerif:internalidentifier
cerif:OrganizationalUnit	bibo:abstract	cerif:linksToOrganisationUnit
cerif:Publication	bibo:issn	cerif:linksToPublication
cerif:Project	bibo:isbn13	cerif:name
	bibo:uri	cerif:title
	bibtex:howPublished	cerif:acronym
	dcterms:identifier	dcterms:issued
	foaf:familyName	foaf:givenName
	im:webOfScience	im:publicationCategory

Table 6.5: ***Classes*** *&* ***properties*** *used to annotate the COMBUST partners Linked Data sets*

Classes	Properties
adms:Identifier	adms:status
dcterms:Agent	dce:coverage
	dcterms:alternative
	dcterms:identifier
	dcterms:subject
foaf:Agent	foaf:homepage
foaf:Organization	foaf:name
gr:BusinessEntity	gr:legalName
	gr:vatID
oslo:Address	oslo:address
oslo:ExtendedAddress	oslo:addressArea
oslo:Building	oslo:addressId
oslo:ResidenceObject	oslo:building, oslo:buildingIdentifier
oslo:Organization	oslo:contact, oslo:contactLocation
oslo:Status	oslo:country
	oslo:establishmentDate, oslo:shutDownDate
	oslo:fullAddress
	oslo:geometry
	oslo:isBrunch
	oslo:mailingLocation
	oslo:legalFormCode
	oslo:locatorDesignator, oslo:locatorNumber
	oslo:poBox, oslo:postCode
	oslo:province
	oslo:thoroughfare
schema:Date	schema:busNumber
schema:Organization	schema:startDate, schema:endDate
	schema:telephone
	schema:vatID
skos:Concept	skos:broader, skos:narrower
skos:ConceptScheme	skos:inScheme
	skos:notation
	skos:prefLabel
vcard:Cell	vcard:email
vcard:Fax	vcard:hasTelephone
vcard:Tel	vcard:url
vcard:VCard	vcard:locality
sf:Point	locn:kind
	locn:postName
	dbo:capacity
	dbo:height
	dbo:openingDate
	dbo:related
	dbo:value

Table 6.6: ***Classes*** *&* ***properties*** *used to annotate* CEUR-WS*.org workshop proceedings volumes*

Classes	Properties
dcterms:Agent, foaf:Person	dcterms:creator
swrc:InProceedings, bibo:Document	dcterms:editor
bibo:Proceedings	dcterms:isPartOf
bibo:Conference	dcterms:hasPart
bibo:Series	event:sub event
bibo:Workshop	bibo:presentedAt
	fabio:supplement
	rdfs:seeAlso

Table 6.7: *Task 1 solutions for Semantic Publishing Challenge: their primary analysis methods, methodologies, implementations basis and evaluation results.*

	Solution 1.1	Solution 1.2	Solution 1.3	Solution 1.4
Primary analysis				
structure-based		✓	✓	
syntactic-based	✓			✓
linguistic-based			✓	
layout-based				✓
Methodology				
method	Crawling	Generic for abstracted mappings	Linguistic structural analysis	Visual layout multi-aspect content analysis
case-specific	✓		✓(partly)	✓(partly)
template-based	✓	✓		
NLP/NER			✓	✓
Implementation				
basis	n/a	RML	GATE	FITLayout
language	Python	Java	Java	Java, HTML
rules language	XPath	RML, CSS	JAPE	HTML,CSS
distinct code/rule		✓	✓	
RegEx	✓	✓	✓	✓
external services			✓	✓
open source	✓	✓		✓
license	MIT	MIT	–	GPL-3.0
Evaluation				
precision progress	11.1%	11.4%	10.7%	–
recall progress	11.3%	11.3%	10.9%	–
best performing	✓(2014)			✓(2015)
most innovative			✓(2014)	✓(2015)

Table 6.8: *Task 1 and 2 solutions: the vocabularies used to annotate the data.*

	Solution 1.1	Solution 1.2	Solution 1.3	Solution 1.4
bibo	✓	✓		✓
co			✓	
DBO	✓		✓	✓
DC	✓	✓	✓	✓
DCterms	✓			✓
event		✓		
FOAF	✓	✓	✓	✓
SKOS	✓			
SPAR		✓	✓	
BiRO			✓	
FaBiO		✓	✓	
FRBR			✓	
PRO			✓	
SWC	✓			✓
SWRC	✓	✓	✓	✓
timeline	✓			✓
vcard			✓	✓

Table 6.9: *Statistics about the produced dataset (Task 1 – 2014 and 2015 editions)*

	Solution 1.1		Solution 1.2		Solution 1.3		Sol 1.4
year	**2014**	**2015**	**2014**	**2015**	**2014**	**2015**	**2015**
dataset size	1.5M	25M	1.7M	7.2M	2.7M	9.1M	9.7M
# triples	32,088	177,752	14,178	58,858	60,130	62,231	79,444
# entities	4,770	11,428	1,258	11,803	9,691	11,656	19,090
# properties	60	46	43	23	45	48	23
# classes	8	30	5	10	10	19	6

Table 6.10: *Number of entities per concept for each solution (Task 1 – 2014 and 2015 editions)*

	Solution 1.1		Solution 1.2		Solution 1.3		Solution 1.4
year	**2014**	**2015**	**2014**	**2015**	**2014**	**2015**	**2015**
Conferences	21	46		46		5	47
Workshops	132	252	14	1,393	1,516	127	198
Proceedings	126	243	65	1,392	124	202	1,353
Papers	1,634	3,801	971	2,452	1,110	720	2,470
Persons	2,854	6,700	202	6,414	2,794	3,402	11,034

Table 6.11: *Statistics about the model (Task 1 – 2014 and 2015 editions)*

	Solution 1.1	Solution 1.2	Solution 1.3	Solution 1.4
	Conferences			
2014	swc:OrganizedEvent	swc:Event	swrc:Event	–
2015	swc:OrganizedEvent	bibo:Conference	swrc:Conference	swrc:ConferenceEvent
	Workshops			
2014	bibo:Workshop	swc:Event	swrc:Event	–
2015	bibo:Workshop	bibo:Workshop	swrc:Workshop	swrc:Section
	Proceedings			
2014	swrc:Proceedings	bibo:Volume	swrc:Proceedings	–
2015	bibo:Proceeding	bibo:Proceeding	swrc:Proceedings	swrc:Proceedings
	Papers			
2014	swrc:InProceedings	bibo:Article	swrc:Publication	–
2015	swrc:InProceedings, foaf:Document	swrc:InProceedings	swrc:Publication	swc:Paper
	Persons			
2014	foaf:Agent	foaf:Person	foaf:Person	–
2015	foaf:Person	foaf:Person	foaf:Person	foaf:Person

7

Conclusions

The main research question that guided my PhD thesis was:

> *How can we access and represent high quality domain level information from distributed heterogeneous sources in an integratable and uniform way?*

Therefore, this PhD thesis proposed complementary techniques, for generating high quality, semantically-enhanced, interrelated information in the form of Linked Data from (semi-)structured heterogeneous data, independently of their original structure and format.

As it was already mentioned, to address this, on the one hand, the representation aspect was required to be investigated:

Research Question #1 *How can we define how to combine and enrich data from different heterogeneous data sources and enrich this knowledge with new relationships?*

Research Question #2 *How can we assess, improve, and verify high quality semantically enhanced representations from heterogeneous data sources?*

The former research question (Research Question #1) was addressed by introducing the RML mapping language in Chapter 2. It is a novel approach which proves that there can be such a formalization that allows to uniformly define how to generate Linked Data from originally heterogeneous data. Our proposed solution efficiently improves a Linked Data set's integrity, its resources' interlinking, and indicates a well considered Linked Data generation policy, as the *per-format* and *per-source* approaches followed so far get surpassed.

The latter research question (Research Question #2) was addressed by introducing a methodology in Chapter 4 for assessing the mapping rules following the same methodology as for Linked Data and verifying that the generated Linked Data is of high quality. The assessment report points exactly to the root causes and can be actively used to refine the violating mapping rules. If violations are encountered, a methodology was proposed to refine the mapping rules until high quality Linked Data is generated. Fixing violations early avoids propagation where a violating mapping rule leads to many faulty RDF triples.

On the other hand, the accessing and executing aspects were also investigated too:

Research Question #3 *How can we enable accessing and retrieving data from distributed heterogeneous data sources on the Web (or not) in a uniform way?*

Research Question #4 *How can we generate, based on the aforementioned definitions, semantically enhanced representations from distributed heterogeneous data sources?*

The former research question (Research Question #3) was addressed by aligning RML with other vocabularies for describing datasets and services, leading to complete Linked Data generation and publication workflows, as it is described in Chapter 5. Machine-interpretable descriptions of data access remained independent so far. We proved that their alignment with uniform machine-interpretable mapping rules leads to a granular but robust solution which further facilitates the Linked Data generation and turns it in a complete workflow. Moreover, relying on machine interpretable descriptions of both data access interfaces and mapping rules, it is achieved to generate more accurate, consistent and complete metadata and provenance in an incremental and systematic way.

The latter research question (Research Question #4) was addressed by providing a theoretical background and corresponding implementation that allows to execute the aforementioned mapping rules and generate the desired Linked Data, as described in Chapter 3.

This is attested by evaluating the execution's performance with respect to accessing and executing related aspects and validation's results with respect to declaring related aspects. The former evaluation showed that our methodology is applicable for heterogeneous data with diverse sizes and hierarchical data structures of different depths, as well as when it is required to generate large Linked Data sets, such as DBpedia. Its execution time is in the same order of magnitude as for a custom implementation deliberately designed and optimized for a certain data source. The latter evaluation showed that our methodology is applicable to Linked Data sets without native mapping rules and large Linked Data sets, such as DBpedia. It was proven that assessing the mapping rules is more efficient in terms of computational complexity, and requires significantly less time to be executed compared to assessing the entire Linked Data set.

Besides the evaluations, our proposed approaches were successfully applied in different use cases and were adopted by large communities, such the DBpedia community, as well as industry. In this thesis, there were mentioned indicatively a few of the most prominent use cases which were related to the following domains: open government, scientific research, industry, and generic domain knowledge.

The results of this work open new horizons in the field of Linked Data generation and may boost further research with respect to different aspects which may improve or emerge.

An aspect that may be further investigated is the mapping rules definition. On the one hand, research is required to be conducted on graphical user interfaces that may facilitate the mapping rules editing. On the other hand, approaches that enable the automated generation of mapping rules also need to be investigated.

Furthermore, research may be conducted with respect to mapping rules validation, too. Optimizations and alternative approaches may be researched, not only with respect to mapping rules and Linked Data, but also with respect to the data values which are extracted from the data sources and the corresponding transformations which are applied.

Besides the mapping rules declaration, research may be conducted with respect to their execution. Currently, we investigated different factors that influence the mapping rules execution and provided a corresponding implementation. Nevertheless, there is room for investigating different optimizations which would allow to improve the execution's performance and, thus, Linked Data generation may be obtained faster.