Towards a Mapping Framework for the Tenders Electronic Daily Standard Forms

Eugeniu Costetchi[1], Alexandros Vassiliades[1] and Csongor I. Nyulas[1]

[1]*Meaningfy SARL, 61 route de Fischbach, L-7447, Lintgen, Luxembourg*

Abstract

Knowledge graphs are frequently built using declarative rules to bridge diverse data sources to a desired ontology and materialise them as RDF. The materialisation of the full knowledge graph may be a complex task when these data sources are extensive, making it unsuitable for an "on-demand" materialisation. In this paper, we present a methodology on how to map Public Procurement Data from the Tenders Electronic Daily website of the European Union by using RML, based on a innovative idea of mapping partitions. We map the aforementioned data into the eProcurement Ontology, which is a popular ontology when it comes to representing public procurement data. We also provide a method of evaluating the quality of the mapped data by using a mechanism that produces SPARQL queries based on the conceptual mapping of the Tenders Electronic Daily website data into the eProcurement Ontology. We then give an empirical evaluation over the quality of the produced data, and provide a detailed discussion on what the method presented in this paper has to offer.

Keywords

Knowledge Graph Generation, RDF Mapping Language (RML), Tenders Electronic Daily (TED), eProcurement Ontology (ePO), Public Procurement Data, Conceptual Mapping, Technical Mapping.

1. Introduction

A knowledge graph (KG), consisting of a relative simple knowledge organisation and linking of a usually very large number of resources represented in RDF, is a suitable knowledge representation structure for any knowledge-based system. While the design and query process of the KG is quite standard, the population of the KG is an aspect that can vary widely, and in many cases, procedural languages are used to map existing data into an ontology. Data mapped to an ontology comes in the majority of cases from relational databases or other structured formats, such as tables or comma separated values, among others. These formats can be easily mapped with procedural languages, result in mapping mechanisms that are non-scalable in order to be mapped.

The RDF Mapping Language (RML)[1] comes to tackle the problem of creating mapping mechanisms based on procedural languages, or that are restricted to a single dataset (e.g. the

KGCW'23: 4th International Workshop on Knowledge Graph Construction, May 28, 2023, Crete, GRE

*alexandros.vassiliadis@meaningfy.ws

eugen@meaningfy.ws (E. Costetchi); alexandros.vassiliadis@meaningfy.ws (A. Vassiliades); csongor.nyulas@meaningfy.ws (C. I. Nyulas)

https://costezki.ro/ (E. Costetchi)

0000-0002-9862-5070 (E. Costetchi); 0000-0001-6035-1038 (A. Vassiliades)

[1] https://rml.io/specs/rml/

RDB to RDF Mapping Language (R2RML)[2]), as RML offers a generic method, based on declarative rules, to map data into an ontology while supporting multiple input data formats [1].

In this paper, we present a methodology to map European Union (EU) Public Procurement Data (PPD) published on the public Tenders Electronic Daily (TED) website[3] into the eProcurement Ontology (ePO)[4] [5]. More specifically, we implement our methodology over a specific subset of the TED data called the Standard Forms for Public Procurement[6]. The method is based on mapping the various concepts that appear in each Standard Form into fragments of ePO, a procedure called Conceptual Mapping (CM) of the data. Then, based on this CM, we create our RML mapping rules that convert data from XML files representing the encoded content of the filled out Standard Forms, into instances of ePO, a procedure that we call Technical Mapping (TM). Finally, we present a validation mechanism that automatically produces SPARQL queries and SHACL Data Shapes to check the quality of the data produced by the mapping process. The RML mapping rules included in a mapping suite are used to translate the XML data into RDF format. Then, a test suite of SPARQL query assertions is automatically built from the CM file. The statements are built based on the references to ontology entities associated to each mapped XPath. These assertions are then used to determine whether a certain ontology fragment is instantiated or not in the output file for an XPath from the CM that was matched in the input.

The motivation behind this paper lies in the fact that mapping heterogeneous data into an ontology is a difficult task which requires a sophisticated analysis of the data to be mapped in order to develop the RML mapping rules. For this reason, we propose a new methodology of mapping heterogeneous data based on the innovative idea of mapping partitions. A group of mapping rules is referred to as a mapping partition when it produces a distinct subset of the knowledge graph. Therefore, a mapping partition is defined based on the output it produces. That is, a mapping partition is a set of mapping rules which produce distinct subgraphs of the knowledge graph. Our interest is in mapping TED data, which is of high importance and value for the EU citizen, into ePO ontology, which is an emerging semantic standard for PPD. Moreover, our motivation lies in the fact that the data that is produced should continuously be evaluated regarding its quality against the input data and the ontology in which is mapped into.

The key contribution of this paper consists in the novel methodology that we present for mapping PPD into fragments of ePO. Considering also the continuous update of PPD, e.g. from Standard Forms to eForms[7], and the version updates of ePO which introduce changes in classes and relations, mapping PPD into ePO becomes an even more challenging task. In our methodology, the CM offers: (a) the identification of the Business concepts in both the source and the target representations; (b) it serves as a source to generate validation tests; (c) it manages the complexity of mapping multiple versions of the source to a version of the target; and (d) organises the mapping rules in terms of mapping suites, as they are designed in Standard Forms. Next, the TM offers a generic mapping methodology for mapping heterogeneous PPD into the ePO ontology by using the RML mapping rules. In this mapping methodology we propose how

[2] http://www.w3.org/TR/r2rml/

[3] https://ted.europa.eu/TED/browse/browseByMap.do

[4] https://joinup.ec.europa.eu/collection/eprocurement/solution/eprocurement-ontology

[5] https://github.com/OP-TED/ePO

[6] https://simap.ted.europa.eu/web/simap/standard-forms-for-public-procurement

[7] https://single-market-economy.ec.europa.eu/single-market/public-procurement/digital-procurement/eforms_en

to manage complexity by having the mapping rules being managed as incomplete fragments, some reusable and some specific to a ”mapping suite” (i.e., Form number). Another contribution of the paper is the validation mechanism that checks for the quality of the produced data, by automatically creating SHACL Data Shapes and SPARQL queries. Finally, we provide a set of Command Line Interface (CLI) tools publicly available[8] to anyone that can be used to aggregate all that is needed for each mapping suite in a self-sufficient package.

The outline of this paper is the following. In Section 2, we present the related work to this paper. Next, in Section 3 we describe the nature of data, we give a high-level analysis of the ePO ontology, and we describe the RML mapping mechanism. We also present our validation mechanism which produces SHACL Data Shapes and SPARQL queries based on the CM of the PPD Standard Forms, and we show the mapping suite dissemination. In Section 4 we present the validation report of our framework. We conclude our paper with a discussion over the resulting methodology by displaying some conclusions and proposing future work directions.

2. Related Work

The related work will be separated into two main subsections, one for generic mapping methodologies that use RML[1] or other mapping languages that exploit declarative rules, and one for methodologies that are concentrated to procurement data. It is worth mentioning that currently RML is perhaps the most commonly used method for mapping knowledge into an ontology, and its popularity is steadily increasing. For instance, the authors in [2] demonstrate how the use of standard declarative mapping rules (i.e., R2RML) guarantees a systematic and sustainable workflow for constructing and maintaining a KG.

Looking at generic mapping methodologies that attempt to map heterogeneous data into an ontology, by using RML or other mapping languages based on declarative rules, we observe that most of these efforts are either treated only at a theoretical level, or are tested only over a handful of context restricted datasets. More specifically, the studies [3, 4] offer a generic method on how to map heterogeneous data into ontologies, but do not test their method over any specific dataset. Next, the studies [5, 6, 7] also offer a generic methodology for mapping data into an ontology, and test their method into specific datasets, but are different from our context of PPD of TED. [5] and [6] use the SDM-Genomic-Dataset [4] and the GTFS-Madrid-Bench [8], while [7] uses the NPD Benchmark [9]. Another interesting aspect, when comparing the aforementioned studies with our study, is that their evaluation method mostly focuses on time performance, while we are interested in the quality of mapped data. One could also read the thesis of David Chaves-Fraga [10], which gives a more complete view on how to map heterogeneous data into an ontology.

The area of mapping heterogeneous data from public procurement databases is quite rich, as well. We can find numerous studies, such as [11], which uses data from the Public Procurement Pilot Experience, and [12], which focuses on the European railway domain. Similarly to the first category of the related works, these studies fall into a different category of experiments with procurement data than our study. For example, the aforementioned studies do not provide

[8] https://github.com/meaningfy-ws/mapping-workbench

a conceptual mapping in a commonly used ontology, and they also lack a validation mechanism that evaluates the quality of the produced data.

An interesting approach for mapping PPD to an ontology is presented in [13]. The difference between this paper and ours, is that we offer a different method of mapping PPD with RML. Metaphor [14, 15] is a spreadsheet parser able to generate mapping rules in three mapping languages: R2RML, RML (with extension to functions from FnO) and YARRRML. In contrast to this paper, our mechanism uses the CM to automatically generate the SPARQL queries that evaluate the produced data, and also the authors of Metaphor do not work on PPD.

Another interesting paper is this of Dimou et al. [16], where the authors incorporate (i) a test-driven approach for assessing the mappings, instead of the RDF dataset itself, as mappings reflect how the dataset will be formed when generated; and (ii) perform semi-automatic mapping refinements based on the results of the quality assessment. The difference to our study is the type of data used, as we work on PPD, while the authors work with DBpedia and iLastic[9].

3. Methodology

In this section, we will start by presenting the mapping methodology overview, followed by a description of the nature of source data and the high-level structure of eProcurement Ontology, which represents the mapping target.

In Figure 1, one can see the architecture of the framework presented in this paper. In the *Conceptual Mapping* layer, the relevant PPD Standard Forms from the TED website are selected (see Subsection 3.1) to be mapped in the CM. A sample dataset is created for the purpose of testing and validating the mapping rules, and, a Conceptual Mapping is created aligning business concepts, XML paths and ontology fragments. Next, in the *Technical Mapping* layer, the CM is being implemented using RML language[1]; this is referred to as *Create Technical Mapping*. Furthermore, the *sample dataset is transformed* with the implemented TM rules to enable quality control. In the third layer (*Validation*), we depict SPARQL and SHACL validation steps which evaluate the quality of the produced data, and if violations and inconsistencies are found the mechanism will point which parts of the CM seem to have an issue. Once the validation is passed successfully, in the fourth layer *Dissemination*, the *Mapping Suite is available* (for a Notice Type) and they are stored in the *Mapping Suite Repository*[10] to be used by the transformation pipeline, when necessary.

For the validation procedure, as well as for transforming data from the XML files of the Standard Forms into RDF, we offer a set of CLI tools that one can use in order to access, transform, and validate the data. The command line interaction tools can be found here[11], where a throughout documentation on how to use them is provided.

3.1. Nature of Source Data

The data we are mapping into ePO refer to the PPD that can be found in the Standard Forms of the TED[6]. These forms exist to help citizens to publish EU PPD in the Official Journal of the

[9] http://explore.ilastic.be/

[10] https://docs.ted.europa.eu/rdf-mapping/repository-structure.html

[11] https://github.com/OP-TED/ted-rdf-conversion-pipeline

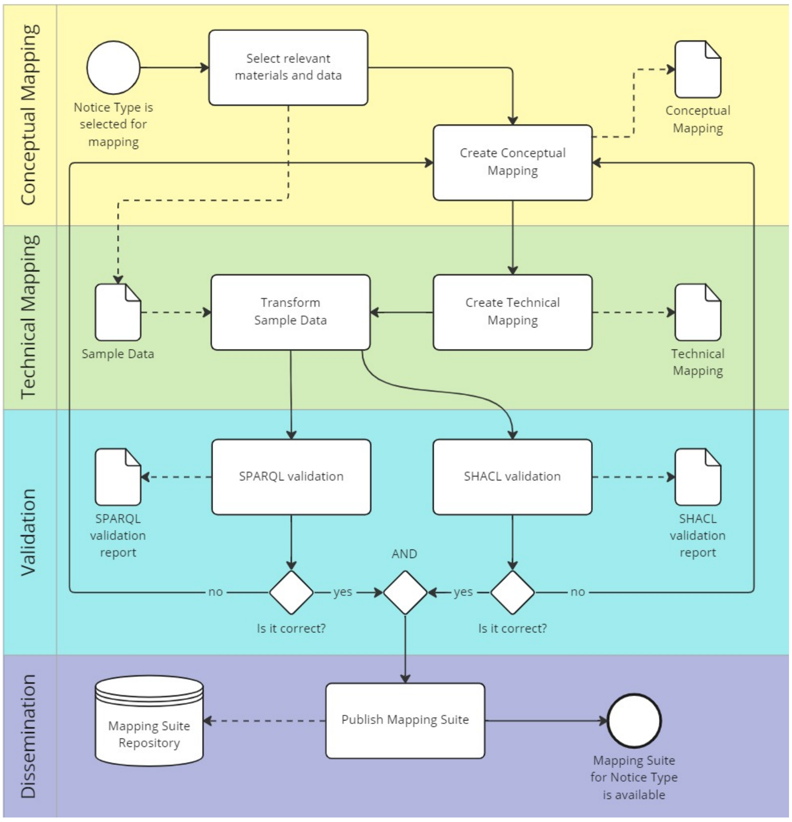

Figure 1: Mapping methodology workflow for the EU public procurement data

EU. The European Commission has created Standard Forms aligned with each of the EU legal bases in place for publishing this data, namely: (i) TED schema forms set out in Regulation (EU) 2015/1986 and (ii) eForms set out in Regulation (EU) 2019/1780. More specifically, currently we mapped forms F03, F06, F13, F20, F21, F22, F23 and F25[12], and we will be progressing with the remaining ones.

The TED Standard Forms that we are currently mapping to ePO are in PDF format, but they also have an XML counterpart[13], for each notice. We work with these XML notices, as it is a more appropriate format to map. By notice, we mean an instance of a completed form, where the types of the forms are the different TED Standard Forms (i.e., form F03, F06, F13, etc).

[12] see Standard Forms for Public Procurement (set out in Regulation (EU) 2015/1986) on SIMAP website: https://simap.ted.europa.eu/standard-forms-for-public-procurement

[13] see TED XML schemas (R2.0.9 & R2.0.8) for Standard Forms on EU Vocabularies website: https://op.europa.eu/en/web/eu-vocabularies/e-procurement/tedschemas

3.2. Target Ontology - the eProcurement Ontology

The eProcurement Ontology (ePO)[14][15] is a semantic data model that conceptualises and formally encodes the knowledge representation of the public procurement domain. Its primary purpose is to bridge the interoperability gap in the European public procurement data space, and can be used for data exchange, access and reuse. The ePO ontology was created because vocabularies and the semantics that they are introduced through PPD, the phases of public procurement that they are covering, and the technologies that they are using all differ. These differences hamper data interoperability, and thus its reuse by the wider public. This creates the need for a common data standard for publishing public procurement data, hence allowing data from different sources to be easily accessed and linked, and consequently reused. ePO facilitates encoding procurement data and making it available in an open, structured and machine-readable format.

The ultimate objective of the eProcurement Ontology project is to put forth a commonly agreed OWL ontology (and other necessary artefacts, such as SHACL data shapes and additional reasoning axioms) that will conceptualise, formally encode and make available in an open, structured and machine-readable format data about public procurement, covering end-to-end procurement, i.e., from notification, through tendering, to awarding, ordering, invoicing and payment.

ePO offers a UML representation[16] with which one could interact to get familiar with the ontology schema and the various object/data properties that it has. The ontology consists of about 140 classes, nearly 300 object properties, about 220 data properties, and uses more than 50 controlled vocabularies.

3.3. Conceptual Mapping

Let us begin with a small example on how the CM works, in order to have an intuitive understanding first. Consider the notice 113175-2023[17], which in Section II, subsection II.1.1 has a title.

Then, for each notice that is similar to notice 113175-2023 (i.e., is of the same form) it is expected for the mapping rule to map the information in the same section to the representation of title in the ontology.

The purpose of the CM is to map sections, subsections and fields of the PPD Standard Form into ePO ontology fragments, which are carefully chosen sequences of properties and classes that represent well the instantiation context. In Table 1, one can see a line (here converted into a column) for one of the concepts from the forms translated into fragments of the ontology. Also, notice that some information was omitted here due to space restrictions.

- The **Form Number** row indicates in which form(s) the concept being mapped is found.
- The **Standard Form Field ID** and **Standard Form Field Name** rows indicate the identifier and description of the **Section/Field** as found in the form.

[14] https://joinup.ec.europa.eu/collection/eprocurement/solution/eprocurement-ontology
[15] https://github.com/OP-TED/ePO
[16] https://docs.ted.europa.eu/EPO/latest/_attachments/html_reports/ePO/index.html?goto=1:1:7:142
[17] https://ted.europa.eu/udl?uri=TED:NOTICE:113175-2023:TEXT:EN:HTML&src=0

- The **Field XPath** row indicates the XPath of the concept in the XML counterpart of the forms. This usually is generic for each form, as the XPath is the same for each notice in a form.
- The **Class Path** row indicates how the concept is being mapped into ePO class, in this case, for the *Title* concept, an instance of the *epo:Procedure* class is being created, which is associated through a property with the datatype *xsd:string*.
- The **Property Path** row indicates how the concept is being mapped into ePO properties, in this case, for the *Title* concept, the instance of the class *epo:Procedure* is associated with a value of *xsd:string* datatype through the property *epo:hasTitle*.

Table 1
Information contained in a CM rule (concise form)

Form Number	3,6,13,20,21,22,23,25
Standard Form Field ID	II.1.1
Standard Form Field Name	Title
Field XPath	*OBJECT_CONTRACT/TITLE*
Class Path	*epo:Procedure/xsd:string*
Property Path	*?this epo:hasTitle ?value*

3.4. Technical Mapping

The RML mapping mechanism refers to the declarative rules that convert the data from the XML files of the Standard Forms into RDF triples, but they are converted only to the extent the toolchain permits, and only for the purpose of validation/testing. The development of the mapping rules was more natural due to the preliminary mapping that we have done on our data, as the CM helped us understand to which class and property we should map each element in the XML files.

We provide an example, to give an intuitive understanding of how the transition from the CM to the RDF occurred. Considering the information in Table 1, the idea behind this RML mapping rule is simple: an instance of the class *epo:Procedure* will be created with a unique name created based on the XPath representing a procurement procedure `EXPORT/FORM_SECTION/STANDARD_FORM_NUMBER/OBJECT_CONTRACT`. This instance will be associated with a title that exists in the XPath `ancestor::STANDARD_FORM_NUMBER/@LG`. `STANDARD_FORM_NUMBER`, which varies according to the Standard Form. Also notice that *epo:* is the namespace prefix of the ePO ontology.

```
                                        RML Mapping Rule
tedm:Procedure a rr:TriplesMap ;
  rr:subjectMap
      [
          rr:template ”ID” ;
          rr:class epo:Procedure
      ] ;
  rml:logicalSource
```

```
        [
            rml:source ”data/source.xml” ;
            rml:iterator ”/TED_EXPORT/FORM_SECTION/STANDARD_FORM_NUMBER/OBJECT_CONTRACT” ;
            rml:referenceFormulation ql:XPath
        ] ;
    rr:predicateObjectMap
        [
            rr:predicate epo:hasTitle ;
            rr:objectMap
                [
                    rml:reference ”TITLE”;
                    rml:languageMap [
                         rml:reference ”lower-case(ancestor::STANDARD_FORM_NUMBER/@LG)”
                    ]
                ] ;
        ] .
```

The URI creation, provided in subject map template, is based on a hashing function. This functionality is accessed through a REST call to a digest API. This guarantees unique reference to the element. This mechanism of generating a unique deterministic URI is useful in both cases: (a) when generating the URI of an instance (in *rml:subjectMap*), and (b) when referring to the URI of an instance (in *rml:objectMap*). Notice that the ID in the *rr:template* is a toy value.

Reflecting on the mapping rules, in most cases we managed to create generic rules that will apply over all Standard Forms. However, there were also exceptions to that, as some mapping rules were restricted to a specific Standard Form. This usually occurred because some Standard Forms contain sections or subsections that were found only in a specific form.

In order to handle the complexity of mapping the Standard Forms into ePO we had to consider some baselines for the RML mapping rules to be more customizable. We have applied the following solutions:

- *Sectioning within a form*, meaning that we have mappings for each form section in order to increase maintainability. When any changes apply to a section, rules for other sections will not be affected.
- *Segregation of rules* (generic and form specific), meaning that there are generic files and a file per mapping suite.
- *Apply relative paths* in the mapping rules for handling versioning in the XML files
- *Reuse of rules across Standard Forms and packages of Standard Forms*, meaning that there is a set of general source files where all the rules are kept as single source of truth. There is a selection and packaging process that picks the necessary modules to form a unified, self-sufficient package for each Standard Form (see Section 3.6).
- *Management of* rml:TripleMap *parts*, meaning that we had to separate the statements of *rml:subjectMap* and *rml:logicalSource* in form-specific modules, whereas the statements of *rml:predicateObject* are contained in modules reused across forms. Only after an assembly of parts (and packaging) process, the mapping rules are integrated and executable (see Section 3.6).

3.5. Mapping Validation

The validation mechanism starts with the transformation of the sample XML data by using the RML mapping rules provided in the mapping suite. Then, from the conceptual mapping file, a test suite of SPARQL query assertions is automatically generated. These generated assertions reference the ontology fragments to which each XPath was mapped to. The assertions are then used to check for a given mapping rule in the CM if the relevant ontology fragment was instantiated or not in the output file.

Moreover, the sample dataset is indexed for unique XPaths found in each sample XML file. This index is used for checking whether an input, specific to a given mapping rule in the CM, is present in the sample file or not.

The SPARQL-based validation of the transformed sample dataset includes, for each RDF file, first, the execution of all SPARQL query assertions, and second, asserting the presence of XPaths mentioned in the CM. The SHACL validation is streamlined to standard application of data shape files to each RDF output. The result is a set of reports that reflect the quality of the data that was produced by the RML mapping mechanism. In more detail, the validation mechanism will does two things: (i) it will create for each line of the CM (see Table 1) a SPARQL query that checks if the data corresponding to the XPath mentioned in that line has been translated to an appropriate RDF triple, and (ii) based on the SHACL Data Shapes provided in the context of ePO[18], checks if the ontology is correctly instantiated.

In addition to the SHACL Data Shapes and SPARQL queries, we offer another form of evaluating the quality of the data, but this time the evaluation is performed on the input data, i.e., in our case the XML files that represent the Standard Forms. This last form of evaluation refers to the XPaths of the concepts that exist in the CM and are about to be mapped in ePO. Basically, what the XPath "checker" does, is to see if there exist or not an XPath for the concept in the XML file, and if does, whether it is unique or not. The XPath checker serves a greater purpose than just checking the existence or plurality of XPaths in the data, as it allows us to interpret violations of the SPARQL evaluator, i.e. the *unverifiable* assertions (when they fail on the output, but no input for the rule is available either), the *warning* assertions (when they succeed in the output, but no input for the rule is available). This helps us understand if the issue lies in the output data, in the input data, or in the mapping rules (technical or conceptual). Section 4 provides a more detailed description of assertion severities.

3.6. Mapping Suite Dissemination

The mappings are part of a larger ecosystem, where they are used for systematically transforming the TED notices. In this context, the mapping rules are being prepared as self-sufficient mapping packages called *mapping suites.* There is a governance procedure for how they are maintained, consumed and disseminated. In this section we focus mainly on how they are structured.

The mapping suites are maintained and published in a GitHub repository[19]. The maintenance is supported by a custom-built toolchain[20]. The repository from the mapping suites are ingested

[18] see the eProcurement Ontology official GitHub repository https://github.com/OP-TED/ePO

[19] see the TED RDF mappings repository in GitHub https://github.com/OP-TED/ted-rdf-mapping

[20] see the mapping workbench toolchain repository in GitHub https://github.com/meaningfy-ws/mapping-workbench

by the transformation pipeline, and is organised as follows:

- `/docs` folder contains the documentation of the project. It is written in AsciiDoc format and compiled with Antora system [21].
- `/mappings` folder contains mapping suite packages organised based on the Standard Forms numbers.
- `/src/mappings` folder holds all the RML mappings files for all Standard Forms in a "single source of truth".
- `/test_data` folder contains sample TED notices selected with advanced search methods.
- `/sampling_XX` subfolder contains the forms produced in the time frame XX, for example /sampling_2014-2021 refers to sample notices produced in the years 2014 to 2021.

If we zoom into a mapping suite, for example */package_F03*, it will be composed of several elements assuring its completeness and self-sufficiency for ingestion, eligibility checking, transformation, validation and reporting processes, undertaken by the transformation pipeline. Such a package also covers the needs in the development and testing of a given "mapping suite".

- `metadata.json` automatically generated from Metadata sheet of `conceptual_mapping.xlsx` describing the parameters for selecting the notices that the mappings can be applied to, and various version information.
- `/transformation/conceptual_mappings.xlsx` is a CM specific to a form number.
- `/transformation/resources` contains additional resources necessary to apply the transformation rules, e.g. JSON and CSV files to map values to controlled vocabulary terms.
- `/transformation/mappings/*.rml.ttl` the relevant RML transformation rules, organized in module files (copied from the "single source of truth" mappings folder) according to the specification in the "RML Modules" sheet of the `conceptual_mappings.xlsx`.
- `/test_data` automatically selected test data (possibly grouped in suborders) that contain a minimal number of sample files, but which are the most representative and complete specimens in the entire data population.
- `/output` is a placeholder folder created at runtime to store outputs of the sample data transformation.
- `/validation/shacl` contains all the SHACL test suites, used in the validation and development process.
- `/validation/sparql` contains all the SPARQL test suites, used in the validation and development process.
- `/validation/sparql/cm_assertions` contains the SPARQL assertion queries automatically generated from the conceptual mapping.

4. Empirical Evaluation

In this section, we analyse briefly the results that the SHACL and SPARQL validators return and how we can interpret the results in order to optimise our mapping rules. Starting with

[21] read more about AsciiDoc and Antora on https://antora.org/

Table 2
Sample SHACL validator results

SHACL Violation	Property of Application	Form	Notice
Value does not have class epo:LotSpecificTerm	epo:isSubjectToLotSpecificTerm	3,6,13 20,21,22 23,25	002705-2021 ... 654902-2021
Value does not have class epo:ProcedureSpecificTerm	epo:isSubjectToProcedureSpecificTerm	3,6,21 22,23,25	013921-2021 ... 359962-2021
Value does not have class epo:Duration	epo:hasQualificationSystemDuration	22	012969-2017 309175-2020
More than 1 values	epo:hasMainActivityDescription ... epo:indicatesAwardOfLotToWinner	3,6,13 20,21,22 23,25	135100-2021 ... 344048-2021
Less than 1 values	epo:hasProcurementScopeDividedIntoLot ... epo:unitType	3,6,13 20,21,22 23,25	000163-2021 ... 654902-2021

the SHACL Data Shape validator, we can see that currently there are three types of violations that refer to either (i) missing class relations (i.e., an instance is not classified correctly), (ii) cardinality constraint for more than one value, and (iii) cardinality constraint for less than one value (see Table 2). Notice that we display only a sample of violations due to space restrictions.

The second column indicates the property to which the violation applies; the third, the form(s) in which it can be found, i.e., Standard Form(s) in which it occurs; and the fourth, the specific notices. For the third column we can comment that due to clustering for display purposes the cardinality violations appear in all packages. The interpretation of errors is much easier with an analysis like this, as for example for the cardinality issues we can check if the constraint in ePO is perhaps too strict and needs to be relaxed, or the mapping rule needs to be modified.

Moving to the SPARQL evaluation, we follow a similar analysis where we summarise the types of SPARQL inconsistencies, i.e., unverifiable queries, invalid queries, warnings and errors.

- *error:* Refer to SPARQL queries which failed with an error, most likely because of incorrect SPARQL syntax or other technical issue.
- *invalid:* Refer to SPARQL queries which concern data that can be found in the input, but not in the output.
- *unverifiable:* Refer to SPARQL queries which concern data that cannot be found either in the input or the output data.
- *warning:* Refer to SPARQL queries which concern data that cannot be found in the input, but can be found in the output.

The validation reports contain five result statuses: *Valid*, *Unverifiable*, *Warning*, *Invalid* and *Error*. Most of the results are *Valid* or *Unverifiable*, in case there is no input data in the sample to trigger a mapping rule. Some *Warning*s are signalled in cases when the field is found in the output, but not detected in the input. *Invalid* results are generated in cases when the data was found in the input, but is missing (or not detected by the current reporting tool) in the output. *Error*s occur when the query is wrong, or cannot be executed. No *Error*s are acceptable, and the few found in current reports are not real errors. A few *Invalid* results are found in the validation reports. Based on our analysis, they are not reflecting incorrect mapping rules or final data.

There are 1466 SPARQL queries automatically generated from the CM, which were distributed over 8 different Standard Form types, and executed over 850 notices. More specifically, for each notice of the forms F03, F06, F13, F20, F21, F22, F23, and F25, a set of 200, 195, 122, 146, 231, 231, 194, and 147 SPARQL queries were executed, respectively. Table 3 shows the number and percentage of queries for each type of inconsistency, over the total number of 217,179 query executions. The 82,477 query executions (or 37.98%), not shown in the table, were *Valid*.

Table 3
SPARQL Validator Result

Type of Inconsistency	Number of occurrence	Coverage
Error	151	0.07%
Invalid	3,988	1.84%
Unverifiable	104,860	48.28%
Warning	25,703	11.83%
Total	**217,179**	**62.02%**

The *Warning* and the *Unverifiable* ones are not so relevant. The first might be the result of situations when multiple XPaths generate similar (i.e. partially matching) RDF fragments, and if one XPATH is present in the data, while the other one is not. For the *Unverifiable* ones, in most cases the issue is a missing XPath in the input data. Nevertheless, we report the *Warning* and *Unverifiable* violations to have a complete view of the coverage of each violation. On the other hand, those that should be analysed and be taken more seriously are the *Invalid* ones because in this case, (a) either the SPARQL query was not correctly generated by the SPARQL validator, (b) the ontology fragment in the CM is not correctly specified, or (c) there is an issue in the selected sample data. Looking at the *Invalid* violations one could get a guide on which data was not mapped into the ontology, although it exists in the input data, and therefore take the necessary actions in order to catch the violation. In our case, the *Invalid* violations helped us narrowing down the data that was not mapped, reaching a point that almost all data (more than 99%) is mapped. The *Error*s might be caused when from the CM information we generate invalid SPARQL queries, which, in our case, were due to a bug in the SPARQL generation CLI.

5. Discussion

Our intuition, when mapping heterogeneous data into an ontology is that the existence of a preliminary mapping methodology before developing RML rules, is mandatory. In most cases, a CM seems to be a direction that eases significantly the task of developing RML rules, and also gives assurance of the quality of the produced data. The intuition behind the CM is to map concepts of the data into fragments of the ontology. The reason for which a preliminary mapping, such as a CM, is important, is because it allows us, on the one hand, to better understand how the mapping rules should be developed according to the understanding from business requirements point of view , and on the other hand, to check if the data was indeed mapped to the correct property and class after the mapping process took place.

Moving to the mapping rules, we consider that when we want to construct a generic mapping mechanism based on RML, we should keep in mind the following key points: (i) sectioning, meaning that the data, if possible, should be split into sections, as this will increase the maintainability of mapping rules, for example changes that are applied to the rules of one section should not affect the rules in other sections, (ii) segregation of generic and form specific rules, meaning that in a mapping suite there are multiple generic mapping files combined with one form-specific mapping file, (iii) use of relative paths in the mapping rules for easier handling of versioning in the XML files, as some concepts from the forms may be found in different places of the XML file over time, and (iv) reuse of rules across the data, by having a general source file where one would keep all the rules as single source of truth and then package, whatever is needed, for each data instance.

Another important aspect when creating a generic mechanism for mapping heterogeneous data into an ontology is the evaluation of the produced data. This procedure, ideally, should be twofold. One should check, on the one hand, the quality of the data in the produced `*.ttl` files, and on the other hand, how the produced data fulfils the constraints posed by the ontology that we mapped to. For the first part, a combination of the XPath and SPARQL-based evaluator, i.e., an evaluator that checks using SPARQL queries if each XPath from the input data has been mapped to a fragment of the ontology, seems ideal. Based on a mechanism like this, we can interpret the types of violations that each SPARQL query returns in order to correct our mapping rules, or maybe clean some noise from the input data. For the violations of the SPARQL queries we can comment that the *Invalid*s are the most important ones, as the *Invalid* violations indicate that for something in the input data there was no corresponding output data found.

For the second part of the evaluation, i.e., the one checking if the produced data respect the constraints posed by the ontology, a SHACL Data Shape validator that automatically extracts all the conditions from the ontology, and checks the SHACL Data Shapes against the produced data, seems o be a suitable option. Such a SHACL validator is very helpful, as it indicates the types of errors in detail, and one could immediately change the mapping rule or the CM if necessary. Based on our experience, many SHACL Data Shape violations are generated for instances not having as their type the class that they were supposed to. Besides being an obvious error in the mappings, this can also happen because the mapping mechanisms do not generate statements to describe the schema of the ontology, e.g. there are no subclass relations present in the produced files. Meaning that the instances which are shown to have missing classes might in fact be instances of the class that is indicated, but as an instance of some subclass of the indicated one. Taking into consideration also the ontology itself and enabling (basic) reasoning during the validation process would eliminate such false violation reports. Another big group of SHACL Data Shape violations are due to cardinality constraints. Besides erroneous mappings, these kind of violations can happen also due to invalid input data, but most often they are due to over- or under-constrained properties in the ontology.

Concerning some potential limitations that are presented in our methodology, we can point out the following. Firstly, the CM is not automatically aligned to the versioning of the ePO ontology, that means that each time there is an update to ePO, if properties or classes are changed/renamed/deleted, then we need to reflect this in the CM by hand. Similarly, in the TM, the mapping rules do not support versioning of ePO. Moreover, another limitation is the mandatory use of absolute paths in our TM. This is due to the fact that many paths are not unique, which results in using absolute paths in numerous instances in the iterators or join conditions of the mapping rules. Unfortunately, this also reduces the scalability of our TM as it may not be able to map all the existing Standard Forms. Finally, concerning the SHACL and SPARQL validators, we could say that a beautification to the summariser would be welcome.

6. Conclusion

In this paper, we presented a mapping methodology that maps Public Procurement Data from the EU TED website into eProcurement Ontology. More specifically, we implemented our methodology over a specific subset of the TED data called the Standard Forms for Public Procurement. The method is based on mapping the various concepts of each Standard Form into fragments of ePO, a procedure called the Conceptual Mapping (CM) of the data, then based on this CM we developed our RML mapping rules, which convert the data from the XML files that the Standard Forms are represented in, into instances of ePO classes, a procedure we call the Technical Mapping (TM). Finally, we presented a validation mechanism that automatically produces SPARQL queries and SHACL Data Shapes to check the quality of the produced data.

We believe that one could benefit significantly from using the methodology presented in this paper when mapping heterogeneous data. Firstly, the existence of a CM allows for a better "control" over where the data will be mapped, it enables quality control for the produced data, and it also makes it easier to develop mapping rules. Next, the bullet points presented for the TM in subsection 3.4 show how we can better partition the data that we have to map to modularize the mapping rules. Finally, the SPARQL and SHACL evaluators ensure to a great extent the quality of the produced data, by indicating where we need to fix or adjust a mapping rule, or change the mapping we have in the CM.

As for future work, we plan to start mapping eForms[22] into ePO, as eForms will gradually replace Standard Forms for storing PPD in EU TED. We are also interested in supporting versioning of ePO, meaning that if changes apply to ePO, we should be able to easily update our CM and TM to maintain the high quality of the generated data. Finally, we plan to further improve the quality of the mapped data, by analysing the various SHACL and SPARQL violations.

References

[1] D. Van Assche, T. Delva, G. Haesendonck, P. Heyvaert, B. De Meester, A. Dimou, Declarative rdf graph generation from heterogeneous (semi-) structured data: A systematic literature review, Journal of Web Semantics (2022) 100753.

[2] D. Chaves-Fraga, O. Corcho, F. Yedro, R. Moreno, J. Olías, A. De La Azuela, Systematic construction of knowledge graphs for research-performing organizations, Information 13 (2022) 562.

[3] S. Jozashoori, D. Chaves-Fraga, E. Iglesias, M.-E. Vidal, O. Corcho, Funmap: Efficient execution of functional mappings for knowledge graph creation, in: The Semantic Web–ISWC 2020: 19th International Semantic Web Conference, Athens, Greece, November 2–6, 2020, Proceedings, Part I 19, Springer, 2020, pp. 276–293.

[4] E. Iglesias, S. Jozashoori, D. Chaves-Fraga, D. Collarana, M.-E. Vidal, Sdm-rdfizer: An rml interpreter for the efficient creation of rdf knowledge graphs, in: Proceedings of the

[22] https://single-market-economy.ec.europa.eu/single-market/public-procurement/digital-procurement/eforms_en

29th ACM International Conference on Information & Knowledge Management, 2020, pp. 3039–3046.

[5] J. Arenas-Guerrero, D. Chaves-Fraga, J. Toledo, M. S. Pérez, O. Corcho, Morph-kgc: Scalable knowledge graph materialization with mapping partitions, Semantic Web (2022) 1–20.

[6] J. Arenas-Guerrero, M. Scrocca, A. Iglesias Molina, J. Toledo, L. Pozo-Gilo, D. Dona, O. Corcho, D. Chaves-Fraga, Knowledge graph construction with r2rml and rml: an etl system-based overview, in: CEUR workshop proceedings., volume 2873, CEUR Workshop Proceedings, 2021, p. 1.

[7] D. Calvanese, B. Cogrel, S. Komla-Ebri, R. Kontchakov, D. Lanti, M. Rezk, M. Rodriguez-Muro, G. Xiao, Ontop: Answering sparql queries over relational databases, Semantic Web 8 (2017) 471–487.

[8] D. Chaves-Fraga, F. Priyatna, A. Cimmino, J. Toledo, E. Ruckhaus, O. Corcho, Gtfs-madrid-bench: A benchmark for virtual knowledge graph access in the transport domain, Journal of Web Semantics 65 (2020) 100596.

[9] D. Lanti, M. I. Rezk, G. Xiao, D. Calvanese, The npd benchmark: Reality check for obda systems, in: Advances in database technology-EDBT 2015: 18th International Conference on Extending Database Technology, Brussels, Belgium, March 23-27, 2015, proceedings, University of Konstanz, University Library, 2015, pp. 617–628.

[10] D. Chaves Fraga, Knowledge Graph Construction from Heterogeneous Data Sources exploiting Declarative Mapping Rules, Ph.D. thesis, ETSI_Informatica, 2021.

[11] C. Guasch, G. Lodi, S. V. Dooren, Semantic knowledge graphs for distributed data spaces: The public procurement pilot experience, in: The Semantic Web–ISWC 2022: 21st International Semantic Web Conference, Virtual Event, October 23–27, 2022, Proceedings, Springer, 2022, pp. 753–769.

[12] J. A. Rojas, M. Aguado, P. Vasilopoulou, I. Velitchkov, D. Van Assche, P. Colpaert, R. Verborgh, Leveraging semantic technologies for digital interoperability in the european railway domain, in: The Semantic Web–ISWC 2021: 20th International Semantic Web Conference, ISWC 2021, Virtual Event, October 24–28, 2021, Proceedings 20, Springer, 2021, pp. 648–664.

[13] M. Nečaskỳ, J. Klímek, J. Mynarz, T. Knap, V. Svátek, J. Stárka, Linked data support for filing public contracts, Computers in Industry 65 (2014) 862–877.

[14] A. Iglesias-Molina, L. Pozo-Gilo, D. Dona, E. Ruckhaus, D. Chaves-Fraga, O. Corcho, Mapeathor: Simplifying the specification of declarative rules for knowledge graph construction., in: ISWC (Demos/Industry), 2020, pp. 25–30.

[15] A. Iglesias-Molina, D. Chaves-Fraga, F. Priyatna, O. Corcho, Towards the definition of a language-independent mapping template for knowledge graph creation, in: Proceedings of the Third International Workshop on Capturing Scientific Knowledge, 2019, pp. 33–36.

[16] A. Dimou, D. Kontokostas, M. Freudenberg, R. Verborgh, J. Lehmann, E. Mannens, S. Hellmann, R. Van de Walle, Assessing and refining mappingsto rdf to improve dataset quality, in: The Semantic Web-ISWC 2015: 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part II 14, Springer, 2015, pp. 133–149.