Semantic Object Mapping using SHACL Validated Resource Graphs

Sarah Bensberg [a,*] and Marius Politze [a]
[a] *IT Center, RWTH Aachen University, Germany*
E-mails: sarah.bensberg@rwth-aachen.de, politze@itc.rwth-aachen.de

Abstract. Metadata describing research data is a core instrument for compliance with the *FAIR Guiding Principles*. RDF is very suitable for this purpose because its schema-lessness provides flexibility for changes and different disciplines. In order to follow defined metadata schemas, RDF needs to be restricted by application profiles. SHACL allows validating whether an RDF based metadata matches a certain shape and meets minimum quality requirements. Described research data hence becomes resources in a validated knowledge graph. Searchability is achieved by SPARQL, which, however, requires considerable technical knowledge. The presented semantic object mapping of validated resource graphs into an JSON object is an approach which is less dependent on the users' background.

The evaluation was performed based on precision, recall and response times using Elasticsearch as a search engine on the mapped object in comparison to generated SPARQL queries. The results show that with the transformation of RDF based (meta-)data into a search index using application profiles and inference rules, a solution was found that is equivalent in these terms. Through the integration of the developed mapping in data management platform *Coscine*, a search of the research data is possible which at the same time promotes the subsequent use.

Keywords: Resource Object Mapping, Knowlegde Graph Validation, Shape Graphs, Research Data Management, Metadata

1. Introduction

As research data is often fundamental for scientific discoveries, it should be stored on a long-term basis and it should be reusable for validation or future research projects. Metadata, a data record describing the data, serves to make the data interpretable [1]. In the field of Research Data Management (RDM), metadata can be used to clarify questions that occur regarding research data, e.g., the date of creation. As additional information is stored by the metadata, it is possible to search research data or to filter it according to certain properties. A standardized presentation of the data makes the information machine-readable and thus a search possible. Through the usage of metadata, information can be linked across the entire World Wide Web [2], which is why it is particularly useful for accessibility and subsequent use.

Kindling and Schirmbacher [3] describe RDM as the entire process that supports the allocation, generation, processing, enrichment, archiving, and publication of digital research data. Within this context RDM services allow researchers to manage their research data according to the *Findable, Accessible, Interoperable and Reusable (FAIR) Guiding Principles* [4].

Description with metadata is crucial to secure long term interpretability within the research data management process. When regarding research data objects as resources, Resource Description Framework (RDF) triples are a common choice for storing such metadata. Due to the schema-lessness and semantics RDF provides flexibility for evolutionary changes and across different disciplines. In order to follow defined metadata schemas, however, RDF graphs need to be restricted by application profiles. Shapes Constraint Language (SHACL) allows validating whether an RDF (meta-)data graph matches a certain shape and thus meets minimum quality requirements. Furthermore, semantic resource object mappings could be generated

*Corresponding author. E-mail: sarah.bensberg@rwth-aachen.de.

because of the known structure of the knowledge graph. These models can be used in search engines to make the resources searchable and therefore to find research data.

1.1. Motivation

The Council for Scientific Information Infrastructures established by the Joint Science Conference has recommended the establishment of a National Research Data Infrastructure (NFDI) for the coordinated further development of the information infrastructure construction [5] that is now funded by the German Federal Ministry of Education and Research. The background to this is the establishment of a coordinated and sustainable data infrastructure for the German science system, which strengthens the competitive position of research in Germany through the systematization of data stocks, good accessibility of research data, and continuous further development of services [6]. "The aim of the National Research Data Infrastructure (NFDI) is to systematically manage scientific and research data, provide long-term data storage, backup and accessibility, and integrate the data both nationally and internationally. The NFDI will bring multiple stakeholders together in a coordinated network of consortia tasked with providing science-driven data services to research communities." [7]

While RDM best practices need to find their way into daily workflows of each individual researcher from Ph.D. student to principal investigator, the goal set by NFDI also requires the central university institutions, such as libraries or computing centers to be involved and provide supporting (IT) services [8]. For this reason RWTH Aachen University (RWTH), as a renowned research institution, decided to invest in institutional RDM structures [9]. RWTH supports researchers to do RDM and to deal with the topic comprehensively to avoid problems and difficulties during research work. Another motivational aspect is the security of research data. RDM should ensure that data is not lost and that it is protected against misuse and theft. The visibility of research data also plays a major role – research data should be documented in a comprehensible way and assigned to the researcher so that queries regarding the data can be made. Another point is the reusability of research data. The goal is to be able to reuse research data for new research projects and to gain more efficient access to already existing research data. Besides, research data is often already stored in different systems and the combination and unification of this data records pose new challenges for researchers [10]. More and more public funding programs by the German Research Foundation (GRF) or EU's Horizon 2020 expect research data to be made available and data management plans to be developed. While the access to research data and related services is usually limited to one project or organization [11]. However, the intention here is to enable cross-disciplinary access.

An important aspect that is relevant for the implementation of appropriate solutions is the diversity and heterogeneity of the various scientific disciplines as well as the decentralized nature of data. This results from the different research data infrastructures used by researchers and consisting different services. However, it should be possible to maintain the discipline-specific equipment and IT systems individually for each institute despite uniform support concerning RDM. The difficulty of reproducibility is thus increased even further. Additionally, initial approaches already exist at some research groups, which is why the different stages of development represent a further challenge [8].

It should be noted that for the researchers themselves, the added value of RDM is often indirect, so active integration into research processes is crucial [12]. Besides, the role of a dedicated "Data Curator", who is familiar with certain techniques/standards and technologies, is progressively disappearing within the context of RDM; instead, every single researcher is under obligation. This trend indicates that many "IT skills" are becoming more and more "general education" in the context of digitization. To support the researcher in all phases of the Research Data Life Cycle (RDLC), the goal of RWTH is to create a basic infrastructure of connected services [8]. To enable research groups to continue using individual components, technology-independent and process-oriented interfaces will be created. These ensure that different systems are bundled to generate added value for the researcher.

Several components are used for this purpose: (i) the EPIC Persistent Identifier (PID) service based on the Handle system for the permanent and unique referencing of research data [13], (ii) external metadata stored with the PID, which enables the use of discipline-specific systems, (iii) a central storage service in which researchers can store their research data, and (iv) the support of the University Library to publish the results in a suitable repository. As the importance of descriptive and explanatory metadata is growing, an application has been developed that allows the creation of metadata for discipline-specific application profiles [14].

In order to connect these existing infrastructures, the IT Center of RWTH is currently developing the research data management integration platform Collaborative Scientific Integration Environment (Coscine) [10], a software platform for allocation and management of IT resources in research projects. The main goal is to provide access to research data regardless of its storage location and to manage existing data *FAIR*. Coscine provides assistance in order to meet the requirements of the Good Research Practice (GRP) [15] such as the guidelines on the use of standards and methods, documentation, allowing to make research results publicly accessible and archiving. The platform is currently in a pilot phase for researchers at RWTH. While Coscine itself does not store research data, it supports the structured storage of metadata and creation of persistent identifiers on research data as described above. Metadata are stored as resource graphs using in the RDF model and are validated with SHACL shapes. At the moment, the implemented search functionality is limited, which hinders the reusability of data. To provide a more sophisticated search interface for researchers, Coscine intends to use an Elasticsearch (ES) [16] search index.

ES should be used as a search engine because it is scalable and with its document-based approach it fits very well with the mapping of metadata records. It also provides a near-real-time search for different types of data. This meets the requirements of a structured search as well as advanced search types, e.g., range queries. The crucial question is how to transform the Coscine knowledge graph into a search index. This is especially important because it is to be benefited the from structure and inference rules, i.e., from the advantages of the RDF model.

1.2. Research Methodology

A part of the search within Coscine is to find research data based on stored metadata. It enables retrieval and thus the reproduction or further development of research projects. The background of the presented paper is the elaboration and evaluation of a resource object mapping to build such a search considering the data structure and requirements of Coscine (see section 2.6).

Initially, the data is available as an RDF-based knowledge graph, which presents some general challenges described by Arnaout and Elbassuoni [17]:

- *Data incompleteness*, since the RDF triples contain a lot of information, but most knowledge is represented as free text.
- *Inflexible querying*, because triple-pattern queries are highly expressive, but at the same time restrictive, because they have to follow a certain structure and queries can only be made according to the underlying data structure and terms. A further restriction is that no query can be made regarding missing resources.
- *Missing result ranking*, since RDF graphs, or queries in such a graph may yield many results that can overwhelm a user. Therefore, a specific ranking is necessary, which can be based on different criteria. However, such a ranking is not provided by SPARQL Protocol And RDF Query Language (SPARQL), thus it has to be implemented explicitly.
- *Result diversity*, because it is important that the topmost results give a broad overview of the results of a query and do not all contain similar aspects.

Since the data is stored in a triple store, the most natural way to access it is the SPARQL Endpoint that allows querying and filtering. However, a user must have some knowledge to do this: (i) RDF and SPARQL, (ii) the structure of the stored data, (iii) used vocabularies and terms as well as their corresponding Uniform Resource Identifiers (URIs)[1]. Even if a user would have all the information, the use of SPARQL is very expressive and therefore challenging and thus can only be used by experienced Data Curators in a satisfactory manner. However, the goal is that every researcher should be able to use RDM tools.

The general user who ends up searching the data records is a researcher who does not necessarily come from a technical environment. Besides, usually no RDF or SPARQL knowledge is present and users are not aware of the structure of the data or the database model.

Based on the state of research on search in RDF graphs, and Natural Language Processing (NLP) oriented approaches from general Information Retrieval (IR), the work is oriented towards the following research hypothesis, which is to be tested:

A mapping of RDF-based metadata records into a search index exists such that a search engine on that

[1] or more specifically an Internationalized Resource Identifier (IRI). Since IRI is a generalization of an URI and both are used very interchangeably in general linguistic usage, only the term URI is used in the following, even if a IRI would be allowed.

index yields the same precision and recall when compared to generated SPARQL queries.

Therefore, this paper presents a semantic resource object mapping for a metadata record which is relies on SHACL based application profiles and SHACL validated resources graphs. Therefore, the following research questions are addressed:

- How to exploit features of SHACL for resource object mapping?
- How does such a mapping influence the search results in terms of precision and recall?

To test the research hypothesis and answer the research question, a method was first developed to convert the metadata graphs into JavaScript Object Notation (JSON) objects. This is necessary to be able to use a search engine like ES. The mapping applies inference rules to convert URIs into literals and to inject further human knowledge into the object.

To validate the mapping and its influence on search results, sample metadata records were created based on a real world application profile and different types of search intentions from the RDM domain were considered. Using these, the results provided by ES queries and SPARQL queries were evaluated in terms of precision and recall. Response times were measured on a small and a large data record to assess scalability. Also, necessary procedures for object mapping and communicating with ES were considered as additional effort.

2. Background

This section provides some background about the used metadata model and the relevant Semantic Web technologies used by Coscine, which form the foundation for the proposed resource object mapping.

2.1. Metadata

Various definitions of metadata exist in the literature. Steffen Staab [18] describes metadata as data that describes selected aspects of other data. Usually, metadata is structured data. Terms are linked to a resource via generic categories (e.g., “title”) and assigned to it. A resource describes a specific object using a link.

Therefore, metadata is information that says something about the creation, content, or context of a resource. They can be used to link data over the World Wide Web [2]. In general, metadata is used to understand data records in a broader context, so that people or users outside of their own project group or institution can interpret the data as well [19].

2.2. Resource Description Framework

RDF is one of the World Wide Web Consortium (W3C) conceived data model for the representation of information in the Web [20]. It is a data model that is used in the area of the Semantic Web. Data is structured in the form of triples (subject-predicate-object) and combined into a graph. For this reason, RDF offers a reasonable way to map metadata as it is possible to present metadata in the same form. A resource (subject) is assigned a term or a value (object) via a category (predicate).

Example 1 shows how to use the Dublin Core [21] vocabulary to map the metadata `X` has creator `Y` as RDF triple.

Example 1: Metadata as RDF triple

```
<X> dcterms:creator <Y>
```

2.3. Metadata Schemas and Application Profiles

The Joint Information Systems Committee in the UK (JISC) (Organization for the Promotion of Digital Technologies in Research and Education) [22] describes a *metadata schema* as a collection of metadata fields that are combined into a set. A metadata field is a resource that is used as a property of a particular subject. For example, the metadata fields “title”, “author”, and “subject” can be combined in a metadata schema. There are many official *metadata standards*, which have gone through an agreement and validation process of the metadata schema at some standardization body (such as the Dublin Core Metadata Initiative (DCMI) or W3C). Examples of such metadata schemas are Dublin Core [23], Data Catalog Vocabulary (DCAT) [24], DataCite [25], or RADAR [26].

Controlled Vocabulary can be a kind of linear keyword list, a hierarchically structured catalog or a taxonomy, thus limiting an input. According to the JISC [27], controlled vocabularies are used in metadata fields with the main goal of more efficient retrieval of resources through searching. They improve data consistency and reduce ambiguity in the language.

An *application profile* [28] is a specification for describing metadata in an application that uses controlled vocabularies and imposes further usage restrictions. The use of different metadata schemas is possible.

2.4. Open and Closed World Assumption

Along with the *Closed World Assumption (CWA)* a system with complete access to all information about a subject exists. If a search is made for a specific piece of information within this system, the correct answer is found and only this answer exists. In this way, unambiguous statements can be made.

In the context of the Semantic Web, the term *Open World* is used. In case of the *Open World Assumption (OWA)*, the information in a system is or can be incomplete. Just because an answer to a question cannot be found in one system, does not imply that it cannot be found in another system, which might contain the necessary information. Thus, it cannot be concluded it is the only result or that there is none. Therefore, the correct and complete answer is unknown.

The main difference between the two assumptions is that the CWA includes the *Unique World Assumption (UNA)*. The UNA [29] states that two things with different names are in fact different unless explicitely defined as the same thing. Example 2 illustrates this.

Example 2: Difference between CWA and OWA

Assuming the scenario [30] that a person can only live in one country and the following assertions are introduced: `Sarah lives-in Berlin` and `Berlin is-in Germany`. by adding `Berlin is-in France`, a CWA system finds contradiction since persons can only live in one country and according to UNA Germany and France are not the same countries. The decisive point is that by the name Berlin is the same city and not two different cities (which would have different URIs) with the same name:

$(\langle Sarah\rangle\ \langle lives\text{-}in\rangle\ \langle Berlin\rangle) \wedge (\langle Berlin\rangle\ \langle is\text{-}in\rangle\ \langle Germany\rangle) \wedge (\langle Berlin\rangle\ \langle is\text{-}in\rangle\ \langle France\rangle) \wedge (Germany \neq France) \rightarrow \bot$

In contrast, in an OWA system there would be no error, but the result would be the statement that Germany and France are the same things:

$(\langle Sarah\rangle\ \langle lives\text{-}in\rangle\ \langle Berlin\rangle) \wedge (\langle Berlin\rangle\ \langle is\text{-}in\rangle\ \langle Germany\rangle) \wedge (\langle Berlin\rangle\ \langle is\text{-}in\rangle\ \langle France\rangle) \rightarrow (\langle Germany\rangle = \langle France\rangle)$

However, an inconsistency could be created here, namely if the statement `Germany is-not France` would be added. The conclusion in OWA would then look like this: A person can only live in one country. Berlin is a city in Germany. Berlin is a city in France. It follows that Germany and France are the same things. But since Germany and France are two different things and therefore cannot be the same, something must be wrong:

$(\langle Sarah\rangle\ \langle lives\text{-}in\rangle\ \langle Berlin\rangle) \wedge (\langle Berlin\rangle\ \langle is\text{-}in\rangle\ \langle Germany\rangle) \wedge (\langle Berlin\rangle\ \langle is\text{-}in\rangle\ \langle France\rangle) \wedge (\langle Germany\rangle \neq \langle France\rangle) \rightarrow \bot$

The Semantic Web is a system with incomplete information. Missing information in this context means that the information has not yet been explicitly generated and that the Semantic Web follows the OWA, precisely because its main goal is to obtain new information [30]. Due to this fact, in Semantic Web applications rules restricting the range of a property cannot be applied so intuitively especially when compared to relational databases. This is mainly since databases are normally based on CWA.

Metadata is intended to provide additional information. Based on inference and the OWA, these statements are not unambiguous (see Example 2). If requirements need to be expressed in an application with metadata, these must be restricted by application profiles following a CWA. Only this allows validating whether they follow a certain schema and use only certain controlled vocabularies.

2.5. Shapes Constraint Language

SHACL [31] is another W3C recommendation and serves the validation of so-called *data graphs* against *shapes graphs*. A shape graph is an RDF graph, which describes conditions and restrictions that the data graph needs to fulfill. For this reason, it is more suitable than Resource Description Framework Schema (RDFS) to model the constraints in terms of a metadata schema. In addition to validation, the use of SHACL serves, among other things, to build user interfaces, generate code or to ensure data integration [32]. SHACL can be divided into the two languages *SHACL Core language* and *SHACL-SPARQL*.

SHACL shapes thus provide the means to describe an application profile with technologies of the Semantic Web.

Example 3 shows a SHACL shape.

Example 3: SHACL shape (RDF application profile)

In this example, `ex:ExampleShape` is defined as SHACL shape via the definition `a sh:NodeShape`. Using `sh:property` a property is defined which in the example corresponds to `ex:author`. The values of `ex:author` must be instances of the class `ex:author`.

```
ex:ExampleShape
 a sh:NodeShape ;
 sh:property [
  sh:path ex:author ;
  sh:class ex:Author ;
 ] .
```

2.6. Technical Details Collaborative Scientific Integration Environment

Coscine is a research data management plattform offered for researchers at RWTH. Internally Coscine uses a project-based system as an internal data model. For each project, any number of data storages can be added. A data storage can further contain multiple data entities which each consist of binaries and metadata records. Following the intention of FAIR digital objects [33], each data storage is assigned a unique PID, which can be used to add further information and which provides a resolution via the *Handle* system [34]. Since the PID is an URI, it is suitable as a node of a *Linked Data knowledge graph*, to represent metadata. Here, the PID with an additional key fragment (since a data storage can contain several data entities) is used as the object for corresponding metadata triples. The predicates are the respective metadata fields and the object is the metadata value. Because RDF was chosen as data format metadata graphs are individually extendable and adaptable. RDF supports the approach of *Linked Data* and supports interoperability and reusability required by the *FAIR Guiding Principles*. However, the flexibility of the RDF model itself does not allow to express specifications or requirements for the metadata. For this reason, SHACL is used to provide shapes for the individual resource graphs, therefore ensuring consistency and reusability of the metadata.

SHACL enables clear structures and restrictions for RDF data. In comparison, ontologies can also define value ranges, but these cannot be validated or due to the OWA allow inconsistencies and supposed contradictions. Ontologies only allow to extract conclusions about existing data. Example 4 demonstrates this.

Example 4: Ontology use

The Organization Ontology (ORG) metadata standard [35] defines the property `org:memberOf`. `org:Agent` is defined as definition range and an `org:Organization` as value range:

```
org:memberOf
 rdfs:domain org:Agent ;
 rdfs:range org:Organization .
```

However, this does not ensure that each triple using the property `org:memberOf` as predicate has an `org:Agent` as subject and an `org:Organization` as object. But in contrast to this, a triple of the form `<x> org:memberOf <y>` is interpreted in such a way that `x` is an `org:Agent` and `y` is an `org:Organization`, although this might not be the case in reality:

(org:memberOf rdfs:domain org:Agent) ∧ (org:memberOf rdfs:range org:Organization) ∧ (<x> org:memberOf <y>) → (<x> a org:Agent) ∧ (<y> a org:Organization)

SHACL supports the validation of RDF graphs against different constraints. Thus, a `sh:NodeShape` allows to validate that the data meets the conditions in any case, otherwise, the data graph does not fulfill the shape graph. In principle, this turns the Open World into a Closed World in which data follows certain principles and rules. The result is a system of semi-structured data due to the fact that on the one hand there are certain (structured) fields and on the other hand there are different SHACL application profiles with different fields, vocabularies, and structures (unstructured).

An application profile defines the structure for metadata to be stored. Within the scope of Coscine, various application profiles are continously developed, each of which is geared to the subject-specific requirements of different disciplines. To ensure uniform use, the SHACL profiles are based on existing metadata schemas. Furthermore, according to the concept of *Linked Data* existing metadata standards and vocabularies can be used. To ensure the use of existing schemas and vocabularies, a knowledge engineer is required who has an overview of existing metadata standards in a variety of domains and who monitors the development process of a new application profile [11].

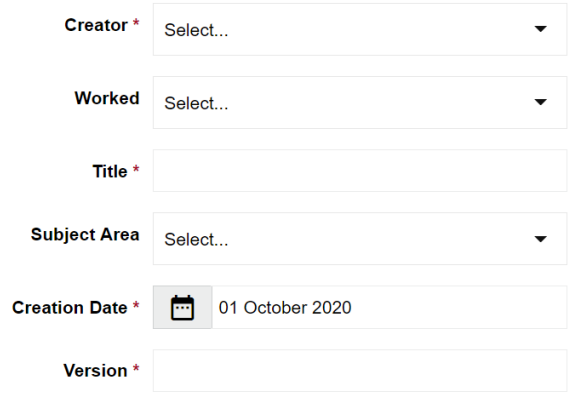

Fig. 1. Generated form from the application profile in Appendix B

Example 5 shows a shortened version of such an application pofile with only one metadata field.

Example 5: Example Application Profile from *EngMeta*

Shortened version of the application profile in Appendix B with only one metadata field:

```
coscineengmeta:
  a sh:NodeShape ;
  sh:targetClass coscineengmeta: ;
  sh:property [
    coscineengmeta:subject
    sh:path dcterms:subject ;
    sh:name
      "Subject Area"@en,
      "Sachgebiet"@de ;
    sh:class dfg:faecher .
  ].
```

The application profile in Appendix B contains the full conversion of the *EngMeta* metadata schema [36] to an application profile for the description of engineering research data from the project *DIPL-ING* [37].

Using the SHACL application profiles, a form is generated, which hides all technical details about the definition and structure of the metadata. Restrictions are checked automatically so that, e.g., mandatory fields are marked as such or a maximum of one element can be selected. Furthermore, the user gets easy access to the ontology, e.g. to the vocabularies. Figure 1 shows the input mask generated from the application profile in Appendix B.

The illustrations shows that class constraints are automatically transformed into drop-down boxes and the user can select an instance without having to know the URI behind it. Data types are also transformed into corresponding form fields.

Within Coscine, a SHACL application profile with the desired metadata fields and restrictions is created as described above. These are adapted to the requirements of a research group, an organization or a project. *Vocabularies* and *Metadata Standards* are reused and created. In the future, this may be supported by an *Application Profile Generator* application. The profile is stored in the *Application Profile Repository* from where it is retrieved by Coscine. A corresponding input mask can be generated if the user wants to create metadata by hand. Alternatively a REST API can be used to submit and validate the metadata using the shape. As a final step, the resulting *RDF graph* is stored via the PID to a resource and the associated research data entity.

Example 6 shows an exemplary metadata record for the previously presented extract of the application profile in Example B.

Example 6: Example metadata record of application profile in Appendix B

```
<http://hdl.handle.net/...@path=x.dat>
  a coscineengmeta: ;
  dcterms:creator
    dbpedia:Thomas_Pynchon ;
  engmeta:worked true ;
  dcterms:title "Design Patterns" ;
  dcterms:subject dfgCS:A409-02;
  dcterms:created
    "2020-01-01+02:00"^^xsd:date ;
  engmeta:version 3 .
```

3. Resource Object Mapping

Based on the SHACL validated resource graphs it is now possible to define a resource object mapping. Two different types of inference rules were established, which are used to create the resulting semantic mapping of resources.

3.1. Literal Rules

The resources to be mapped are described by triples in the knowledge graph. As previously described, each resource (sub-)graph has been validated against a SHACL shape. Within the resource graphs predicates are in the form of URIs and the objects are in the form of URIs or literals. It is assumed that a user generally is not interested in the URIs or does not even know it

but the corresponding label. A matching literal associated by the user to the URI can then be found elsewhere in the graph depending on the instance or class. Generally it cannot be assumed that there always is a suitable `rdfs:label` property available, as described in the approach [38], which could be used to generate a suitable name. However, all used classes and bounding conditions are known and can therefore be used to find literals with the help of respective SHACL shapes.

Additionally, further literals for the description of a resource may exist in the knowledge graph. Compound literals or the direct application of inferences to represent human knowledge as well as relations and to improve the search would also be conceivable. If this implicit information is stored additionally as SHACL rule [39] for each class and thus make it explicit, the user's search query can automatically be applied to the matching literals. In the following, these rules are called *literal rules*. Example 7, 8 and 9 serve as demonstration of these. As can be seen from the three examples, SHACL rules can be used to map literals and structures of any complexity. Since the mapping may need to differ for different shape graphs, they can be added and applied per shape.

Example 7: Literal rule for the class `foaf:Person`

Suppose there exists the following input graph which describes the person Thomas Pynchon:

```
dbpedia:Thomas_Pynchon
 a foaf:Person ;
 foaf:name "Thomas Pynchon"@en .
```

From this, the following graph is to be generated for the literals:

```
dbpedia:Thomas_Pynchon
 rdfs:label "Thomas Pynchon"@en .
```

The following SHACL rule shows the generation of the matching literal of a `foaf:Person`, where `\$this` is the focus node, which is replaced by the respective instance of the class.

```
foaf:Person
 sh:rule [
  a sh:SPARQLRule ;
  sh:construct """
   CONSTRUCT {
    $this rdfs:label ?label
   } WHERE {
    $this foaf:name ?label
   }
  """ ;
 ] .
```

Regarding Example 7, if not only the name but also the first and last name individually or the corresponding organizational unit and organization should be matched, the literal rule in Example 8 is used.

Example 8: Advanced literal rule for the class `foaf:Person`

Assuming the input graph from Example 7, the SHACL rule could be altered like this:

```
foaf:Person
 sh:rule [
  a sh:SPARQLRule ;
  sh:construct """
   CONSTRUCT {
    $this rdfs:label ?label
   } WHERE {
    UNION {
     $this foaf:givenName ?label
    } UNION {
     $this foaf:familyName ?label
    } UNION {
     ?mem a org:Membership .
     ?mem org:member $this .
     ?mem org:organization ?org .
     ?org rdfs:label ?label
    } UNION {
     ?mem a org:Membership .
     ?mem org:member $this .
     ?mem org:organization ?unit .
     ?org org:hasUnit ?unit .
     ?org rdfs:label ?label
    }
   }
  """ ;
 ] .
```

Additionally, literal rules allow returning multiple values for the returned properties to construct lists. As shown in Example 9 this can be used to resolve class or instance hierarchies.

Example 9: Literal rule for the subject classification of the GRF

For instances of the GRF subject classification, the literals of the superclass are also applied because an instance of the subclass is also automatically an instance of the superclass. The following graph is assumed:

```
dfgCS:A409-02
 a dfgCS:A409-02 ;
 rdfs:label "Software Engineering and
  Programming Languages"@en ;
```

Example 9 (continued)

```
 rdfs:subClassOf dfgCS: .

dfgCS:
 a dfgCS: ;
 rdfs:label "Computer Science"@en .
```

From this, the following graph for the literals should be generated:

```
dfgCS:A409-02
 rdfs:label
  "Software Engineering and
   Programming Languages"@en,
  "Computer Science"@en .
```

The following SHACL rule shows the generation of the matching literals:

```
dfg:
 sh:rule [
  a sh:SPARQLRule ;
  sh:construct """
   CONSTRUCT {
    $this rdfs:label ?label
   } WHERE {
    ?class rdfs:subClassOf* ?sup .
    ?sup rdfs:label ?label .
    $this a ?class .
  }
  """;
 ] .
```

3.2. Additional Rules

Furthermore, it is possible to use *additional rules*, which enable the automatic generation of further triples for specific resources. A human being is aware of further knowledge or information due to the specified properties of a resource, which are withheld from a machine and thus from the mapping. The rules are used to slightly close this gap. Example 10 contains a simple additional rule to create a triple indicating whether the referenced data is in the last step.

Example 10: Additional rule for last step

It is assumed that a metadata field `engmeta:step` is filled by instances of the self-created vocabulary `steps` [40]. The instances are processing steps that follow a certain order. An additional triple should be created, which uses a boolean value and the predicate `coscinesc:isLastStep` to indicate if a metadata record is the last step. The following graph is given:

Example 10 (continued)

```
<http://hdl.handle.net/...@path=x.dat>
 engmeta:step coscinestep:storage .

coscinestep:storage
 rdfs:label "Data storage"@en ;
 a <https://purl.org/.../step/> .
```

From this, the following graph should be generated:

```
<http://hdl.handle.net/...@path=x.dat>
 coscinesc:isLastStep true .
```

The following rule creates this additional triple. The focus node `\$this` is replaced by the URI of the resource, in context of Coscine by the Identifier (ID) of the metadata record.

```
<https://purl.org/coscine/ap/engmeta/>
 sh:rule [
  a sh:SPARQLRule ;
  sh:construct """
   CONSTRUCT {
    $this coscinec:isLastStep ?value
   } WHERE {
    $this engmeta:step ?stepA .
    BIND (not exists {
     ?stepA schema:predecessorOf ?stepB
    } as ?value)
   }
  """;
 ] .
```

In Example 10, only a triple is created for the given resource. However, the rules can be more complex and also influence other resources as shown in Example 11.

Example 11: Additional rule for newest version

We assume a property `engmeta:version` contains the version number of a resource in the form of an integer value. It is assumed that the associated data entity is used as a kind of backup, which contains the same data in different versions. Thus, it is possible for a human being to identify the resource with the most current version from a set of resources. However, this information has to be created explicitly for the mapping because a machine cannot automatically develop such connections. Suppose the following input graph exists:

```
<http://hdl.handle.net/...@path=v1.dat>
 engmeta:version 1 ;
```

Example 11 (continued)

```
 coscine:isFileOf coscine:resource1 .

<http://hdl.handle.net/...@path=v2.dat>
 engmeta:version 2 ;
 coscine:isFileOf coscine:resource1 .
```

From this, the following graph should be generated:

```
<http://hdl.handle.net/...@path=v1.dat>
 ex:isNewestVersion false .

<http://hdl.handle.net/...@path=v1.dat>
 ex:isNewestVersion true .
```

The rule for creating a field indicating the newest version in a set of resources based on the property `engmeta:version` and the data model of Coscine [41] is as follows. Furthermore, this rule does not only affect a single resource but must always be considered in the context of the given set of resources.

```
<https://purl.org/coscine/ap/engmeta/>
 sh:rule [
  a sh:SPARQLRule ;
  sh:construct """
    CONSTRUCT {
      ?s ex:isNewestVersion ?value
    } WHERE {
      ?s engmeta:version ?ver .
      ?s coscine:isFileOf ?res .
      $this coscine:isFileOf ?res .
      {
        SELECT ?newVer
        WHERE {
          ?file engmeta:version ?newVer .
          ?file coscine:isFileOf ?res .
          $this coscine:isFileOf ?res .
        } ORDER BY DESC(?newVer) LIMIT 1
      } .
      BIND( ?ver = ?newVer AS ?value)
    }
  """ ;
 ] .
```

Also, the additional rules do not have to generate boolean objects. With such rules, any additional information can be generated automatically or additionally to extend or improve the mapping. It is possible to define general or SHACL shape specific additional rules.

The approach considered is based on creating a JSON object from the RDF graphs to be able to ingest that into existing search engines. As advantages for the search, it is possible to benefited from functionalities and optimizations for user input and search features. Examples are the handling of grammatical differences or spelling and the use of auto-completions or search suggestions. The possibilities that a search index provides for the user and generally the optimization of the search sound very promising. Evaluating this mapping would therefore support the core hypothesis (see section 1.2).

3.3. Semantic Mapping of a Resource into a JSON Object

A JSON object consists of properties and associated values. Furthermore, different data types like `boolean`, `number` and `string` are supported for the values[2]. For all properties, a name and an appropriate value must be created from the resource graph triples to describe the corresponding field. In case of a literal it is taken as a value. The literal rules are used for instances. According to the same principle, additionally generated triples can be stored in the JSON object by using the additional rules. Thus, any arbitrarily complex graph is transformed into a flat JSON object, which consists only of field-value(s) pairs, as specified in the following definition:

Definition 1: Semantic Mapping of a Resource into a JSON Object

Given is resource graph G from a knowledge graph K such that G is a sub graph of K with a root node S_0 identifying a resource to be mapped to a JSON object, the edges correspond to the properties and the objects to the respective values.
For each triple $\langle S, P, O\rangle \in G$, $S \in \boldsymbol{I}$, $P \in \boldsymbol{U}$ and $O \in (\boldsymbol{U} \cup \boldsymbol{L})$ applies, where $\boldsymbol{I}$ denotes resources, $\boldsymbol{U}$ denotes URIs, and $\boldsymbol{L}$ denotes literals. Then G' is the resulting mapped graph of G by applying the SHACL rules on K.
The mapping to a JSON object then is a tree $\hat{G}$ with exactly height one that includes triples from G' such that $\forall_{i,j}(\langle S'_i, P'_i, O'_i\rangle, \langle S'_j, P'_j, O'_j\rangle) \in G' : S'_i = S'_j = S_0$ and $\forall_i \langle S'_i, P'_i, O'_i\rangle \in G' : S' \in \boldsymbol{I}$, $P' \in \boldsymbol{S}$, and $O' \in \boldsymbol{L} \cup \boldsymbol{L^n}$ is valid, where $\boldsymbol{S}$ are strings, and $\boldsymbol{L^n}$ are lists of literals.

To demonstrate the mapping Example 12 is considered.

[2] In the ES index `date` and number formats like `integer` and `float` are added and `text` replaces `string`.

Example 12: Mapping of an RDF metadata record into an ES document

Again, the example metadata record of Example 6 from section 2.6 is considered. Furthermore, it is assumed that the following additional information is stored in the knowledge graph:

```
dfg:EngineeringSciences
 a dfg:EngineeringSciences;
 rdfs:label "Engineering Sciences"@en ;
 rdfs:subClassOf dfg: .

dfgCS:
 a dfgCS: ;
 rdfs:label "Computer Science"@en ;
 rdfs:subClassOf ⌋
   dfg:EngineeringSciences .

dfgCS:A409-02
 a dfgCS:A409-02;
 rdfs:label "Software Engineering and ⌋
   Programming Languages"@en ;
 rdfs:subClassOf dfgCS: .

rwth:
 a org:FormalOrganization ;
 rdfs:label "RWTH Aachen University" ;
 org:hasUnit rwth:U022000 .

rwth:U22000
 a org:OrganizationalUnit ;
 rdfs:label "IT Center" .

dbpedia:Thomas_Pynchon
 a foaf:Person ;
 foaf:givenName "Thomas"@en ;
 foaf:familyName "Pynchon"@en ;
 foaf:name "Thomas Pynchon"@en .

[]
 a org:Membership ;
 org:member dbpedia:Thomas_Pynchon ;
 org:organization rwth:22000 .
```

If it is assumed that only the presented rules from the Examples 7 and 9 from section 3.1 and no other additional rules are applied, the following ES document results from the mapping:

```
{
 "creator": [
  "Thomas",
  "Pynchon",
  "Thomas Pynchon",
  "IT Center",
  "RWTH Aachen University"
 ],
 "worked": true,
 "title": "Design Patterns",
 "subject": [
  "Software Engineering and ⌋
     Programming Languages",
  "Computer Science",
  "Engineering Sciences"
 ],
 "created": 2020-01-01,
 "version": 3
}
```

It is important that all URIs are transformed into literals or lists of literals during this transformation. For this reason, the mapping of Definition 1 is not injective. It may happen that two different RDF resources map to the same JSON object as shown in Example 13.

Example 13: Mapping into a JSON object is not injective

Given are the two RDF triples:

```
ex:SWEP
 rdfs:label "Software Engineering and ⌋
   Programming Languages" .

dfgCS:A409-02
 rdfs:label "Software Engineering and ⌋
   Programming Languages" .
```

Assuming there are two metadata records which differ only in one metadata value (the two URIs) and no other rules are applied. The corresponding mapping could not be distinguished because the URIs are mapped to the same literal and thus the same ES document is created.

3.4. Resolving Ambiguous Property Data Types

Ideally, the same properties are defined in different SHACL shapes with the same data type or are described using instances of one class. In this case, they can receive exactly this defined data type as type in ES (`date`, `text`, `boolean` or `integer`) or instances of a class are always mapped as `text`. If in a knowledge graph several different resources were validated based on a shape and the same properties in different shapes have different data types, a generic data type should be found, which is then generally valid. The property is mapped to the type `text`, because as mentioned before the same property in ES may only have one unique type per index. Table 1 shows all possible mappings of data types between the existing SHACL shapes and ES.

Table 1
Data types mapping between SHACL shapes and ES where * means that the same properties were described in different shapes with the same data type.

SHACL Shape	JSON	ES
`date*`	`string`	`date`
`string*`	`string`	`text`
`boolean*`	`boolean`	`boolean`
`integer*`	`number`	`integer`
`class*`	`string`	`text`
different data types	`string`	`text`

4. Evaluation[3]

During the evaluation of the resource object mapping, the context of Coscine was used and the mapping of metadata records in the context of searchability was evaluated. For the considered tests described in this section exemplary metadata records were generated. All of them are based on a modified form of the EngMeta application profile [43]. Some classes and data types of the profile have been modified to demonstrate the different possibilities that the SHACL definitions provide. Furthermore, the data does not represent real metadata of research data records but serves for illustration.

The following results are based on the assumptions made in the context of the paper, the presented test setup, implementation, and the generated sample data.

4.1. Evaluated Search Intentions

For testing the research hypothesis some search queries were considered. Behind each query is the intention to find one or more data records (which are described by metadata). They were selected to cover as many different search types as possible and to be related to the RDM use case:

- Data records which were published at the IT Center
- Data records with version number 10
- Data records which were created in 2007
- Data records about design patterns and with version number less than 5
- Data records about computer science
- The newest data record of resource 2
- Data records which are experiments or simulations and not analysis
- Data records about non computability of the human consciousness
- Data records which are available since 03.07.2020
- Data records about political left
- Data records which contain a variable with the value 2 meters
- Data records about object-oriented software which are published before 2015
- Data records which are published by Thomas Pynchon, created in 2016 and about object-oriented software

This intention has to be formulated differently as a request, because a certain syntax has to be followed when using ES [44]. Furthermore, the search query is formulated in such a way that it is at least well-fitted or close to optimal. This means that the intention "published on IT Center", e.g., in the query is translated to `publisher:IT Center` and thus the correct property is used. Also, it is assumed that the user searches for the correct words, i.e. those that are present in the text and not for variants or synonyms of them.

4.2. Test Setup

.NET Framework 4.8 was used with the programming language *C#* and the library *dotNetRDF* [45] was included in the version 2.6. As RDF database *Virtuoso* in the open-source variant *7.20.3217* with the default settings was applied. Only the parameter *NumberOfBuffers* was changed to 660000 and *MaxDirtyBuffers* to 495000 in the *virtuoso.ini* file. For the implementation ES version *7.6.2* was used. In addition, all default settings have been adopted, i.e., in particular only one *node*, one *cluster*, one *shard*, and one *replica* have been created. In order to keep the conditions, especially for the measurement of times and their comparison, as similar as possible, the tests were carried out under the same conditions and the same hardware. A 64-bit system with the operating system *Microsoft Windows Server 2016 Standard*, 16GB RAM and two *Intel(R) Xeon(R) CPU E5-2695 v3 @ 2.29GHz* processors was used.

4.3. Precision and Recall

To evaluate precision and recall the presented search intentions were considered. First, the search inputs for the ES syntax were created (see Appendix D) and then executed on 35 selected metadata records. To see which metadata records are relevant, the query was translated

[3]The evaluation is based on data record [40] and evaluation code [42] discussed in detail in [41].

Table 2
Summed up confusion matrix

Received \ Relevant	+	-
+	51	2
-	0	402

Table 3
Precision, recall, and F1 score (rounded to two digits)

	Mapping
Precision	0,96
Recall	1
F1 Score	0,98

as a reference into a pure SPARQL query considering the existing structure and data (see Appendix C). A full-text search, i.e., if searching in strings and no corresponding URI exists, is only possible by using an regular expression or the function *bif:contains* provided by Virtuoso. Based on the results of the query it was decided which records should be found for the respective search intention. Afterwards, confusion matrices (see section E) were set up, which compare the relevant search results with the found ones and thus make a concrete analysis possible. Table 2 contains a summed up matrix, which adds up the results of the different search intentions.

The confusion matrix shows that all relevant records were found and only the two records were irrelevant. The advantages of this approach are that a search can be done in concrete fields, any combination of fields is possible and specification of value ranges is allowed. Furthermore, the year and boolean values (due to the search in concrete fields) can be found without additional created fields. Besides, a ranking of the found records is also available. Additional metadata records could be found for specific phrases containing stop words. If there had been records containing the word center as a publisher, they would also have been found in the search intention "published at the IT center". This is because the word "it" counts as a stop word and thus only the word "center" is searched for. This is not shown in the example queries or data but would be a disadvantage.

The search intention "left (political)" finds two irrelevant records. This is because a pure search for "left" leads to the fact that in addition to the political left, the left side or the past tense of the word "leave" can also be meant. For a computer, this semantic difference is not recognizable. However, the search could be extended by the word `political` or a wildcard variant `politic*` and would in this case only find the one relevant data record. However, this requires that the word also occurs in the same string. If this is not the case, it is not possible to distinguish between them. If the topic would be political science or similar (which is not the case in the data), the search could also be further limited by this information. This semantic problem also exists when using SPARQL queries and is not a disadvantage compared to them.

Table 3 summarizes the results by calculating precision, recall, and F1 score of the total confusion matrix. The resource object mapping reaches the same values as generated SPARQL queries and is therefore not inferior.

4.4. Response Time

The response time was measured for all presented search intentions on the search data record. Nielsen [46] differs several types of response times: One limit is set to 0.1 seconds so that the user feels that the system is responding directly to the interaction. Another limit is 1.0 second: the user notices the delay but remains uninterrupted. This concludes that the search results (i.e., from entering the search query to displaying the found metadata) should be found within 0.1 seconds if possible and within one second at most.

To make the comparison of the reponse times as fair and realistic as possible, an executable file was created, which receives the query (see section 4.3) and performs the corresponding function with it. The function returns a list with the PIDs of the found metadata records. It was omitted to run through and output of these results to avoid that large result lists influence the runtime. With the executables, an attempt is made to map a complete user request, which goes beyond simply executing the request against a database. The search input of the user has to be sent to the backend to add the visibility restrictions. However, in the future, a web service will be running, and not every time a process will be started that reloads the used libraries. Therefore a basic process was created for comparison, which loads the libraries for *dotNetRDF.Data.Virtuoso* needed in all approaches and queries a simple graph in Virtuoso. An execution time of 221 ms was measured. Also, this type of measurement was chosen because it is not about comparing different queries within an approach, but about the differences between the approaches and the time elapsed for the user. To get a more accurate measurement, in each approach the executable was executed for each query 100 times in a row and then the average value was

Table 4
Response time in ms (rounded to a full number) of user request in small search data record

Intention	Guideline	Mapping
Published at IT Center	311 ± 21	572 ± 78
Version 10	301 ± 16	550 ± 21
Created in 2007	294 ± 10	552 ± 23
Design Patterns and Version < 5	295 ± 12	552 ± 24
Computer Science	295 ± 16	548 ± 24
Newest Version Res. 2	311 ± 8	558 ± 29
Experi./Simu. not Anal.	311 ± 10	555 ± 22
Non Comput. Human Consci.	302 ± 27	556 ± 26
Av. s. 03.07.20	303 ± 25	556 ± 18
Left (politicial)	295 ± 19	553 ± 25
2 Meters	311 ± 30	555 ± 29
Pub. < 2015 Oo. Softw.	288 ± 17	551 ± 23
Pub. T. Pyn. Created 2016 Oo. Softw.	298 ± 24	556 ± 27
∅	301 ± 18	555 ± 28

Table 5
Response time in ms (rounded to a full number) of user requests in large search data record

Intention	Guideline	Mapping
Published at IT Center	316 ± 28	557 ± 69
Version 10	305 ± 17	549 ± 20
Created in 2007	308 ± 9	551 ± 20
Design Patterns and Version < 5	290 ± 11	548 ± 23
Computer Science	373 ± 11	549 ± 29
Newest Version Res. 2	313 ± 9	550 ± 18
Experi./Simu. not Anal.	415 ± 16	554 ± 18
Non Comput. Human Consci.	348 ± 129	556 ± 31
Av. s. 03.07.20	336 ± 12	556 ± 20
Left (politicial)	292 ± 7	558 ± 33
2 Meters	317 ± 30	549 ± 23
Pub. < 2015 Oo. Softw.	291 ± 18	549 ± 21
Pub. T. Pyn. Created 2016 Oo. Softw.	291 ± 18	550 ± 26
∅	323 ± 24	552 ± 27

calculated. The results are shown in Table 4. Besides, the standard deviation was calculated to illustrate the inaccuracies of the measurements.

4.5. Scalability

Coscine is a system that will grow steadily the more metadata about research data is stored and collected. That means that the system should be able to handle large amounts of data and enable searching in this amount of data.

To measure the scalability, the search queries were executed on the 10,000 metadata records large data record the same way described the section above. The results of the response times on the large data record are shown in Table 5.

For the assessment of the scalability, the factor by which the execution times have increased are considered since they indicate how the response times will behave on even larger data records. Table 6 shows the factor between the respective values of the average times from the tables 4 and 5.

Table 6
Factor (rounded to two digits) by which the response time has increased compared to the small data record

Guideline	Mapping
1.07	0.99

4.6. Additional Effort

This category describes the additional effort (storage and calculations) required to implement the respective approach which is necessary in addition to the previous storage of metadata records in the RDF database. In addition to the storage of the literal rules described in section 3.1, the implementation of the additional administrative triples and the mapping of the metadata records for the search index from section 3 is required. This results in completely redundant storage of the ES mappings of all metadata records. Besides the additional memory consumption, this also leads to necessary synchronization steps. The already existing data must be indexed at the beginning. Besides, new metadata records must be created, updated, and deleted synchronously. If application profiles change or new ones are added, in the worst case a new index must be created.

The times to perform these actions are shown in Table 7. As with the measurement of the response times, executable files were created for this purpose, whose execution time and standard deviation were measured. Since the initial indexing has to be done only once at the beginning and the reindexing differs only by deleting the old index and switching the alias, which are fast queries, only the reindexing was measured. Due to its duration, it was executed ten times independently with the 10,000 test data and the average was given. The creation, updating, and deletion of individual records were performed 100 times in a row and then the average value was calculated. The additional rule for saving the latest version (see Example 11) was used once and once without it, because it can affect other metadata records and thus increases the execution time. When (re)-indexing, the additional rules are always executed,

Table 7
Times in ms (rounded to a full number) for the actions of the additional effort. Columns with * show values where additional rules which influence other metadata records were used.

(Re)-Indexing	Add	Update	Delete
1646706 ± 53376	837 ± 143	839± 129	735 ± 121

Add*	Update*	Delete*
3055 ± 233	2996 ± 97	2901 ± 148

but they do not affect the other documents every time, since all knowledge is already available in the knowledge graph and thus can be directly queried correct and complete. To see the effects of the additional rules, the actions were executed on the large data record. For adding, 9900 records already existed and the remaining 100 were added. When updating and deleting, all 10,000 records were already indexed.

It is important that these delays are not noticeable for the user, because they can run parallel in the background without limiting the search possibilities. Furthermore, it is clearly visible that creating, updating and deleting is faster if the other metadata records are not affected. In order to see the relation between the additional time required to add a metadata record and the time required to store and validate it in Virtuoso, the execution time for running an executable with this function was measured. The average value for 100 consecutive executions is 3217 ms. This time was calculated using a small metadata record that only fills the metadata fields `dcterms:author`, `dcterms:title`, `dcterms:subject`, and `dcterms:created`. It follows that the additional mapping in ES results in a further time expenditure of 26%. This number shows that the main effort lies in the necessary validation and storage in Virtuoso.

5. Related Work

To enable a search in an RDF-based knowledge graph there are so-called *SPARQL query builder*, which allow a step-by-step construction of SPARQL queries using cleverly chosen user interfaces. There is a wide variety of such query builders. Kuric et al. [47] have compared the best-known query builders regarding their usability for laypeople. Since such tools all suffer from usability problems and have already been extensively evaluated [47–49], this paper focuses on an alternative variant: mapping the graph data into a smart search index.

Related to this topic is the graph summarisation by Campinas [50], where this process takes place in reverse order. From an existing knowledge graph, a summary is constructed that represents the structure. In our context, the structure already exists through SHACL validations with whose assistance JSON objects are to be generated from individual resources.

Delbru et al. [51] describe an *Entity Retrieval Model* for Web Data to describe semi-structured information from heterogeneous and distributed sources for semi-structured information. The model consists of three components: *dataset*, *entity*, *view*. A *dataset* is a collection of entity descriptions and can be uniquely referenced by an URI. An *entity* is something that can be defined via an URI (such as documents, events or persons) and for which descriptions exist in the form of properties. A *view* is a piece of information accessible via an URI which is used to describe the dataset. The *Entity Attribute-Value Model* [51] is a model to describe the entities of a dataset. This is done with an ID describes the entity, a set of labeled properties, and a set of values for these properties. Furthermore, Delbru presents a search model for web data which, in contrast to conventional web search engines, does not represent entities as a bag of words but describes them as a set of attribute-value pairs. For the search, three different search types can be distinguished: (i) *full-text queries* which represent a search request with unknown structure, (ii) *structural queries* which represent more complex queries in form of key-value-pairs if the structure is known and (iii) *semi-structural queries* as the combination of both. In principle, the *Entity Attribute-Value Model* and the corresponding *Search Model for Web Data* are applied in the presented approach, where the metadata records are the entities. Resources (in the context of Coscine only the metadata records) are considered as documents, the corresponding properties (metadata fields) as fields, and all objects as values. However, the model serves only as a structure. The work presents how such a mapping is created smartly and semantically.

Linked swissbib [52] converts bibliographic data into a RDF-based data model and divides it into six different concepts. By using the *JSON-LD* serialization of RDF, they are indexed in the search engine *ES*. Therefore, the approach precedes as a good example for mapping RDF data into a search index. However, it is outdated, since *ES* no longer supports the use of different types in a search index [53]. Furthermore, the pure use of the *JSON-LD* serialization in our context does not make sense, because otherwise URIs would be indexed for which the users do not search.

For this reason, *Open Semantic Search* [54] additionally indexes the corresponding `rdfs:label` to a URI. Nevertheless, this is not sufficient since it is not always set or can be found in other properties depending on the data structure. In this approach the associated labels of the RDF annotation to URIs for indexing data in Solr are stored.

Semplore [55] translates Semantic Web data into documents, fields, and terms properly so that the IR engine can index them in their inverted index structure and provide retrieval functions via the data. The presented approach for mapping objects into a search index is oriented on Semplore. The differences are the concrete generation of the mapping, which in the considered approach can contain, for example, inference rules, and the possibility to use ES to benefit from advanced search types like value range queries.

Rocha et al. [56] presented a hybrid approach to keyword-based search in the Semantic Web, where the focus is on finding concepts. For this purpose, an instance graph is created for each concept based on the values of its properties, which is searched with traditional IR techniques. Afterwards, spread activation techniques are applied to find related concepts. A knowledge engineer has to weight the paths in the graph to the related concepts since they are always domain-dependent. In the context of Coscine, the domain is not known in advance or can change by further application profiles and application areas. Thus, such an approach is not suitable. Changes and adjustments would have to be made again and again, which is why it is not pursued further. Consequently, the creation of the instance graphs is interesting, which store all terms from associated properties for a concept to make them referenceable.

The main contribution of the developed approach is the use of SHACL as the basis for the creation of the inference based mapping. By using graph shapes a validated knowledge graph exists whose structures, vocabularies, and terms are known. Additionally, inference rules can be created and applied to the graph. This makes it possible to store human knowledge and use it for resource object mapping. Furthermore, it allows the transformation of unrecognized and unreadable URIs into suitable human-readable literals, which can be used to describe resources.

6. Conclusion and Future Work

This paper introduced a semantic resource object mapping for the ingestion of SHACL validated resource graphs into a search engine. It builds upon SHACL and the SHACL-SPARQL extension to allow the definition of inference rules that are used to generate object properties. The mapping was further evaluated regarding its performance with respect to precision and recall when prepared to generated SPARQL queries.

As the results show, the resource object mapping when using ES as search engine is inferior to the manual generation of SPARQL queries in terms of speed, but it also clearly meets a maximum response time of one second that is common for web applications. This applies to both the small and large data record ensuring scalability of the mapping. However, both approaches result the same in terms of effectiveness: For the example metadata graphs and search intentions, they achieve the same precision and recall values.

Two disadvantages of the mapping compared to the use of SPARQL queries are its additional resource consumption regarding memory and computing time as well as the additional storage of rules. Since both are not directly visible to the users and do not bring any disadvantage for them, this point of criticism is not to be taken very serious. SHACL rules only have to be created once and can be reused. They also allow a very flexible and customizable extension, which has a great influence on the search results.

The definition of rules has the further advantage that a user no longer has to think about such structures and connections of the RDF data and individual users do no longer have to map, e.g., the transitive relationship of the subclasses for hierarchies (see Example 7 and 9). The user might not even be aware of these relations, so it is advantageous to include them from the beginning. The necessary knowledge about the inference rules and the internal structure of the graph is pushed to a knowledge engineer building the SHACL shapes. Naturally, this implies that only such relations can be found in the search, which are mapped by rules for the search index. For example, if the transitive rule for the subclasses of the GRF subject classification is not used, no subclasses can be found in the search for a superclass. This means that all knowledge or structures must be aware of and considered in advance.

6.1. Answering Research Question

In the following, the answers to the research questions posed in section 1.2 are briefly discussed.

How to exploit features of SHACL for resource object mapping?

The main challenges when querying RDF knowledge graphs are the unknown structure and URIs. These challenges have been eliminated due to the mapping and the application of the literal rules, as more complex graph structures and URIs are thus resolved. The SHACL application profiles form an essential component in the implementation of the resource object mapping approach. Based on the assumption that resource graphs are SHACL validated SHACL rules can be used to build (specific) literal and additional rules (see section 3.1). The rules allow the extension and mapping of a resource graph into an JSON object (see section 3).

How does such a mapping influence the search results in terms of precision and recall?

The advantages of mapping to a search index are that every user can make complex search queries because the provided search syntax of ES is very simple and intuitive and no further knowledge is required. Besides, the data structure, vocabularies, and schemas are hidden. The mapping delivers successful results and reasonable times due to the previous indexing of the data. Besides, ES is specifically designed for scalable and fast searches [16]. Since SPARQL is specially designed as a query language for RDF graphs ES cannot fully keep up with the speed of executing SPARQL queries. Through ES different search types like keyword search, value ranges, boolean queries, and full-text search are possible. For the handling of large amounts of data, it was considered how the approach works with large amounts of data (see section 4.5).

With both answers at hand and the discussed evaluation in Section 4, this allows assessing the initially posed research hypothesis:

A mapping of RDF-based metadata records into a search index exists such that a search engine on that index yields the same precision and recall when compared to generated SPARQL queries.

The results of the evaluation largely confirms the hypothesis in the considered context when using application profiles. The mapping of resource graphs into a search index for use in a search engine yields the same precision and recall as manually generated SPARQL queries. For this reason the same results can be achieved with much less effort for the user. However, this is only possible by using previously carefully considered inference rules for mapping literals and further relations or cognitions.

6.2. Discussion

In this work RDF data, SPARQL, and SHACL are used. SPARQL queries map semantic knowledge in the form of SHACL rules. This knowledge includes, e.g., inference rules like subclasses or the mapping of structures to search in the corresponding literals of URIs. This allows capturing existing semantics into the object mapping.

The introduced mapping is limited to SHACL shapes being available to specify corresponding fields and data types. Only with this information an index can be created in ES in advance. Furthermore, when creating different shapes, it is important to ensure that the same data types are used for the same properties, to provide a data type specific mapping in ES instead of a string representation. In turn, a property should be used consistently for the same metadata fields throughout different shapes, otherwise there will be ambiguous mappings for the same property (see Example 14). In any case, reuse of existing shapes and metadata schemas reduces the impact of this issue and is generally desirable.

Example 14: Ambiguous definition of properties

In an application profile, the metadata field of the author is described via the property `dcterms:creator`, where the associated label is "creator".

```
coscineengmeta:creator
  sh:path dcterms:creator ;
  sh:name "Creator"@en .

dcterms:creator
  rdfs:label "creator" .
```

In a second application profile, the property `ex:author` is defined with the associated label "author".

```
ex:creator
  sh:path example:author ;
  sh:name "Creator"@en .

ex:author
  rdfs:label "author" .
```

This leads to the fact that in JSON object there will be a field `creator` and a field `author`, although semantically they may describe the same concept. In a third application profile, the property `ex:author` is defined with the associated label "creator".

Example 14 (continued)

```
ex:creator
  sh:path example:author ;
  sh:name "Creator"@en .

ex:author
  rdfs:label "creator" .
```

Hence, the values would be merged by the proposed algorithm even though they may not semantically describe the same concept.

In this example it might be considered to check if such a field already exists in the ES index and use it for both properties. This has to be considered when creating the data types. However, it would have to be ensured that both properties are actually referring to the same concept. This decision is not necessarily unambiguous and trivial, and is an area of research of its own. If two different metadata fields (which also have a different semantic meaning) are identified with the same label, which can occur due to homonyms, a workaround must be developed so that the fields can be clearly distinguished and offered to the user as search fields. Different solutions exists like prefixing or requiring the knowledge engineer creating an application profile to provide an alternative name. The resource object mapping generally follows a CWA and as such assumes that properties with different names are indeed different. However, an extension to respect definitions like `owl:sameAs` could provide a reasonable extension. The whole example shows the problem of the semantic gap [57], namely that the same data or information can be represented using different vocabularies and RDF terms. An arbitrary number of different graphs exists, which represent the same semantic information. Within the context of Coscine, this problem is countered by the uniform creation of application profiles, the reuse of both metadata schema and vocabularies, as well as the controlled creation of literal and additional rules.

For the creation of the literal and additional rules a knowledge engineer with SPARQL knowledge is required. Although this saves the user a lot of cognitive work, the creation of application profiles is more complex. When creating the rules, it is especially important to pay attention to their meaningfulness. The rule from the example 8, for example, also creates the corresponding institutional organization for a person as value. This makes sense in the context of a publisher metadata field. However, it must also be taken into account that the associated institutional organization can change in the knowledge graph. Consequently, the new institute is stored instead of the previous specified at the time of publication, which makes less sense.

Another aspect worthy of discussion is the arbitrary extensibility of the rules. The question arises, if any rule can be constructed. Anything can be mapped that can be constructed using the SPARQL `CONSTRUCT`s based on the database, if the corresponding instance of a class (literal rule) or the URI of a resource (additional rule) is used as focus node. In general, the rules are very generic and can be used across all application profiles. It is crucial that a knowledge engineer decides on the basis of the rules which information of the metadata graph is mapped and thus is searchable and which information is lost through the mapping according to ES and thus cannot be searched. Furthermore, in the evaluation it was found that the use of additional rules that influence other metadata records essentially increases the indexing time of a new metadata record. For this reason, they should only be used very rarely and with caution.

6.3. Future Research Suggestions

The entire evaluation refers to the context of research data and Coscine (especially the use of application profiles which provide a specific structure). The sample data was generated and the search intentions were devised with exactly this focus. In addition, the search for metadata records is also a specific issue. Other entities like single persons should not be found. In addition, the most optimal search queries were assumed, i.e., that the user does not make any mistakes during the input and that the structure and appropriate choice of words for the approach were adhered to. To what extent such an approach is transferable to other use cases and whether ES can be generally considered a suitable search engine for arbitrary RDF knowledge graphs remains an open question for future research projects.

Within Coscine, there are currently no connections between the individual metadata records. With the expansion to resources, a further area of research could be the linking of the resources to each other, which is currently completely lost through mapping into the literals. It is no longer clear whether a literal was created by a resource or was specified as such.

Depending on the vocabularies and data used, it might be interesting to enrich the knowledge graph with information from other data sources according to the principle of Linked Data. In this way, further rules could be mapped.

Up to now, a knowledge engineer is required to map the literal and additional rules. An interesting question is whether it is possible to have these rules created by a user without the appropriate knowledge. For example, would a user interface be conceivable in which rules can be clicked together based on the vocabularies, metadata standards, and ontologies used? Thinking the other way around, the question arises whether and how it is possible to generate such rules automatically when creating graph summaries as proposed by Campinas [50].

6.4. Closing Remarks

In conclusion, the main finding of this work is that the use of search engines can also be suitable for searching in RDF-based knowledge graphs if a skillful mapping of the data and the knowledge contained in its structure is applied.

Two more subtle but noticeable advantages of ES as a search engine were experienced but their effect was not further investigated during this research: (i) ES is additionally able to map synonyms and find results by using stemming without having to map them literally in the data. Just the possibility to enter singular or plural words leads to an extreme increase in user satisfaction and error tolerance. In this case, it is no longer necessary to assume that the search is optimal in every case, so that, for example, a search can be made for `2 meters` instead of `2 meter`. (ii) ES brings along a ranking compared to SPARQL. This is especially helpful if a search query returns many results. This can be used to highlight metadata records for which a certain search query is more specific than others. Both features increased the experiences quality of search results significantly.

Acknowledgements

The work was partially funded with resources granted by *NFDI4Ing* and the project *AIMS* funded by GRF under project number 432233186.

References

[1] RWTH Aachen University, Forschungsdatenmanagement von A - Z. https://www.rwth-aachen.de/cms/root/Forschung/Forschungsdatenmanagement/~svkj/A-bis-Z/.

[2] Jisc, Describing metadata. https://www.jisc.ac.uk/guides/metadata/describing-metadata.

[3] M. Kindling and P. Schirmbacher, "Die digitale Forschungswelt" als Gegenstand der Forschung / Research on Digital Research / Recherche dans la domaine de la recherche numérique, *Information - Wissenschaft & Praxis* **64**(2–3) (2013). doi:10.1515/iwp-2013-0017.

[4] M.D. Wilkinson, M. Dumontier, I.J.J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.-W. Boiten, L.B. da Silva Santos, P.E. Bourne, J. Bouwman, A.J. Brookes, T. Clark, M. Crosas, I. Dillo, O. Dumon, S. Edmunds, C.T. Evelo, R. Finkers, A. Gonzalez-Beltran, A.J.G. Gray, P. Groth, C. Goble, J.S. Grethe, J. Heringa, P.A.C. 't Hoen, R. Hooft, T. Kuhn, R. Kok, J. Kok, S.J. Lusher, M.E. Martone, A. Mons, A.L. Packer, B. Persson, P. Rocca-Serra, M. Roos, R. van Schaik, S.-A. Sansone, E. Schultes, T. Sengstag, T. Slater, G. Strawn, M.A. Swertz, M. Thompson, J. van der Lei, E. van Mulligen, J. Velterop, A. Waagmeester, P. Wittenburg, K. Wolstencroft, J. Zhao and B. Mons, The FAIR Guiding Principles for scientific data management and stewardship, *Scientific data* **3** (2016), 160018. doi:10.1038/sdata.2016.18. https://pubmed.ncbi.nlm.nih.gov/26978244/.

[5] NFDI, Nationale Forschungsdateninfrastruktur NFDI. https://www.nfdi.de/informationen.

[6] RfII, Themen - RfII. http://www.rfii.de/de/themen/.

[7] DFG, National Research Data Infrastructure. https://www.dfg.de/en/research_funding/programmes/nfdi/index.html.

[8] D. Schmitz and M. Politze, Forschungsdaten managen – Bausteine für eine dezentrale, forschungsnahe Unterstützung: 76-91 Seiten / o-bib. Das offene Bibliotheksjournal / herausgegeben vom VDB, Bd. 5 Nr. 3 (2018) / o-bib. Das offene Bibliotheksjournal / herausgegeben vom VDB, Bd. 5 Nr. 3 (2018), *o-bib. Das offene Bibliotheksjournal / Herausgeber VDB* **5**(3) (2018), 76–91. doi:10.5282/o-bib/2018H3S76-91.

[9] RWTH Aachen University, Leitlinie zum Forschungsdatenmanagement an der RWTH Aachen. https://www.rwth-aachen.de/cms/root/Forschung/Forschungsdatenmanagement/~ncfw/Leitlinie-zum-Forschungsdatenmanagement/.

[10] M. Politze, F. Claus, B. Brenger, M.A. Yazdi, B. Heinrichs and A. Schwarz, How to Manage IT Resources in Research Projects? Towards a Collaborative Scientific Integration Environment, in: *European Journal of Higher Education IT 2020-2*, Y. Epelboin, M. Mennielli, A. Pacholak, P. Kähkipuro, G. Ferell, C. Diaz, L.M. Riberio, J. Bergström, T. Koscielnieak, E. Cardoso, R. Vogl, B. Cordewener, N. Wilson, C. Vilarrasa, N. Harris and O. Tasala, eds, 2020.

[11] M. Politze and B. Decker, Ontology Based Semantic Data Management for Pandisciplinary Research Projects, in: *Proceedings of the 2nd Data Management Workshop*, C. Curdt and C. Wilmes, eds, Kölner Geographische Arbeiten, Cologne, Germany, 2016. doi:10.5880/TR32DB.KGA96.10.

[12] M. Politze and T. Eifert, On the Decentralization of IT Infrastructures for Research Data Management, in: *European Journal of Higher Education IT 2019-1*, Y. Epelboin, M. Mennielli, A. Pacholak, P. Kähkipuro, G. Ferell, C. Diaz, L.M. Riberio, J. Bergström, T. Koscielnieak, E. Cardoso, R. Vogl, B. Cordewener, N. Wilson, C. Vilarrasa, N. Harris and O. Tasala, eds, Paris, France, 2020.

[13] T. Kálmán, D. Kurzawe and U. Schwardmann, European Persistent Identifier Consortium - PIDs für die Wissenschaft, in: *Langzeitarchivierung von Forschungsdaten*, R. Altenhöner and

C. Oellers, eds, Scivero Verl., Berlin, Germany, 2012, pp. 151–164. ISBN 978-3-944417-00-4.

[14] M. Politze, S. Bensberg and M. Müller, Managing Discipline-Specific Metadata Within an Integrated Research Data Management System, in: *Proceedings of the 21st International Conference on Enterprise Information Systems ICEIS 2019, Heraklion, Crete - Greece, May 3 - 5, 2019*, J. Filipe, ed., ICEIS (Setúbal), SciTePress, [S. l.], 2019, pp. 253–260. ISBN 978-989-758-372-8. doi:10.5220/0007725002530260.

[15] Deutsche Forschungsgemeinschaft, Gute wissenschaftliche Praxis. https://www.dfg.de/foerderung/grundlagen_rahmenbedingungen/gwp/.

[16] Elasticsearch B.V., What is Elasticsearch? | Elasticsearch Reference [master]. https://www.elastic.co/guide/en/elasticsearch/reference/master/elasticsearch-intro.html.

[17] H. Arnaout and S. Elbassuoni, Effective Searching of RDF Knowledge Graphs, *SSRN Electronic Journal* (2018). doi:10.2139/ssrn.3199315.

[18] S. Staab, Wissensmanagement mit Ontologien und Metadaten, *Informatik Spektrum* **25**(3) (2002), 194–209. doi:10.1007/s002870200226.

[19] University of Edinburgh, Documentation, metadata, citation. https://mantra.edina.ac.uk/documentation_metadata_citation/.

[20] W3C, RDF 1.1 Concepts and Abstract Syntax. https://www.w3.org/TR/rdf11-concepts/.

[21] Dublin Core Metadata Initiative, DCMI: DCMI Metadata Terms. https://www.dublincore.org/specifications/dublin-core/dcmi-terms.

[22] Jisc, Metadata management. https://www.jisc.ac.uk/guides/metadata/management.

[23] Dublin Core Metadata Initiative, DCMI: Dublin Core™. https://www.dublincore.org/specifications/dublin-core/.

[24] W3C, Data Catalog Vocabulary (DCAT) - Version 2. https://www.w3.org/TR/vocab-dcat-2/.

[25] DataCite - International Data Citation Initiative e.V., DataCite Schema. https://schema.datacite.org/.

[26] A. Kraft, M. Razum, J. Potthoff, A. Porzel, T. Engel, F. Lange, K. van den Broek and F. Furtado, The RADAR Project—A Service for Research Data Archival and Publication, *International Journal of Geo-Information* **5**(3) (2016), 28. doi:10.3390/ijgi5030028. https://www.researchgate.net/publication/297607606_The_RADAR_Project-A_Service_for_Research_Data_Archival_and_Publication.

[27] Jisc, Controlled vocabulary. https://www.jisc.ac.uk/guides/metadata/controlled-vocabulary.

[28] Dublin Core Metadate Initiative, DCMI: Application Profile. https://www.dublincore.org/resources/glossary/application_profile/.

[29] N. Drummond and R. Shearer, The open world assumption, in: *eSI Workshop: The Closed World of Databases meets the Open World of the Semantic Web*, Vol. 15, 2006.

[30] J. Sequeda, Introduction to: Open World Assumption vs Closed World Assumption - DATAVERSITY, 2012. https://www.dataversity.net/introduction-to-open-world-assumption-vs-closed-world-assumption/.

[31] H. Knublauch and D. Kontokostas, Shapes Constraint Language (SHACL). https://www.w3.org/TR/shacl/.

[32] H. Knublauch, Form Generation using SHACL and DASH, 2020. http://datashapes.org/forms.html.

[33] K. de Smedt, D. Koureas and P. Wittenburg, FAIR Digital Objects for Science: From Data Pieces to Actionable Knowledge Units, *Publications* **8**(2) (2020), 21. doi:10.3390/publications8020021.

[34] Corporation for National Research Initiatives, Handle.Net Registry. https://www.handle.net/.

[35] W3C, The Organization Ontology. https://www.w3.org/TR/vocab-org/.

[36] Universität Stuttgart, EngMeta - Beschreibung von Forschungsdaten. https://www.izus.uni-stuttgart.de/fokus/engmeta/.

[37] Universität Stuttgart, FDM-Projekt DIPL-ING. http://www.ub.uni-stuttgart.de/dipling#.

[38] S. Shekarpour, S. Auer, A.-C.N. Ngomo, D. Gerber and C. Stadler, Keyword-Driven SPARQL Query Generation Leveraging Background Knowledge, 2011, pp. 203–210. doi:10.1109/WI-IAT.2011.70.

[39] W3C, SHACL Advanced Features. https://www.w3.org/TR/shacl-af/#rules.

[40] S. Bensberg, Sample Dataset for Search Engine Evaluation for Research Data in an RDF-based Knowledge Graph, RWTH Aachen University. doi:10.18154/RWTH-2020-09886.

[41] S. Bensberg, An Efficient Semantic Search Engine for Research Data in an RDF-based Knowledge Graph, RWTH Aachen University. doi:10.18154/RWTH-2020-09883.

[42] S. Bensberg, Search Engine Evaluation for Research Data in an RDF-based Knowledge Graph, RWTH Aachen University, 2020. doi:10.18154/RWTH-2020-09885.

[43] Universität Stuttgart, EngMeta - Beschreibung von Forschungsdaten | Informations- und Kommunikationszentrum | Universität Stuttgart, 2020. https://www.izus.uni-stuttgart.de/fokus/engmeta.

[44] Elasticsearch B.V., Query string query | Elasticsearch Reference [master] | Elastic, 2020. https://www.elastic.co/guide/en/elasticsearch/reference/master/query-dsl-query-string-query.html.

[45] dotNetRDF Project, dotNetRDF. https://www.dotnetrdf.org/.

[46] J. Nielsen, *Usability engineering*, Academic Press, Boston, 1993. ISBN 0125184050.

[47] E. Kuric, J.D. Fernández and O. Drozd, Knowledge Graph Exploration: A Usability Evaluation of Query Builders for Laypeople, in: *Semantic Systems. The Power of AI and Knowledge Graphs*, M. Acosta, P. Cudré-Mauroux and M. Maleshkova, eds, Information Systems and Applications, incl. Internet/Web, and HCI, Springer International Publishing, Cham, 2019, pp. 326–342. ISBN 978-3-030-33220-4.

[48] P. Grafkin, M. Mironov, M. Fellmann, B. Lantow, K. Sandkuhl and A.V. Smirnov, SPARQL Query Builders : Overview and Comparison, CEUR-WS, 2016. http://hj.diva-portal.org/smash/record.jsf?pid=diva2%3A1070495&dswid=6004.

[49] A. Styperek, M. Ciesielczyk, A. Szwabe and P. Misiorek, Evaluation of SPARQL-compliant semantic search user interfaces, *Vietnam Journal of Computer Science* **2**(3) (2015), 191–199. doi:10.1007/s40595-015-0044-y.

[50] S. Campinas, Graph summarisation of web data: data-driven generation of structured representations (2016). https://www.semanticscholar.org/paper/Graph-summarisation-of-web-data%3A-data-driven-of-Campinas/6a6fec6cb64c9f9aa7e3e73fd301eb7c80565ef0.

[51] R. Delbru, S. Campinas and G. Tummarello, *Searching Web Data: An Entity Retrieval and High-Performance Indexing Model*, 2012. doi:10.2139/ssrn.3198931.

[52] Linked swissbib - swissbib. http://www.swissbib.org/wiki/index.php?title=Linked_swissbib.
[53] Elasticsearch B.V., Removal of mapping types | Elasticsearch Reference [7.9] | Elastic. https://www.elastic.co/guide/en/elasticsearch/reference/current/removal-of-types.html.
[54] M. Mandalka, Meta data enrichment or annotator from Resource Description Framework (RDF) to Solr or Elasticsearch | Open Semantic Search. https://www.opensemanticsearch.org/enhancer/rdf.
[55] L. Zhang, Q. Liu, J. Zhang, H. Wang, Y. Pan and Y. Yu, Semplore: An IR Approach to Scalable Hybrid Query of Semantic Web Data, in: *The semantic web*, K. Aberer, ed., Lecture Notes in Computer Science, Springer, Berlin, 2007, pp. 652–665. ISBN 978-3-540-76298-0.
[56] C. Rocha, D. Schwabe and M.P. Aragao, A hybrid approach for searching in the semantic web, in: *13th International Conference on World Wide Web Conference*, S. Feldman, M. Uretsky, M. Najork and C. Wills, eds, ACM Press, New York, N.Y., 2005, p. 374. ISBN 158113844X. doi:10.1145/988672.988723.
[57] A. Freitas, S. O'Riáin and E. Curry, Querying and Searching Heterogeneous Knowledge Graphs in Real-time Linked Dataspaces, in: *Real-time linked dataspaces*, E. Curry, ed., SpringerOpen, Cham, Switzerland, 2020, pp. 105–124. ISBN 978-3-030-29664-3. doi:10.1007/978-3-030-29665-0_7.
[58] RWTH Aachen University, Coscine ApplicationProfiles - GitLab, 2020. https://git.rwth-aachen.de/coscine/graphs/applicationprofiles.

Appendix A. Commonly used Prefixes

For brevity definition of prefixes were omitted in the examples. The following is the comprehensive list of prefixes used:

```
@prefix coscine: <https://purl.org/coscine/terms#> .
@prefix coscineengmeta: <https://purl.org/coscine/ap/engmeta/> .
@prefix coscinesc: <https://purl.org/coscine/terms/searchextension/> .

@prefix dbpedia: <http://dbpedia.org/resource/> .

@prefix dfg: <https://www.dfg.de/dfg_profil/gremien/fachkollegien/> .
@prefix dfgCS: <https://www.dfg.de/dfg_profil/gremien/fachkollegien/liste/index.jsp?id=409#> .

@prefix dcterms: <http://purl.org/dc/terms/> .

@prefix ex: <http://example.org#> .

@prefix engmeta: <http://www.ub.uni-stuttgart.de/dipling#> .

@prefix foaf: <http://xmlns.com/foaf/0.1/> .

@prefix org: <http://www.w3.org/ns/org#> .

@prefix owl: <http://www.w3.org/2002/07/owl#> .

@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

@prefix rwth: <https://ror.org/04xfq0f34/> .

@prefix schema: <https://schema.org/> .

@prefix sh: <http://www.w3.org/ns/shacl#> .
```

Appendix B. EngMeta Metadata Schema as Application Profile

The following is an application profile based on an excerpt of the *EngMeta* metadata schema [36] created for the evaluation. A more extensive definition is available in the Coscine source repository [58].

```
<https://purl.org/coscine/ap/engmeta/>
  a sh:NodeShape ;
  sh:targetClass <https://purl.org/coscine/ap/engmeta/> ;
  sh:property coscineengmeta:creator ;
  sh:property coscineengmeta:worked ;
  sh:property coscineengmeta:title ;
  sh:property coscineengmeta:subject ;
  sh:property coscineengmeta:created ;
  sh:property coscineengmeta:version .

coscineengmeta:creator
  sh:path dcterms:creator ;
  sh:order 1 ;
  sh:minCount 1 ;
  sh:name "Creator"@en, "Autor"@de ;
  sh:class foaf:Agent .

coscineengmeta:worked
  sh:path engmeta:worked ;
  sh:order 2 ;
```

```
  sh:maxCount 1 ;
  sh:datatype xsd:boolean ;
  sh:name "Worked"@en, "Hat funktioniert"@de .

coscineengmeta:title
  sh:path dcterms:title ;
  sh:order 3 ;
  sh:minCount 1 ;
  sh:minLength 1 ;
  sh:datatype xsd:string ;
  sh:name "Title"@en, "Titel"@de .

coscineengmeta:subject
  sh:path dcterms:subject ;
  sh:order 4 ;
  sh:maxCount 1 ;
  sh:name "Subject Area"@en, "Sachgebiet"@de ;
  sh:class dfg:.

coscineengmeta:created
  sh:path dcterms:created ;
  sh:order 5 ;
  sh:minCount 1 ;
  sh:maxCount 1 ;
  sh:datatype xsd:date ;
  sh:name "Creation Date"@en, "Erstellungsdatum"@de .

coscineengmeta:version
  sh:path engmeta:version ;
  sh:order 6 ;
  sh:minCount 1 ;
  sh:maxCount 1 ;
  sh:datatype xsd:integer ;
  sh:name "Version"@en, "Version"@de .
```

Appendix C. Generated Reference Queries

For validation of the search queries all search intentions were built as reference queries. Their results were used as a reference for the evaluation of precision and recall of the mapping.

Data records which were published at the IT Center

```
SELECT ?s WHERE {
  {
    SELECT ?s {
      ?s dcterms:publisher <https://www.rwth-aachen.de/22000> .
    }
  } UNION {
    SELECT ?s {
      ?s dcterms:publisher ?person .
      ?membership a org:Membership.
      ?membership org:member ?person.
      ?membership org:organization <https://www.rwth-aachen.de/22000> .
    }
  }
}
```

Data records with version number 10

```
SELECT ?s WHERE {
  ?s engmeta:version 10 .
}
```

Data records which were created in 2007

```
SELECT ?s WHERE {
  ?s dcterms:created ?date .
  FILTER (?date >= xsd:date("2007-01-01") && ?date < xsd:date("2008-01-01"))
}
```

Data records about design patterns and with version number less than 5

```
SELECT ?s WHERE {
  {
    SELECT ?s {
      ?s dcterms:title ?title .
      ?title bif:contains "'design patterns'" .
    }
  } UNION {
    SELECT ?s {
      ?s dcterms:abstract ?abstract .
      ?abstract bif:contains "'design patterns'" .
    }
  }
  ?s engmeta:version ?version .
  FILTER (?version < 5) .
}
```

Data records about computer science

```
SELECT ?s WHERE {
  {
    SELECT ?s {
      ?s dcterms:abstract ?abstract .
      ?abstract bif:contains "'computer science'" .
    }
  } UNION {
    SELECT ?s {
      ?s dcterms:subject ?subject .
      ?class rdfs:subClassOf* ?superclass .
      ?subject a ?class .
      FILTER (STR(?superclass) = "http://www.dfg.de/.../liste/index.jsp?id=409") .
    }
  }
}
```

The newest data record of resource 2

```
SELECT ?s WHERE {
  ?s engmeta:version ?version .
  ?s coscineprojectstructure:isFileOf |
     <https://purl.org/coscine/vocabularies/resource#resource2> .
  {
    SELECT ?newestVersion WHERE {
      ?file engmeta:version ?newestVersion .
```

```
      ?file coscineprojectstructure:isFileOf
          <https://purl.org/coscine/vocabularies/resource#resource2> .
    } ORDER BY DESC(?newestVersion) LIMIT 1
  }.
  FILTER( ?version = ?newestVersion).
}
```

Data records which are experiments or simulations and not analysis

```
SELECT ?s WHERE {
  ?s engmeta:mode ?mode .
  FILTER ((?mode = <https://purl.org/coscine/terms/mode#experiment> || ?mode =
    <https://purl.org/coscine/terms/mode#simulation>) && ?mode !=
    <https://purl.org/coscine/terms/mode#analysis> ).
}
```

Data records about non computability of the human consciousness

```
SELECT ?s WHERE {
  ?s dcterms:abstract ?abstract .
  ?abstract bif:contains "'human consciousness' AND non-algorithmic".
}
```

Data records which are available since 03.07.2020

```
SELECT ?s WHERE {
  ?s dcterms:available "2020-07-03+02:00"^^xsd:date .
}
```

Data records about political left

```
SELECT ?s WHERE {
  ?s dcterms:abstract ?abstract .
  ?abstract bif:contains "left".
}
```

Data records which contain a variable with the value 2 meters

```
SELECT ?s WHERE {
  {
    ?s engmeta:controlledVariable <https://purl.org/coscine/terms/variable#variable1>.
  } UNION {
    ?s engmeta:measuredVariable <https://purl.org/coscine/terms/variable#variable1>.
  }
}
```

Data records about object-oriented software which are published before 2015

```
SELECT ?s WHERE {
  ?s dcterms:issued ?date .
  ?s dcterms:abstract ?abstract .
  ?abstract bif:contains "'object-oriented software'".
  FILTER (?date <= xsd:date("2015-01-01")) .
}
```

Data records which are published by Thomas Pynchon, created in 2016 and about object-oriented software

```
SELECT ?s WHERE {
  ?s dcterms:publisher <http://dbpedia.org/resource/Thomas_Pynchon> .
  ?s dcterms:created ?date .
  ?s dcterms:abstract ?abstract .
  ?abstract bif:contains "'object-oriented software'" .
  FILTER (?date >= xsd:date("2016-01-01") && ?date < xsd:date("2017-01-01")).
}
```

Appendix D. Search Queries for the Mapped Objects in Elasticsearch

The search intentions were modeled as ES search queries using the *Query string query* syntax [44].

Data records which were published at the IT Center

```
publisher:"IT Center"
```

Data records with version number 10

```
version:10
```

Data records which were created in 2007

```
date_created:[2007-01-01 TO 2007-12-31]
```

Data records about design patterns and with version number less than 5

```
"design patterns" AND version:<5
```

Data records about computer science

```
"computer science"
```

The newest data record of resource 2

```
is_file_of:"resource 2" AND is_newest_version:true
```

Data records which are experiments or simulations and not analysis

```
mode:((experiment OR simulation) AND NOT analysis)
```

Data records about non computability of the human consciousness

```
abstract:("human consciousness" AND non-algorithmic)
```

Data records which are available since 03.07.2020

```
date_available:2020-07-03
```

Data records about political left

```
left
```

Data records which contain a variable with the value 2 meters

```
controlled\_variables:"2 meter" measured_variables:"2 meter"
```

Data records about object-oriented software which are published before 2015

```
abstract:"object-oriented software" AND date_issued:* TO 2015-01-01
```

Data records which are published by Thomas Pynchon, created in 2016 and about object-oriented software

```
publisher:"Thomas Pynchon" AND date_created_year:2016 AND
abstract:"object-oriented software"
```

Appendix E. Confusion Matrices for Search Intentions

The following shows confusion matrices of the *Query string query* syntax of ES.

Published at IT Center

Received \ Relevant	+	-
+	2	0
-	0	33

Version 10

Received \ Relevant	+	-
+	1	0
-	0	34

Created in 2007

Received \ Relevant	+	-
+	1	0
-	0	34

Design Patterns Version < 5

Received \ Relevant	+	-
+	1	0
-	0	34

Computer Science

Received \ Relevant	+	-
+	4	0
-	0	31

Newest Version Res. 2

Received \ Relevant	+	-
+	1	0
-	0	34

Experi./Simu. not Anal.

Received \ Relevant	+	-
+	22	0
-	0	13

Non Comput. Human Consci.

Received \ Relevant	+	-
+	2	0
-	0	33

Av. s. 03.07.20

Received \ Relevant	+	-
+	10	0
-	0	25

Left (political)

Received \ Relevant	+	-
+	1	2
-	0	32

2 Meters

Received \ Relevant	+	-
+	5	0
-	0	30

Pub. < 2015 Oo. Softw.

Received \ Relevant	+	-
+	0	0
-	0	35

Pub. T. Pynt. Created 2016 Oo. Softw.

Received \ Relevant	+	-
+	1	0
-	0	34

In Total

Received \ Relevant	+	-
+	51	2
-	0	402