Bachelor's Thesis

Integrating component metadata into the linked data world: a case study

Author
Nino Meisinger
nino.meisinger@student.uni-tuebingen.de

Supervisor
Prof. Erhard Hinrichs
erhard.hinrichs@uni-tuebingen.de
Dr. Thorsten Trippel
thorsten.trippel@uni-tuebingen.de

A thesis submitted in partial fulfillment
of the requirements for the degree of

Bachelor of Arts
in
International Studies in Computational Linguistics

Seminar für Sprachwissenschaft
Eberhard Karls Universität Tübingen

20th September 2021

Antiplagiatserklärung

Ich erkläre hiermit,

dass ich die vorliegende Arbeit selbständig verfasst habe,

dass ich keine anderen als die angegebenen Hilfsmittel und Quellen benutzt habe,

dass ich alle wörtlich oder sinngemäß aus anderen Werken übernommenen Aussagen als solche gekennzeichnet habe,

dass die Arbeit weder vollständig noch in wesentlichen Teilen Gegenstand eines anderen Prüfungsverfahrens gewesen ist,

dass ich die Arbeit weder vollständig noch in wesentlichen Teilen bereits veröffentlicht habe,

dass das in Dateiform eingereichte Exemplar mit dem eingereichten gebundenen Exemplar übereinstimmt.

Tübingen, den 20. 09. 2021 N. Meisinger

(Nino Meisinger)

Abstract

Due to the increasing popularity of Linked Data in the Linguistic field, this thesis explores the current state of the Linked Data world and uses the findings to develop a new way of transforming Component Metadata Infrastructure (CMDI) instances into JSON-LD, a Linked Data format.
A system is developed, mapping re-occurring metadata elements onto Schema.org vocabulary, a well-known ontology, and building a general representation of CMDI in a JSON-LD format. Furthermore, the benefits of authority control are investigated.
The resulting tool produces high-quality Linked data by making use of SPARQL, a query language. New ways of interacting with the resulting data are shown, particularly how linking entities to authority files enhances the quality of the metadata. It enables users to access other Linked Data databases, which opens up new opportunities in how to extract information from a single metadata record.

Contents

List of Tables

List of Figures

1 Introduction

Linked Data are increasingly becoming more popular, providing new ways to work with data and metadata alike. However, large parts of linguistic metadata, such as from the *Common Language Resources and Technology Infrastructure* (CLARIN), a research infrastructure providing access to a wide variety of language resources and tools, still uses a traditional representation of metadata. Their standard, the *Component Metadata Infrastructure* (CMDI) (Broeder et al., 2021; ISO 24622-2:2019) is a complex system enabling the creation of interconnected metadata schemata, expressed in *Extensible Markup Language* (XML) (Bray et al., 2000).
This thesis explores a new way of transforming CMDI to a Linked Open Data format. The aim is to develop an easy-to-use, yet powerful system to map existing CMDI schemata onto established vocabulary from the Linked Data world. This would provide a well-documented interface that users unfamiliar with the CMDI structure can access and help future integration of CMDI into the Linked Data world.

Section 2 will introduce the Component Metadata Infrastructure and technologies for creating Linked Data, such as the *Resource Description Framework* (RDF) and one of its serialization formats JSON-LD. Furthermore, the current state of the Linked Data world is explored by providing a short overview of its history and its current use cases. Section 3 will proceed to cover related work regarding the transformation of CMDI to Linked Data.
Finally, section 4 will showcase the tool developed for transforming CMDI to Linked Data, providing an overview of the project's layout and how it works. Section 5 will show how the system was used, highlighting possible ways to use the resulting data. This will follow into a discussion on how to further improve the underlying CMDI data and the tool.

2 Background

2.1 Component Metadata Infrastructure

Previous experience in metadata modeling showed, that a single generic schema is not specific enough to be applied to a wide range of various resources or tools. To overcome this problem the Component Metadata Infrastructure was created (Broeder et al., 2021; ISO 24622-2:2019). CMDI is a modular framework that lets users define their own metadata schemata, using descriptions and names deemed appropriate for the given resource at hand. Yet, the schemata themselves are still part of the overlying CMDI framework, to achieve semantic interoperability, i.e. to establish a shared meaning, as well as interpretability across the different schemata.
To achieve this, the CMDI framework allows the creation of new metadata schemata using a modular approach. The most basic building blocks are so-called "components". Components usually describe one specific aspect of a given resource (e.g. its creators) and consist of various elements, called "concepts" (e.g. name of a creator, their e-mail address, etc.). A given metadata schema, called "profile", under the CMDI framework is a collection of those components. To avoid redundancy, as well as to increase semantic interoperability, metadata creators are encouraged to reuse existing components when creating their own profiles. This is done by providing two registries: The CLARIN concept registry (CCR) (Schuurman et al., 2016) and the Component Registry (Broeder et al., 2012). The former provides a definition for all concepts, whereas the latter provides a definition for components and profiles. Both platforms are accessible over the web. Since CMDI metadata instances, which will be referred to as *CMDIs* in this thesis, are defined as XML documents, one can also download an XML Schema for a given profile to use it for validation purposes.
Concepts are identified by a unique HTTP URL leading to their CCR entry, whereas profiles and components are identified with a unique identifier. Said identifier is a string to identify profiles or components, instead of a URL. As such it cannot be used to access the corresponding Component Registry entry.

As CMDIs are usually hosted across various CLARIN data centers, the Virtual Language Observatory (VLO) (Van Uytvanck et al., 2012) was established as a specialized search application

using the CMDIs as its source. The VLO provides a web interface to explore different CMDI metadata, no matter where they are hosted, as well as providing a variety of search filters (name, description, creators, etc.), called "facets". Those facets are a good example of showcasing the semantic interoperability of the framework. Mapping each different CMDI schema to a specific facet space manually would be too work-intensive, considering that over 180 public profiles are currently in use. Instead, the VLO maps different concepts declared in the CCR onto facets or group fields with similar semantics so they can be explored together. As long as a profile uses established concepts or components, it can be automatically mapped to different facets. For rare cases however, the system also provides a way to create mappings by hand.

While this system works in theory quite well, Trippel and Zinn (2016) highlighted a few problems concerning its semantic interoperability: Many metadata creators do not make use of the reusability offered by the CMDI framework. The CCR contains duplicated concept definitions, whereas the Component Registry contains duplicated components and entire profiles that describe the same resource type. To describe the name of a resource the `ResourceName` or `ResourceTitle` concepts are often used interchangeably for example. This leads to the question of whether distinguishing between both of them serves a purpose.
Not all profiles consistently utilize the CCR and Component Registry for all elements in the schema. This leads to many elements not having a unique identifier at all. This can be partly explained by the fact, that the CRR got introduced in 2016 (Schuurman et al., 2016), several years after the introduction of the CMDI framework and partly, because unique identifiers are not enforced. The Component Registry, for example, allows the definition of in-line components. These are sub-components without an extra Component Registry entry. As such, they do not necessarily receive an identifier. However, it is also possible to assign an external identifier originating from other metadata vocabularies to components.
Another issue is the lack of controlled vocabulary used in many profiles (Trippel and Zinn, 2016). Using controlled vocabulary is beneficial for automation tasks, such as statistical evaluations of the metadata at hand (e.g. "How many Spanish text corpora do exist?") and benefits the user experience when searching for resources online. McCutcheon (2009) for example argued that both, keywords and controlled vocabulary, complement each other to improve information retrieval.
For many subjects, a list of established terminology already exists that could be used to enhance controlled vocabulary usage inside CMDIs. Languages, for example, are often described with the ISO 639-3 code list. The list encodes each language as three characters (e.g. "English" becomes "eng", "German" "deu", etc.).
For some entities, such as persons or organizations, authority control can be used instead. Different organizations like the Online Computer Library Center (OCLC) or the German National Library (DNB) maintain authority files that can be freely accessed and used to assign unique identifiers to these entities inside a CMDI.

2.2 The Linked Data World

2.2.1 Representation Format of Linked Data: The Resource Description Framework

In theory, Linked Data can be found in a variety of different formats. Berners-Lee (2006) specified four different rules that should be fulfilled, as well as giving a rough definition of the term:

1. *Unique resource identifiers* (URIs) should be used as names for things.
2. Said URIs should be HTTP URIs, so people are able to look up those names.
3. Using Linked Data standards, URIs should provide useful information when someone looks them up.
4. URIs should include links to other URIs, so it becomes possible to discover more things.

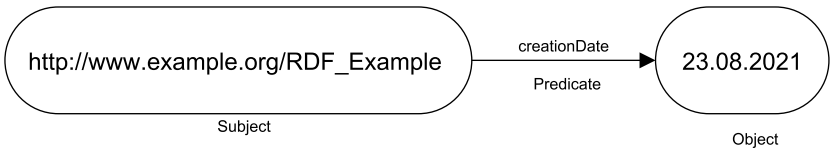
Linked Data claims to focus on meaningful connections to other Linked Data. As such, these rules stress, that one's data should conform to the standard and link to other data using URIs.

```xml
<?xml version="1.0"?>
<rdf:RDF
xmlns:rdf="https://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:scm="https://www.example.org/rdf#">
  <rdf:Description rdf:about="http://www.example.org/RDF_Example">
    <scm:creationDate>23.08.2021</scm:creationDate>
  </rdf:Description>
</rdf:RDF>

```

(a) Triple in RDF/XML

```
@prefix scm: <https://www.example.org/rdf#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<http://www.example.org/RDF_Example> scm:creationDate
    "23.08.2021"^^xsd:string .

```

(b) Triple in Turtle



(c) Graph visualization of triple

Figure 1: Triple representation in RDF/XML, Turtle, and the corresponding graph visualization

Furthermore, Berners-Lee (2010) added a five-star system to rate the quality of Linked Open Data (LOD) – Linked Data that is freely available to others, following open source principles:

1. The data should be available under an open license on the Web (★)
2. The data should be available as machine-readable structured data (★ ★)
3. The data should use a non-proprietary format (★ ★ ★)
4. Including the above, the data should use open standards for Linked Data defined by the W3C, so other people can link to the data (★ ★ ★ ★)
5. Including the above, one should link one's data to other people's data to provide more context (★ ★ ★ ★ ★)

The most common format found for Linked Data is the "Resource Description Framework" (RDF), a metadata model to describe resources on the web in a graph-based structure. Work on RDF began in 1997 (Lassila, 1997), leading to a W3C recommendation in 1999 (Brickley et al., 1999) and resulted in the first specification of it in 2004 (Beckett and McBride, 2004).
Due to the complexity of the initial serialization format RDF/XML, additional formats to express RDF are available nowadays, such as Turtle, N-Triples, or JSON-LD.
The core structure of RDF is a set of so-called "triples", which are modeled after simple statements in natural language. Each triple has three components:
A *subject* denoting a resource, a *predicate* describing different aspects of the resource and connecting the *subject* to an *object*.
Such a triple can be seen in Figure 1. Figure 1a depicts the RDF/XML notation with its corresponding graph visualization in Figure 1c. The triple stands for the statement, that the website

"http://www.example.org/RDF_Example" was created on the 23rd of August in 2021. This translates into a simple graph containing two nodes, the subject (the website URL) and the object (the creation date). Both are connected by an edge, which is the predicate and denotes the relationship between the two of them (that the object's value is the creation date).
The same triple can be seen in Turtle in Figure 1b. The syntax to express a triple follows a simplified notation of the form: `subject predicate object .`
The `.` states that the triple is finished. In other words, the statement has been made.

Real RDF documents tend to be more complex in nature, being able to represent complex graph structures with more than one triple. One would for example be able to extend the example, by adding information about the web site's title or list of authors. Authors could be a nested set of triples, describing various aspects of an author, such as their given name, family name, date of birth, or similar.
To establish links to other data, the subject or object should be identified by a URI. However, it is allowed for objects to take other values and for subjects to be blank nodes, called "anonymous resources". The latter can be useful in describing multi-component structures, e.g. adding a list of authors as described above.
The predicate `creationDate` has the namespace prefix `scm`, meaning it evaluates to a URI as well. The namespace in question leads to one of the other core components of RDF: ontologies.

Ontologies are a description of the concepts and their possible relationships in an RDF document. They are usually expressed in RDF schemata – RDF documents defining classes of objects and how they relate to each other. In our example, the RDF schema behind the namespace `scm` would define that one possible relation for the given subject is `creationDate`. Leading to an *object* of a date format of the type *dd.mm.yyyy*. If we would add the list of authors, it could further define that one or multiple objects of type `Person` are a valid relation for our *subject*.
Berners-Lee et al. (2001) recognizes one problem that comes with the use of ontologies: A lack of semantic interoperability across documents using different ontologies. He gives the example of one document using *zip codes* to identify addresses and the other *postal codes*. In our example, one ontology might use `givenName` and `familyName` to describe persons and another one a simple concatenation of both under the value `name`. In Berners-Lee et al.'s case, they argue that a program parsing Linked Data from different resources should automatically be capable of recognizing that both concepts are related and convert them to the preferred one accordingly.

RDF data can be queried with SPARQL (Eric Prud'hommeaux, 2008), which is a query language specifically developed for it. Queries are mainly made by writing triples patterns in Turtle format.

2.2.2 The History of Linked Data

The term "Linked Data" itself was first coined in 2006 by Tim Berners-Lee, director of the World Wide Web Consortium (W3C) and inventor of the World Wide Web, in relation to the Semantic Web project (Berners-Lee, 2006) – more than five years after the introduction of RDF itself. The Semantic Web was introduced in 1999, when Berners-Lee et al. published an article, presenting it as the next evolution of the Web. In 2001, Berners-Lee et al. (2001) explained it in more detail.
The World Wide Web as it is today is mostly designed so humans can understand it. Just by parsing an HTML document, computers can, at most, understand its syntactic structure. They fail at understanding the semantic meaning of a website. The Semantic Web aims to change that by adding an additional layer on top of the normal Web. This is achieved by integrating metadata expressed in XML into a web page. Said metadata annotations are realized through RDF and provide a machine-readable interface of the page, as well as linking it to other resources on the Web.
Berners-Lee et al. (2001) gives the example of software being able to automatically extract the business hours of a doctor's clinic, without relying on keywords inside the HTML document and "without needing artificial intelligence on the scale of 2001's Hal". Other things would be possible as well, such as encoding the information of an author of a news article, linking to more information such as their date of birth, current employee, etc. This could lead to a World

Wide Web quite different from the way it is nowadays, e.g. instead of having social media save information about your friend list on their servers (Target, 2018), they could retrieve said information from your own website via Semantic Web technologies – leading towards a more decentralized internet as a whole.

Although Linked Data is very much alive nowadays, this proposed idea of the Semantic Web struggles to come to fruition and did not gain mainstream adoption (Swartz, 2013; Cagle, 2016; Verborgh and Vander Sande, 2020). To understand, how the current state of the Linked Data world came to be, it is useful to take a look at some of the core criticisms directed towards the Semantic Web:
As early as 2001, data activist and journalist Cory Doctorow released an often-cited essay defining seven key problems regarding the Semantic Web and metadata (Doctorow, 2001).
Most problems can be summed up as underestimating the naivety or unwillingness of people to care about metadata. For one, he expects people to lie about correctly annotating their website if they assume to gain a competitive edge from it. Pringle et al. (1998) for example, realized, that website creators were trying to gain a more favorable search ranking, by adding fake keywords to the head of their HTML documents, or by adding an invisible font to their pages. For another, he expects most website creators to simply not care about adding high-quality metadata annotations.
Ontologies are often criticized as being flawed by design. Doctorow argues that they assume, concepts and their relationships can be categorized in a "correct" way, which does not mirror the real world. As such, it is unlikely that single ontologies can be used on a global scale.

It is often argued, that the Semantic Web lacks incentives for website creators to produce Linked Data. Linked Data benefits from other published Linked Data. If there is none, there is no reason for others to create it. As a result, there is also a lack of software that makes use of it. This is often referred to as a "chicken-and-egg problem" (Neish, 2015; Mika, 2017).
Alongside this, many people consider the Semantic Web standards as too long and difficult to understand (Cagle, 2016). Swartz, one of the creators of the RDF-based web feed format RSS, criticizes the overly academic mindset behind the project. Unlike the Web, which was built with a "let's just build something that works" attitude, too much time and effort were spent on academic research and the creation of standards and drafts for ontologies (Swartz, 2013). He emphasizes, that standards should be written after the proposed technology works already, not before. As a result, potential users are overwhelmed by the high entry barrier, having to understand the different technologies (various RDF serialization formats, ontologies, SPARQL, etc.) first, before being able to create their own data.

Finally, advances in technology have rendered the idea of the Semantic Web less useful. RESTful APIs, an architectural style for application programming interfaces (APIs) that uses HTTP requests to access and use data, have become more widespread and popular leading to a similar, although weakened result as the Semantic Web (Target, 2018). Website data can be parsed, but instead of being able to access the data freely, one has to find and sign up for each API they want to use. For many companies, however, it comes with the advantage of being in control of how, and which, data can get accessed.
Cabeda (2017) on the other hand, argues that advancements in machine learning made the idea of the Semantic Web obsolete. Instead of making data machine-readable, the idea is to make machines understand natural language. In that regard fast progress is made, for example, a system extracting knowledge bases from text improved its F1 score by 20% over the course of a year (Getman et al., 2017). Experiments like the one from Mori et al. (2012) support this idea. They successfully tried, whether recipe text processing using data found online is possible. Modern named entity recognition systems achieve over 90% accuracy (Wang et al., 2020), meaning persons (such as author names from a news article), can be recognized with machine learning as well.

Despite all of that, work on the Semantic Web and its technologies have continued, leading to an increase of Linked Data on the Web.
However, the Linked Data found on the Web nowadays, differs from the original idea of the

Semantic Web. Instead of having lots of self-hosted, decentralized data on basically every web page, Linked Data was found out to be popular for specific applications and domains (Neish, 2015). This is mainly the case, because most of the problems described above, can be overcome if single organizations or dedicated communities of users work on publishing Linked Data, instead of individual uncoordinated efforts.
In 2007, DBpedia was released, a Linked Data database with information extracted from Wikipedia (Auer et al., 2007). Since then, the project has been a huge success, with a steadily increasing user count. Whereas in 2014, the average amount per request to the DBpedia service was 3.4 million, this number increased to 7.7 million by the end of 2016 (van Kleef, 2018). In a similar vein, the Wikimedia Foundation started the Wikidata project in 2012 (Vrandečić, 2012). Unlike DBpedia which strictly extracts content from Wikipedia, Wikidata is a collaboratively edited knowledge graph, meaning, anyone can add data to it. As of the 24th January 2021, Wikidata itself reports 22 923 active users[1] and over a hundred million queries a month for their SPARQL endpoint in 2018 (Malyshev et al., 2018).
In the academic field, as well as on a government level, Linked Data too, have become popular (Neish, 2015; Malyshev et al., 2018). One example of this would be the Atlas of Living Australia (ALA), created in 2010 (Belbin et al., 2021). The ALA is a biodiversity database, established under the Australian Government's National Collaborative Research Infrastructure Strategy.
Another early adopter of Linked Data are libraries. The German National Library[2] for example started to use Linked Data in 2010 already. Others are following suit. The usage enables them to establish links to other online services, as well as to retrieve information from them (Hallo et al., 2016).

With this, the chicken-and-egg problem is getting solved. Organizations usually already have a reason in mind, when deciding to publish Linked Data (as is the case for libraries for example). However, looking at the high usage numbers for DBPedia suggests, that others found reasons to use Linked Data as well.
Companies like Amazon and Apple, for example, use Wikidata for their voice recognition systems Alexa and Siri (Malyshev et al., 2018), although to what extent is not known. In a similar vein, Haase et al. (2017) made use of Wikidata to develop an application for Amazon's Alexa, enabling users to retrieve information from it. Researchers and journalists can make use of Wikidata for automated information extraction, as has been the case in a report showing how women have been covered in coronavirus media coverage. For this, the gender and occupation of over 50 000 people were identified using the service (Jones, 2020).
Applications, such as the visualization tool Scholia[3], have been developed using Wikidata as well.
ALA on the other hand is used by the government, higher education, non-government organizations, and more.

The usage of Linked Data in voice recognition systems points towards a larger trend: Linked Data as a way to improve machine learning applications. Although Cabeda (2017) argues that machine learning rendered the need for Linked Data less important, research points in another direction: Creating hybrid systems using both. Mountantonakis and Tzitzikas (2017) for example found in a study from 2017, that Linked Data can help in the automatic creation of datasets for machine learning training. Features retrieved from the Linked Data, helped increasing the accuracy of the models, which could become beneficial in the future, considering that researchers tend to spent 50%-80% of their time preparing features by hand (Mountantonakis and Tzitzikas, 2017).
Thus, it can be seen that Linked Data can help machine learning in two ways. Help, gathering data to build new machine learning models, as well as enhancing AI-style applications by letting them make use of the structured content provided by it.

[1] `https://www.wikidata.org/wiki/Wikidata:Statistics`, Last accessed: 25.08.2021
[2] `https://www.dnb.de/EN/Professionell/Metadatendienste/Datenbezug/LDS/lds_node.html#doc328464bodyText1`, Last accessed: 25.08.2021
[3] `https://scholia.toolforge.org`, Last accessed: 25.08.2021

As mentioned, Doctorow (2001) thinks that ontologies are a flawed concept, unable to ever work on a global scale. However, in recent years, not only are website creators starting to annotate their HTML data with metadata, they largely use the same ontologies and provide them in a standard-conform way. Individually, there would be little incentive to provide metadata markup of one's website. Due to the increasing power of social media, as well as Google and other search engines, this is not the case anymore.
Facebook uses the Open Graph Protocol (OGP) ("The Open Graph protocol", 2021). Using RDFa, a subtype of RDF for the annotation of HTML pages, website creators are encouraged to implement OGP annotations, so Facebook can parse them to display their website within the platform. Since social media increases website traffic (Dolega et al., 2021), website creators are incentivized to implement OGP, which lead to widespread adoption of it. As providing fake metadata annotations would lead to faulty representation of a site within Facebook, the data tends to be of good quality. Good enough for Google to make use of it for their search technologies (Neish, 2015).
Furthermore, Google, Yahoo, and Bing created Schema.org in 2011 (Guha et al., 2016). A single ontology covering a wide range of different concepts and topics. Just like OGP, Schema.org gained widespread adoption, since website creators hope to gain a better page ranking using it. The variety of different concepts and high adoption rate made the ontology suitable for the conversion of CMDI to Linked Data. Thus the following section will cover the ontology in more detail.

2.2.3 Schema.org and JSON-LD

The major search engines have tried to use Linked Data to improve search results for several years prior to Schema.org (Guha et al., 2016). However, they were presented with various problems. For one, each engine used to implement its own recommendations on how to annotate web pages. This was coupled with the problem, that the existing solutions did not scale well for an increasing amount of topics. As a result, website creators often refused to add any metadata or added incorrectly formatted one.
Thus the idea of Schema.org was born, a single ontology supported by all major search engines, with good scalability to cover a wide range of topics. Its schemata are extended and modified in accordance with the needs of its community.
The project is considered a success (Guha et al., 2016). Among 10 billion sampled web pages, Guha et al. (2016) found that in 2016 31.3% are using Schema.org markup, an increase of 9.3% from 2015. The first use case was for Google's "Rich Snippets" – advanced text excerpts displayed in the search results of Google, such as ratings (for companies, products, etc.), the average cooking time for a recipe, and more. Followed by this, Schema.org annotations are used as a data source for Google's "Knowledge Graph", used by Google to display infoboxes on search results.
Due to its success, the metadata was quickly adopted for use in other technologies. E-mails nowadays often contain Schema.org vocabulary, which is parsed by e-mail clients to add entries to the user's calendar for example. Microsoft's virtual assistant Cortana and Apple's Siri, have been shown to interpret the data as well.

Officially supported are annotations using RDFa, Microdata (an HTML5 standard to nest metadata inside web pages), and JSON-LD, with the latter being the recommended format by Google as of the writing of this thesis[4].

JSON-LD (Sporny et al., 2014) is a JSON-based serialization method for Linked Data providing full compatibility to RDF. Due to the growing complexity of RDF, a big focus when developing JSON-LD was to make it more accessible. One of its co-inventors, Manu Sporny, expressed that he was trying to avoid any mention of RDF in its specification (Sporny, 2014) and design it around the needs of programmers, who were already using JSON in their daily life. As such, it differs from the RDF serialization methods seen so far.

[4] https://developers.google.com/search/docs/advanced/structured-data/intro-structured-data#markup-formats-and-placement, Last Accessed: 08.09.2021

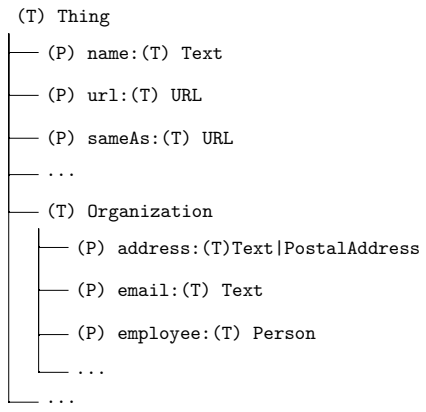
```
{
  "@context": "https://www.example.org/rdf#",
  "@id":"http://www.example.org/RDF_Example",
  "creationDate": "23.08.2021"
}

```

Figure 2: Triple representation in JSON-LD

At its core, JSON-LD is still based around RDF triples. Figure 2 shows one way to write the basic triple from Figure 1a in a JSON-LD format.
Any key value of a JSON object not starting with a `@` character, will be interpreted as a `predicate`. If they start with a `@` character, they denote special keywords. Some of the more commonly used ones are:

- `@context`: The context of a JSON-LD file. In general, the context allows one to map terms to any *Internationalized Resource Identifier* (IRI). IRIs are a superset of URIs. Unlike URIs, they are not limited to US-ASCII characters and are supposed to replace URIs. This can be used to define a structure similar to an RDF schema. In most cases, the context only contains the ontologies in use, as is the case in Figure 2.
  If no context is specified, all IRIs need to be written out.
- `@vocab`: This keyword can be used to specifically declare the default vocabulary from which all terms are derived and can be useful if more than one ontology is used in the context. In Figure 2's case, only one ontology is given, which is automatically used as default vocabulary.
- `@id`: The id keyword is entirely optional. It maps the current JSON object to an IRI. If as subject's `@id` is not defined, it will be a blank node.
- `@type`: To specify the type of a graph's node value, the `@type` is used. It serves as a shorthand to map to the RDF syntax type description and describes that a graph node is an instance of a class.
- `@language`: If declared in the context, the `@language` keyword will set the default language for all graph nodes of the said concept. It can also be used for individual graph nodes.
- `@value`: This expresses the value of a graph node. If no other keyword is used to describe the node in more detail (e.g., its language), it does not need to be specified explicitly.

Schemata defined by Schema.org consist of two key building blocks: types and properties. Each type has a variety of properties describing it. To illustrate this with an example, Figure 3 shows, how one could describe the University of Tübingen with Schema.org.
Figure 3a depicts the structure as described in Schema.org. Every type originates from the base type "`Thing`". Every child type inherits the properties from its parent. In case of `Organisation`, this means it can be described using properties, such as `name` or `url`, inherited from `Thing`.
Furthermore, each property is expected to take a value of a specific type. Types will be categorized as simple and complex in this thesis.
Simple types are string values. They can be required to follow a specific format, such as a URL however.
Complex types are realized as new JSON objects when written in JSON-LD. As such, they are normally defined as their own schema on Schema.org. Examples would be `Organization` or `PostalAddress` in Figure 3a.
Not every property needs to be filled with a value. As such the resulting JSON-LD structure can use whatever properties deemed appropriate to describe the University of Tübingen. In the example given in Figure 3b those are the properties `name, url` and `address`.
Although in this example, `url` and `@id` share the same value, this is not always the case. `url`



(a) Vocabulary description of Schema.org, where (T) stands for "Type" and (P) for "Property"

```
{
  "@context": "https://schema.org",
  "@type": "Organization",
  "@id": "https://uni-tuebingen.de/",
    "name": {
      "@language": "en",
      "@value": "University of Tuebingen"
    },
    "url": "https://uni-tuebingen.de/",
    "address": "Geschwister-Scholl-Platz,
                72074 Tuebingen"
}
```

(b) The University of Tübingen described in JSON-LD with Schema.org vocabulary

Figure 3: Translating Schema.org vocabulary into a JSON-LD representation

should lead to a page that can be visited, whereas `@id`'s should be unique, which is often achieved using URI fragments (optional fragments in an URL, introduced with #).

3 Related Work

In the field of Linguistics, there has been a push in recent years, to connect language resources using Linked Data. McCrae et al. (2016) are working on a Linguistic Linked Open Data cloud for example. For this, they connect various resources, described with different ontologies, together. Hoppermann et al. (2011) on the other hand showed, how enriching metadata with Linked Data by adding authority control will help improve its quality. Even in metadata collections stemming from a single institution, its name is often spelled in various ways. That makes it difficult to locate all resources originating from it with simple queries. By providing a link to authority files, however, this can be changed. The DNB for example holds a high quantity of data for organizations, persons, and more. That way, all publications from a given person can be extracted from a single metadata instance for example.

In recent years, there have been various attempts to convert CMDI metadata into a Linked Data format as well. In fact, many profiles are already getting converted to MARC21 (Library of Congress, 2021) and Dublin Core ("Dublin Core Metadata Initiative", 2021). The former is used for bibliographic data and even though it is not part of the Linked Data family, various efforts have been made to convert it into RDF (Kumar et al., 2013); (Pazooki and Keshavarzian, 2020). Dublin Core on the other hand was designed on the basis of the Semantic Web
However, each CMDI provider is required to implement their own transformation solution. Zinn et al. (2016) noted that a general mapping would be too difficult in its implementation. Both formats come with information loss, as they are not capable to represent CMDI metadata in its entirety.

McCrae and Cimiano (2015) build an aggregator consisting of Linked Data harvested from multiple resources, including CMDI metadata. Since only Dublin Core exports get harvested, the data is lacking in quality due to the aforementioned information loss.

Durco and Windhouwer (2014) started the CMDI2RDF project. They devised a model, to convert CMD profiles to RDF without loss in information.
They built an RDF schema, supporting the CMD meta model. This means the basic building blocks, i.e. components, concepts, and profiles, as well as its basic principles, such as the nesting

of components is supported by it. Thus, CMDI values can be mapped to RDF.
In 2017, this project was further developed and realized (Windhouwer et al., 2017). The VLO facet mapping (Van Uytvanck et al., 2012) was adopted to create better linking to other Linguistic Linked Open Data datasets. A new predicate was introduced to the schema, so facets can be directly added to the RDF files. The project itself can be accessed with a SPARQL endpoint. At the time of writing, however, the RDF schemata are not yet integrated into the Component Registry and CCR as originally planned.

Trippel and Zinn (2016) introduce another possibility to include the CMDI framework into the Linked Data world. Since the framework makes use of URIs inside the CCR and Component Registry, it should be possible to map the existing vocabulary to established Linked Data vocabulary – which is explored in this thesis.

4 Transforming CMDI to JSON-LD

4.1 System Overview

The aim of this thesis has been to explore an alternative way of mapping all of the CMD record collection to a Linked Open Data format. Whereas CMDI2RDF does provide a full transformation of CMDI to RDF, it is also entirely made up of its own vocabulary, leading to a higher entry barrier to new users, as well as making it more difficult to integrate into existing Linked Data. Previous projects, such as mapping CMDI to DC, on the other hand, came with a substantial loss in information.

As such the *CMDI2JSONLD* tool[5] has been developed. It is a Java command-line application that can convert CMDI data to JSON-LD using pre-defined mappings to Schema.org schemata, as well as using the existing Component Registry and CCR.
A user can decide whether to transform a single CMDI file or an entire directory, as well as where to save the output. Alternatively, it is also possible to enrich the input data with authority files first.
At its core, the system is a single Java class, interpreting the command line arguments and using XSLT 3.0 stylesheets (a language for transforming XML documents) for the actual transformation. The benefit of this system is, that it will be easy to migrate stylesheets to different applications.
For each input CMDI, the tool first checks if the corresponding profile schema is available. If not, it will be downloaded and saved for future use. Once done, the tool will call the XSLT stylesheets.
The mapping system is expressed in XML and can be freely edited by the user.

4.2 Enriching CMDI Data with Authoritative IDs

As mentioned by Trippel and Zinn (2016), most CMDI data lack authority control. Due to that, the Center for Sustainability of Linguistic Data (NaLiDa) at the University of Tübingen introduced the `AuthoritativeIDs` component for their CMD profiles. These are profiles marked in the Component Registry as being created by the user NaLiDa. Said component is applied to organizations and persons in their profiles to add a variety of different authoritative IDs, such as from viaf.org, a service hosted by OCLC, and the DNB. Each entity can be correctly identified that way.

For Linked Data, the IDs can be used to access other Linked Data. As such, it should become possible to perform complex queries identifying additional characteristics of the entities not encoded in the CMDI data (e.g. the address behind an organization).
Since large parts of older NaLiDa data, as well as other CMDI data, does not feature any authoritative IDs, the Seminar of Sprachwissenschaften at the University of Tuebingen developed

[5] `https://github.com/nmeisinger/CMDI2JSONLD`; Available under: GNU Public Licence 3

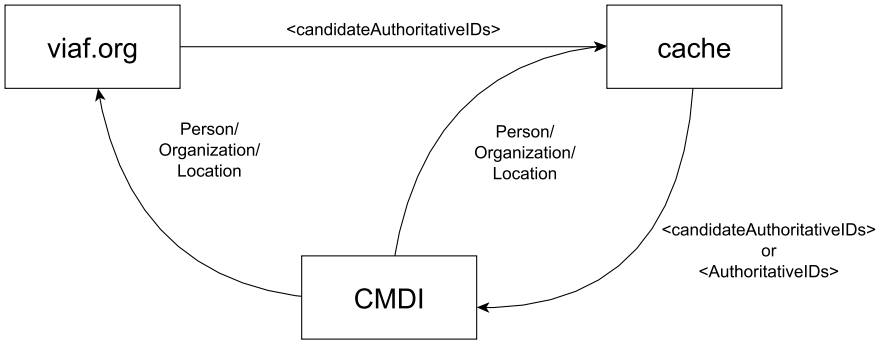


Figure 4: The basic workflow of the BioDataNER tool

the BioDataNER[6] tool – a Python application to automatically annotate CMDI data with authoritative IDs.
Its workflow is described in Figure 4: A directory of CMDI data is taken as input. For each CMDI, the tool will try to extract person, organization, or location names, from components that were specified prior in a separate file. If a file already contains authoritative IDs, they will be added to a cache. If not, an API request towards viaf.org will be executed to gain a list of potential IDs for the given named entity, referred to as *candidate IDs*. They need to be verified manually since, for a single entity, multiple entities sharing the same name or multiple IDs for the same entity might exist. Those candidate IDs will be then added to the cache.
In a second step, the CMDI files will be enriched with the IDs from the cache. Either the candidate IDs from viaf.org will be added or verified IDs. In the latter case, all candidate IDs are removed from the CMDI and the cache.
In theory, the tool can be applied to a variety of CMD profiles, although it was mainly tested and developed with profiles created by NaLiDa. As such, one drawback to it is that it can only enrich components with authoritative IDs. If persons or organizations are encoded on a concept level, IDs cannot be added. Also, if the corresponding profile schema does not support the added authoritative IDs component, the XML will no longer be valid.

As long as the BioDataNER tool is already installed, it can be called from the CMDI2JSONLD tool to enrich the input data with additional authoritative IDs.
For the transformation to JSON-LD, only validated authoritative IDs will be taken into consideration. Candidate entries are ignored.

4.3 Mapping CMDI to Schema.org

Since Schema.org is a well-documented, popular ontology, covering a wide range of different concept descriptions, it was decided to map CMD profiles to its vocabulary. Furthermore, if it is integrated into an HTML representation of a CMDI, it will increase the findability of the resource in all major search engines. To map CMD profiles to Schema.org, a system similar to the VLO facet mapping (Van Uytvanck et al., 2012) is utilized. Since Schema.org descriptions tend to be more complex in nature, the system had to be extended to handle a variety of different transformation scenarios.
The mapping is expressed in XML with the purpose to be as generic as possible. This was done, to not restrict users to Schema.org, although alternative ontologies were not explored in this thesis. The general outline of the mapping is expressed in Figure 5. As will become apparent, it closely models the resulting JSON-LD layout.
First, the corresponding schema of a CMD profile needs to be defined. If none is given, Schema.org's `DataSet` will be the default choice. This is the most sensible choice since the

[6] `https://github.com/SfS-ASCL/BiodataNER`; Developed by: Neele Falk and Nino Meisinger; Available under GNU Public Licence 3, Funded by: Mellon Foundation

```
<Mappings>
  <Schema.org Type (e.g. DataSet)>
    <Context/>
    <Profiles>
      <TextCorpusProfile>clarin.eu:cr1:p_1527668176122</
    TextCorpusProfile>
    </Profiles>
    <Mapping>
      <id/>
      <name/>
      <description/>
    </Mapping>
  </Schema.org Type (e.g. DataSet)>
  <Schema.org Type (e.g. SoftwareApplication)>
    [...]
  </Schema.org Type (e.g. SoftwareApplication)>
</Mappings>

```

Figure 5: General Structure of the mapping in XML

majority of CMD profiles describe a language resource, such as various corpora. It is also the only schema providing support to encode the `ResourceProxyList` element all CMD profiles contain.
Each schema in the mapping is paired with a JSON-LD `context` description. The `context` is freely modifiable, mainly to support ontologies other than Schema.org. It will be directly inserted into the resulting JSON-LD.
Finally, one can define mappings for all properties of the given schema. This can be done in three ways:

1. Specifying a concept's identifier from the CCR.
2. Specifying XPath expressions, in case no concepts can be used. XPath (Clark et al., 1999) is a query language for selecting nodes of an XML document.
3. Blacklisting profiles. If a profile is blacklisted, it will only be evaluated against the XPath expressions, not the concepts.

The last option differs from the VLO mapping. For the VLO mapping, certain subtrees of a CMDI can be excluded from concept evaluation. Most of the time, only a handful of profiles need to be blacklisted. In those cases, the relevant information is only encoded a single time and multiple subtrees need to be excluded. Specifying the correct XPath expression directly should help to increase the maintainability of the mappings.

Due to the complexity of both, Schema.org and CMD profiles, a simple property to CCR mapping is not always possible. Thus, there are three ways to define how a CMDI should map to a property. Each is illustrated by showing the responsible mapping, excerpts of a `TextCorpusProfile` defined by NaLiDa, and the resulting JSON-LD:

1. A simple 1:1 mapping. This can be done for properties that expect a simple type or a list of simple types. The `id` element will get transformed to `@id`:

```
<Mapping>
  <id>
    <pattern>/*:CMD/*:Header/*:MdSelfLink</pattern>
  </id>
  <name>
    <concept>http://hdl.handle.net/11459/CCR_C-2544
    _3626545e-a21d-058c-ebfd-241c0464e7e5</concept>
  </name>
  <description>
    <concept>http://hdl.handle.net/11459/CCR_C-2520
    _9eeedfb4-47d3-ddee-cfcb-99ac634bf1db</concept>
  </description>
</Mapping>

```

```
<cmd:Header>
  <cmd:MdSelfLink>https://hdl.handle.net/11022/example</
   cmd:MdSelfLink>
</cmd:Header>
[...]
<cmdp:GeneralInfo>
  <cmdp:ResourceName>Example ResourceName</cmdp:
   ResourceName>
  <cmdp:Descriptions>
    <cmdp:Description type="long" xml:lang="de">Beispiel
    Beschreibung</cmdp:Description>
    <cmdp:Description type="long" xml:lang="en">Example
    Description</cmdp:Description>
  </cmdp:Descriptions>
</cmdp:GeneralInfo>

```

```
{
  "@id": "https://hdl.handle.net/11022/example",
  "name": "Example ResourceName",
  "description": [
    {
      "@language": "de",
      "@value": "Beispiel Beschreibung"
    },
    {
      "@language": "en",
      "@value": "Example Description"
    },
  ],
}

```

2. If a property expects a single complex type, it must be specified in the mapping, by making use of the type attribute:

```
<Mapping>
  <license type="CreativeWork">
    <name>
      <concept>http://hdl.handle.net/11459/CCR_C-2457
    _45bbaa1a-7002-2ecd-ab9d-57a189f694a6</concept>
    <name>
  </license>
</Mapping>

```

```
<cmdp:Access>
  <cmdp:Licence>CC-BY-NC-SA</cmdp:Licence>
</cmdp:Access>

```

```
{
"license": {
    "@type": "CreativeWork",
    "name": ["CC-BY-NC-SA"]
  }
}

```

3. Some properties can take a list of complex types as value. In those cases using concepts is often prone to errors, as multiple concepts belonging to a component need to be mapped correctly to the type. In the following example, `firstName` and `lastName` need to belong to the same `Person` component. Mapping to entire components would have been one way to solve this issue but is hindered by the fact, that often only specific components contain the relevant information. Not all `Person` components in a CMDI for example also denote the creators of the resource. As such it was decided that only XPath expressions are allowed to be used.
   If the attribute `expand` is set to `"true"`, one can initiate this kind of mapping. Using the `<expand type="">` node the complex Type can be set. Using `expandPattern` defines the context node all further property XPath expressions will be evaluated against. This enables the use of relative XPath expressions to map the concepts inside the component to the properties of the complex type:

```
<Mapping>
  <creator expand="true">
    <expand type="Organization">
      <expandPattern>
      /*:CMD/*:Components/*:TextCorpusProfile/*:Project
    /*:Institution/*:Organisation
      </expandPattern>

      <name>
        <pattern>./*:name</pattern>
      </name>
      <sameAs>
        <pattern>./*:AuthoritativeIDs//*:id</pattern>
      </sameAs>
    </expand>

    <expand type="Person">
      <expandPattern>
      /*:CMD/*:Components/*:TextCorpusProfile/*:Creation
    /*:Creators/*:Person
      </expandPattern>

      <givenName>
        <pattern>./*:firstName</pattern>
      </givenName>
      <familyName>
       <pattern> ./*:lastName</pattern>
      </familyName>
      <sameAs>
        <pattern>./*:AuthoritativeIDs//*:id</pattern>
      </sameAs>
    </expand>
  </creator>
</Mapping>

```

```
<cmdp:Institution>
  <cmdp:Organisation>
    <cmdp:name>University of Tuebingen</cmdp:name>
    <cmdp:AuthoritativeIDs>
      <cmdp:AuthoritativeID>
        <cmdp:id>http://viaf.org/viaf/155435537</cmdp:id>
        <cmdp:issuingAuthority>VIAF</cmdp:
    issuingAuthority>
      </cmdp:AuthoritativeID>
      <cmdp:AuthoritativeID>
        <cmdp:id>http://isni.org/isni/0000000121901447</
    cmdp:id>
        <cmdp:issuingAuthority>GND</cmdp:issuingAuthority
    >
      </cmdp:AuthoritativeID>
    </cmdp:AuthoritativeIDs>
  </cmdp:Organisation>
</cmdp:Institution>
[...]

<cmdp:Creation>
  <cmdp:Creators>
    <cmdp:Person>
      <cmdp:firstName>Nino</cmdp:firstName>
      <cmdp:lastName>Meisinger</cmdp:lastName>
    </cmdp:Person>
  </cmdp:Creators>
</cmdp:Creation>

```

```
"creator": [
  {
    "@type": "Organization",
    "@id": "http://viaf.org/viaf/155435537",
    "name": "University of Tuebingen",
    "sameAs": [
        "http://viaf.org/viaf/155435537",
        "http://d-nb.info/gnd/36187-2"
    ]
  },
  {
    "@type": "Person",
    "givenName": "Nino",
    "familyName": "Meisinger"
  }
],

```

For the mapping to work, the following assumptions are made:

1. For any given overachieving type (e.g. `DataSet`), all CMD profiles share the same `context`.
2. Only one XPath expression applies to a given CMD profile. If not, the expressions currently in use are too generic and need to be specified.
3. If a concept is found in a given CMDI, the XPath expressions are not evaluated.

If a `lang` attribute is found inside an XML node, it will be automatically converted to a `@language/@value` object in JSON-LD. Furthermore, the mapping allows one to nest property/type relations if necessary. For example, a `locationCreated` property, that expects a `address` property of type `PostalAddress` can be easily realized with the mapping.
If all concepts and XPath patterns do not yield any result for a given property, it will not be included in the JSON-LD. This avoids null nodes, as well as reduces the size of the resulting files.
Since the mapping relies on an XSLT 3.0 stylesheet, XPath 3.0 is supported. This gives a lot of flexibility in how to design the XPath expressions, although it should be kept in mind that the more complicated the expressions become, the more error-prone the transformation process.

4.4 Transforming CMDI to JSON

Mapping CMDI data to Schema.org is the first step in the JSON-LD transformation. However, Schema.org is not detailed enough to encode all information of any given profile. As such, the second part of the resulting JSON-LD, consists of a CMDI representation using the unique identifiers in the given profile schema:

```
<cmdp:Creators>
    <cmdp:Person>
        <cmdp:firstName>Nino</cmdp:firstName>
        <cmdp:lastName>Meisinger</cmdp:lastName>
        <cmdp:role>creator</cmdp:role>
    </cmdp:Person>
</cmdp:Creators>

```

```
    "clarin.eu:cr1:c_1442920133044": {
        "@type": "Component",
        "clarin.eu:cr1:c_1447674760335": {
            "@type": "Component",
            "http://hdl.handle.net/11459/CCR_C-6493_e10ad1ee-82ac-c03f-a582-b14d5787576b": "Nino",
            "http://hdl.handle.net/11459/CCR_C-6492_9d5f20aa-e6bd-c141-15df-ffdae159e5e1": "Meisinger",
            "http://hdl.handle.net/11459/CCR_C-3807_1479532e-b741-f5d4-99f4-881c5908cfd9": "creator"
        }
    },

```

This transformation operates on a simple algorithm to transform XML to JSON:

1. All element names of the CMDI will be replaced by their respective identifier.

2. The identifier of a component/concept always acts as the key name of a JSON object.
3. XML nodes that contain text are transformed into key/value pairs, where the value is the text.
4. If an XML node contains children, the value will be another JSON object, where the keys will be the children's component/concept identifier.
5. If two adjacent XML nodes share the same name, they will be put into a list to avoid duplicate names.
6. If a XML node contains a `lang` attribute, it will get converted into a `@language/@value` pair.
7. Components will receive the type `Component`, defined in the context of the JSON-LD.
8. The resulting JSON-LD's type will be the profile's identifier, as well as the Schema.org schema from the mapping.
9. Empty nodes and nodes where all descendants resolve to empty nodes will be ignored.
10. XML nodes that contain text, yet have no concept identifier, will be ignored.
11. If a component has no identifier, but one of its descendants has an identifier and text, the children will get appended to its parent.
12. There are no two elements in any CMDI that share the name, yet have a different identifier.

5 Results and Discussion

Table 1: Profiles that were tested with the mapping system

Component Ref	Profile Name	Files
clarin.eu:cr1:p_1361876010587	AnnotatedCorpusProfile-DLU	3
clarin.eu:cr1:p_1456409483202	BatImageBundle	3
clarin.eu:cr1:p_1361876010608	BilingualDictionaryProfile-DLU	3
clarin.eu:cr1:p_1527668176116	CEDIFOR_TextCorpus	3
clarin.eu:cr1:p_1427452477080	CommunicationProfile	3
clarin.eu:cr1:p_1505397653792	CourseProfile	1
clarin.eu:cr1:p_1527668176125	CourseProfile	8
clarin.eu:cr1:p_1455633534543	DGDCorpus	3
clarin.eu:cr1:p_1456409483189	DGDEvent	3
clarin.eu:cr1:p_1288172614023	DcmiTerms	266
clarin.eu:cr1:p_1524652309872	ExperimentProfile	1
clarin.eu:cr1:p_1527668176126	ExperimentProfile	113
clarin.eu:cr1:p_1521028545544	Framework	3
clarin.eu:cr1:p_1371047304769	FrequencyListProfile-DLU	1
clarin.eu:cr1:p_1369752611624	IDSAGD_Corpus	3
clarin.eu:cr1:p_1361876010680	IDSAGD_Event	3
clarin.eu:cr1:p_1369140737145	IDSAGD_Speaker	3
clarin.eu:cr1:p_1527668176047	LCC_CorpusProfile	3
clarin.eu:cr1:p_1527668176046	LCC_DataProviderProfile	3
clarin.eu:cr1:p_1290431694579	LexicalResourceProfile	1
clarin.eu:cr1:p_1548239945774	LexicalResourceProfile	1
clarin.eu:cr1:p_1527668176123	LexicalResourceProfile	54
clarin.eu:cr1:p_1380106710823	LexicalResourceProfile-DLU	2
clarin.eu:cr1:p_1387365569700	LexicalResourceProfile-DLU	3
clarin.eu:cr1:p_1288172614026	OLAC-DcmiTerms	2
clarin.eu:cr1:p_1548239945774	ResourceBundle	5
clarin.eu:cr1:p_1331113992512	SL-IPROSLA	3
clarin.eu:cr1:p_1381926654456	SpeechCorpus-DLU	3
clarin.eu:cr1:p_1527668176128	SpeechCorpusProfile	1
clarin.eu:cr1:p_1422885449343	SpokenCorpusProfile	1
clarin.eu:cr1:p_1524652309874	TextCorpusProfile	1
clarin.eu:cr1:p_1562754657343	TextCorpusProfile	1
clarin.eu:cr1:p_1527668176122	TextCorpusProfile	131
clarin.eu:cr1:p_1527668176124	ToolProfile	12
clarin.eu:cr1:p_1380106710825	TreebankProfile-DLU	3
clarin.eu:cr1:p_1396012485083	VALID	3
clarin.eu:cr1:p_1320657629644	WebLichtWebService	54
clarin.eu:cr1:p_1345561703620	collection	6
clarin.eu:cr1:p_1417617523856	lat-SL-session	3
clarin.eu:cr1:p_1407745712064	lat-corpus	4
clarin.eu:cr1:p_1387365569699	media-corpus-profile	3
clarin.eu:cr1:p_1336550377513	media-session-profile	3
clarin.eu:cr1:p_1361876010571	resourceInfo	3
clarin.eu:cr1:p_1375880372976	singlePaperPackage	3
clarin.eu:cr1:p_1475136016232	talkbank-license-session	3
clarin.eu:cr1:p_1357720977494	wnd_subcollection_core_data	3

To test how well the CMDI2JSONLD tool performs against real data, mappings were created and tested against 46 CMD profiles, as shown in Table 1. 33 do not originate from NaLiDa and

were retrieved from the CLARIN Virtual Language Observatory[7]. For each of them between three to five profile instances were randomly sampled. All CMDIs defined with NaLiDa profiles, as well as `DcmiTerms`, were taken directly from the Tuebingen Archive of Language Resources[8], which explains their high number.
Each CMDI file was enriched with authoritative IDs using the BioDataNER tool. Its cache contained authoritative IDs from all the CMDIs taken from the Tuebingen Archive of Language Resources.
A total of 741 CMDI files (Meisinger, 2021) were transformed to JSON-LD. All files were validated, to make sure that they are syntactically correct JSON, as well as valid JSON-LD.
`singlePaperPackage` and `DcmiTerms` were mapped to schemata of type `ScholaryArticle`, `WebLichtWebService` to `SoftwareApplication`. The remaining profiles were mapped to the default `DataSet`.

5.1 Querying the data with SPARQL

Having mapped all CMDI data to a Schema.org schema should provide several benefits:
Similar to the VLO facets, semantic interoperability is increased. No matter which profile, the underlying CMDI catalogue can be queried using the Schema.org properties. Normally one would identify all possible paths to a resource's name across all profiles using XQuery, a query language for XML documents (Boag et al., 2002), or similar on the original CMDI data. With the Linked Data, querying for the Schema.org `DataSet`'s name property will achieve the same result.
Second, retrieving properties not contained in the Schema.org markup should be equally possible by using the CCR/Component Registry identifiers. This enables users to create more complex queries when combined with Schema.org markup.
More importantly, however, by making use of the authoritative IDs, queries retrieving information from other Linked Data should become possible. This opens up new possibilities that the original XML format of the CMDI data does not provide.
The viability of the resulting JSON-LD regarding these three cases can be showcased with the following SPARQL queries. The first query will cover the first two cases:

[7] `https://vlo.clarin.eu/?0`, Last Accessed: 19.09.2021
[8] `https://talar.sfb833.uni-tuebingen.de/`, Last Accessed: 19.09.2021

```
PREFIX schema: <http://schema.org/>

SELECT DISTINCT ?Handle
(GROUP_CONCAT(DISTINCT ?TextCorpusProfile;separator=";") as ?
    Names)
?Corpustype ?LicenceName
(COUNT(DISTINCT ?Creator) AS ?Creators)
(COUNT(DISTINCT ?Distribution) AS ?DistributionLinks)
WHERE{
  ?Handle ?p <clarin.eu:cr1:p_1527668176122> .
  ?Handle schema:distribution ?Distribution.
  ?Handle schema:name ?TextCorpusProfile .
  ?Handle <clarin.eu:cr1:c_1527668176117> ?TextCorpusContext
    .
  OPTIONAL{ ?Handle schema:creator ?Creator . }
  OPTIONAL{ ?TextCorpusContext <http://hdl.handle.net/11459/
    CCR_C-3822_ed57a8f6-05f2-0731-6350-8158e74fcb5f> ?
    Corpustype . }
  OPTIONAL{
    ?Handle schema:license ?Licence .
    ?Licence schema:name ?LicenceName
  }
}
GROUP BY ?Handle ?Corpustype ?LicenceName

```

The query extracts only NaLiDa `TextCorpusProfile`. It then returns the link to the resource, all names found for a given resource, the corpus type, its licence, how many creators were involved with it, as well as the number of files a user can download. As can be seen, one can easily switch between Schema.org markup and CCR/Component Registry markup to query for data. 131 profiles are returned, which fits the number of all profiles in the database.

SPARQL can only query pre-specified endpoints. However, one can design a query accessing multiple endpoints. As such, it is necessary to know, which endpoints provide information one is interested in.
The following query uses the local CMDI catalogue to extract person names, as well as their birthdate and gender as noted on Wikidata[9] and the GND4C SPARQL endpoint[10]. It should be noted, that the latter is not an official SPARQL endpoint from the German National Library, as they currently do not provide a SPARQL interface[11]. Instead, the GND4C endpoint currently operates using GND data from 2019[12]:

[9] `https://query.wikidata.org/`, Last Accessed: 19.09.2021
[10] `http://gnd4c.digicult-verbund.de/sparql.html`, Last Accessed: 19.09.2021
[11] `https://www.dnb.de/EN/Professionell/Metadatendienste/Datenbezug/LDS/lds_node.html`, Last Accessed: 07.09.2021
[12] `http://gnd4c.digicult-verbund.de/`, Last Accessed: 07.09.2021

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX schema: <http://schema.org/>
PREFIX wdt: <http://www.wikidata.org/prop/direct/>
PREFIX wikibase: <http://wikiba.se/ontology#>
PREFIX gndo: <http://d-nb.info/standards/elementset/gnd#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

SELECT DISTINCT ?givenName ?familyName ?birthdateWikidata ?
    genderWikidata ?birthdateGND ?genderGND
WHERE {
  SERVICE <http://localhost:3030/CMDI2JSONLD/sparql>
  {
    ?subject schema:creator ?creator .
    ?creator schema:givenName ?givenName .
    ?creator schema:familyName ?familyName .
    ?creator schema:sameAs ?uri .
    filter(regex(str(?uri), "viaf"))
    BIND(REPLACE(str(?uri), ".*/", "") AS ?id)
  }
  OPTIONAL{
  SERVICE <http://query.wikidata.org/sparql>
    {
      ?item wdt:P214 ?id .
      OPTIONAL{?item wdt:P569 ?birthdateWikidata .}
      OPTIONAL{?item wdt:P21 ?genderWikidata .}
    }
  }
   OPTIONAL{
   SERVICE <http://gnd4c.digicult-verbund.de:3030/gndt/sparql
   >
    {
      ?s owl:sameAs ?uri .
      OPTIONAL{?s gndo:dateOfBirth ?birthdateGND .}
      OPTIONAL{?s gndo:gender ?genderGND .}
    }
  }
}

GROUP BY ?creator ?givenName ?familyName ?id ?
    birthdateWikidata ?birthdateGND ?genderGND ?genderWikidata

```

First, the query extracts the name of all creators, as well as their VIAF ID from the local database. Said ID is used to connect to the Wikidata SPARQL endpoint to extract the corresponding entity's birthdate and gender if one exists. The same is repeated for the GND SPARQL endpoint.
In total, 59 entities are enriched with a viaf.org ID in the local dataset. For 39, information was either found on Wikidata or on GND and returned with the query. Gender was found more often than a date of birth. An excerpt of the result set of the query, showing five entries, can be seen in Figure 6.
This paints promising results. Among persons with an authoritative ID, information from either Wikidata or GND4C could be retrieved for more than 50%. Should the German National

Library decide to offer their own SPARQL endpoint in the future, chances are, that this number will further increase.

QUERY RESULTS

Table | Raw Response

Showing 1 to 59 of 59 entries

Search: Show All entries

	givenName	familyName	birthdateWikidata	genderWikidata	birthdateGND	genderGND
1	"Tilman"	"Berger"	"1956-07-08T00:00:00Z"^^xsd:dateTime	<http://www.wikidata.org/entity/Q6581097>	"1956"^^xsd:gYear	<http://d-nb.info/standards/vocab/gnd/gender#male>
2	"Erhard"	"Hinrichs"	"1954-01-01T00:00:00Z"^^xsd:dateTime	<http://www.wikidata.org/entity/Q6581097>	"1954"^^xsd:gYear	<http://d-nb.info/standards/vocab/gnd/gender#notKnown>
3	"Peter"	"Koch"	"1951-03-01T00:00:00Z"^^xsd:dateTime	<http://www.wikidata.org/entity/Q6581097>	"1951-03-01"^^xsd:date	<http://d-nb.info/standards/vocab/gnd/gender#male>
4	"Anne"	"Cutler"	"1945-01-17T00:00:00Z"^^xsd:dateTime	<http://www.wikidata.org/entity/Q6581072>	"1945"^^xsd:gYear	<http://d-nb.info/standards/vocab/gnd/gender#female>
5	"Wiltrud"	"Mihatsch"	"1970-01-01T00:00:00Z"^^xsd:dateTime		"1970"^^xsd:gYear	<http://d-nb.info/standards/vocab/gnd/gender#female>

Figure 6: Excerpt of the result set from the second SPARQL query

5.2 Quality of the Linked Data

Overall, the second query shows that the usage of authority control is useful to identify entities across other Linked Data. Just using the name of a person might either lead to multiple entries, due to other people sharing the same name or no entries, should the other database use a different spelling for a person. Since JSON-LD, an RDF serialization format, was used for the transformation, it adheres to the rules described Berners-Lee (2006) as to what constitutes as Linked Data.

Despite the overall success of the transformation of CMDI to JSON-LD, there are still a few issues decreasing its quality:
During the work on this thesis, it became apparent that some creators do not correctly distinguish between the CCR and Component Registry. Some profiles treat concepts as components, by adding children elements to them. This increases the difficulty of parsing the profiles in question. Other profiles reuse the same CCR entries for different element nodes in their profiles. This means, inside the profile, the element nodes have different names, but they all use the same (supposedly) unique identifier.
Not every CMD profile uses the same language identifiers inside their `lang` attribute. Some use ISO 639-2, others ISO 639-3, and others came up with individual solutions. This makes it intransparent, how to filter for certain languages.
Different CMD creators treat empty values for concept definitions differently. Some will remove empty nodes, some will keep them. In some instances, empty strings or custom values are inserted (e.g. “No known copyright restrictions” for an empty licence field).
Similarly, different profiles treat concepts differently as well. Some use the licence concept, for example, to add a URL to the licence of a resource, others will insert a string of its name. This might affect the mapping to Schema.org vocabulary. The licence property either expects a `URL` property or a `CreativeWork` property. If the concept does not indicate, whether a URL or not is used, one has to map profiles to XPath expressions.
Sometimes, properties are mapped to concepts, that would be better suited for components. `Address` would be such a case. Due to that, addresses are just a simple string. Separating them into “Street”, “Zip Code”, “City”, etc. would greatly increase the richness of the CMDI metadata, as well as its JSON-LD representation.
For some concepts, using IRIs would greatly increase the linking aspect. Instead of string values for a licence, using a URL would increase the machine readability. It also removes potential

ambiguity which licence is meant for users.
It would be unwise for the transformation system to handle these issues, as it would only treat the symptoms and not the cause. Fixing these issues in the underlying CMDI metadata, would not just benefit the JSON-LD transformation, but also increase semantic interoperability in general.

The transformation system itself can be improved as well. There are currently a few edge cases, where the transformation of a CMDI to JSON might behave in an unexpected way. For one, the CMDI's `Header`, `Resources` and `IsPartOfList` are not converted. The reason for that is because all elements outside the `Components` element of a CMDI do not have unique identifiers. For now, `Resources` are mapped to Schema.org's `DataSet distribution` property however.
All attributes except the `lang` attribute are ignored for two reasons: First, they do not possess a unique identifier that could be used as IRI. Second, should an element contain a language attribute, the node will get converted to a `@language/@value` object. In that case, it is not possible to append the information from the remaining attributes to the JSON object.
In the future, it would be beneficial to find a way of encoding attributes. While often neglectable, in some cases, they carry important information. Profiles defined by NaLiDa for example use the `ResourceProxyListInfo` component. It encodes additional information for the downloadable files of a resource. For that, it uses an attribute to make clear which file it applies to. Losing that information renders the component mostly useless.
Furthermore, if two non-adjacent XML nodes share the same name, they will not be grouped into a list. While this does not result in invalid JSON, most JSON implementations do not accept duplicate keys.
Lastly, components are currently referred to by their identifier. Since this is currently not a URL, linking to an RDF schema, the resulting data does not fulfil all requirements for good Linked Data. Similar is the case for the CCR entries. Once, RDF representations are implemented for both registries, as currently planned, this problem will resolve itself.

The mapping system could benefit from a way to express a list of complex types using Component Registry information. This would ideally fully automate the system, so new profiles can be transformed to JSON-LD without any manual labour involved. Instead of checking new profiles for corresponding XPath expressions, they would be compatible with the pre-existing mapping as long they reuse existing components and concepts.

6 Conclusion

This thesis explored a new way of transforming CMDI into a Linked Open Data format. Using the CCR and Component Registry, a new system was developed to map concepts onto Schema.org vocabulary – an established ontology that gained widespread adoption in the past few years. As Schema.org does not provide complex enough schemata to encode all information of a CMD profile, a second representation was developed to represent the profiles in a Linked Data format using the unique identifiers of concepts and components. This resulted in JSON-LD files containing both representations. Furthermore, a new tool to enrich CMDI data with links to authority files was applied to further improve the quality of the resulting Linked Data.
While not perfect, the Linked Data provides exciting new possibilities in how to interact with the metadata. Users are able to write queries that can extract information from other Linked Data on the Web not encoded in the local data. This enables users to collect more information about entities occurring, be it more details about persons who created a resource or organizations.
In the future, it might also become possible to collect information from a resource's metadata and the resource itself through a single query, as long as the resources are provided as Linked Data themselves. In the field of Linguistics, this could enable researchers to collect specific information from a variety of different corpora without much effort.
Furthermore, the resulting Schema.org vocabulary can be integrated into the HTML representation of the CMDI data. That way, it will provide better results by search engines for users and get harvested and utilized in Google's Knowledge Graph.

7 Bibliography

References

Auer, S., C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives (2007). Dbpedia: A nucleus for a web of open data. In *The semantic web*, pp. 722–735. Springer.

Beckett, D. and B. McBride (2004). Rdf/xml syntax specification (revised). *W3C recommendation 10*(2.3).

Belbin, L., E. Wallis, D. Hobern, and A. Zerger (2021). The atlas of living australia: History, current state and future directions. *Biodiversity Data Journal 9*.

Berners-Lee, T. (2006). Linked data – design issues. `https://www.w3.org/DesignIssues/LinkedData.html`, Last accessed on 2021-08-22.

Berners-Lee, T. (2010). Is your linked open data 5 star? `https://www.w3.org/DesignIssues/LinkedData.html`, Last accessed on 2021-08-22.

Berners-Lee, T., D. Connolly, and R. R. Swick (1999). Web architecture: Describing and exchanging data. *W3C Nota 7*.

Berners-Lee, T., J. Hendler, and O. Lassila (2001). The semantic web. *Scientific american 284*(5), 34–43.

Boag, S., D. Chamberlin, M. F. Fernández, D. Florescu, J. Robie, J. Siméon, and M. Stefanescu (2002). Xquery 1.0: An xml query language.

Bray, T., J. Paoli, C. M. Sperberg-McQueen, E. Maler, F. Yergeau, and J. Cowan (2000). Extensible markup language (xml) 1.0.

Brickley, D., R. V. Guha, and A. Layman (1999). Resource description framework (rdf) schema specification.

Broeder, D., M. Windhouwer, D. Van Uytvanck, T. Goosen, and T. Trippel (2012). Cmdi: a component metadata infrastructure. In *Describing LRs with metadata: towards flexibility and interoperability in the documentation of LR workshop programme*, Volume 1.

Cabeda, J. (2017). Semantic web is dead, long live the ai. *Hackernoon, June 4*.

Cagle, K. (2016). Why the semantic web has failed. `https://www.linkedin.com/pulse/why-semantic-web-has-failed-kurt-cagle/`, Last accessed on 2021-08-22.

Clark, J., S. DeRose, et al. (1999). Xml path language (xpath).

Doctorow, C. (2001). Metacrap: Putting the torch to seven straw-men of the meta-utopia. *Retrieved June 10*, 2003.

Dolega, L., F. Rowe, and E. Branagan (2021). Going digital? the impact of social media marketing on retail website traffic, orders and sales. *Journal of Retailing and Consumer Services 60*, 102501.

Durco, M. and M. Windhouwer (2014, May). From clarin component metadata to linked open data. pp. 13.

Eric Prud'hommeaux, A. S. (2008). Sparql query language for rdf. `https://www.w3.org/TR/rdf-sparql-query/`, Last accessed on 2021-08-22.

Facebook (2021). The open graph protocol. `https://ogp.me/`, Last accessed on 2021-08-22.

Getman, J., J. Ellis, Z. Song, J. Tracey, and S. M. Strassel (2017). Overview of linguistic resources for the tac kbp 2017 evaluations: Methodologies and results. In *TAC*.

Guha, R. V., D. Brickley, and S. Macbeth (2016). Schema. org: evolution of structured data on the web. *Communications of the ACM 59*(2), 44–51.

Haase, P., A. Nikolov, J. Trame, A. Kozlov, and D. M. Herzig (2017). Alexa, ask wikidata! voice interaction with knowledge graphs using amazon alexa. In *International Semantic Web Conference (Posters, Demos & Industry Tracks)*.

Hallo, M., S. Luján-Mora, A. Maté, and J. Trujillo (2016). Current state of linked data in digital libraries. *Journal of Information Science 42*(2), 117–127.

Hoppermann, C., T. Trippel, and C. Zinn (2011). Managing linguistic resources by enriching their metadata with linked data. In *10th International Semantic Web Conference (ISWC2011) Bonn*. Citeseer.

Initiative, D. C. M. (2021). Dublin core metadata initiative. `https://www.dublincore.org/`, Last accessed on 2021-08-22.

International Organization for Standardization (2019). ISO 24622-2:2019(en) Language resource management Component Metadata Infrastructure (CMDI) Part 2: Component metadata specification language.

Jones, L. (2020). Women's representation and voice in media coverage of the coronavirus crisis. *Global Institute for Women's Leadership, Kings College London*.

Kumar, S., M. Ujjal, and B. Utpal (2013). Exposing marc 21 format for bibliographic data as linked data with provenance. *Journal of Library Metadata 13*(2-3), 212–229.

Lassila, O. (1997). Introduction to rdf metadata–w3c note. *World Wide Web Consortium, Cambridge, MA. URL http://www. w3. org/TR/NOTE-rdf-simple-intro-971113. html*.

Library of Congress (2021). Marc 21 format for bibliodgraphic data. `https://www.loc.gov/marc/bibliographic/`, Last accessed on 2021-08-22.

Malyshev, S., M. Krötzsch, L. González, J. Gonsior, and A. Bielefeldt (2018). Getting the most out of wikidata: Semantic technology usage in wikipedia's knowledge graph. In *International Semantic Web Conference*, pp. 376–394. Springer.

McCrae, J. P., C. Chiarcos, F. Bond, P. Cimiano, T. Declerck, G. De Melo, J. Gracia, S. Hellmann, B. Klimek, S. Moran, et al. (2016). The open linguistics working group: Developing the linguistic linked open data cloud. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pp. 2435–2441.

McCrae, J. P. and P. Cimiano (2015). Linghub: a linked data based portal supporting the discovery of language resources. *SEMANTiCS (Posters & Demos) 1481*, 88–91.

McCutcheon, S. (2009). Keyword vs controlled vocabulary searching: the one with the most tools wins. *Indexer 27*(2).

Meisinger, N. (2021). Integrating component metadata into the linked data world: a case study - cmdi metadata collection. Persistent identifier: `https://hdl.handle.net/11022/0000-0007-F0AA-A`, Last accessed on 2021-09-19.

Mika, P. (2017). What happened to the semantic web? In *Proceedings of the 28th ACM Conference on Hypertext and Social Media*, pp. 3–3.

Mori, S., T. Sasada, Y. Yamakata, and K. Yoshino (2012). A machine learning approach to recipe text processing. In *Proceedings of the 1st Cooking with Computer Workshop*, pp. 29–34.

Mountantonakis, M. and Y. Tzitzikas (2017). How linked data can aid machine learning-based tasks. In *International Conference on Theory and Practice of Digital Libraries*, pp. 155–168. Springer.

Neish, P. (2015). Linked data: what is it and why should you care? *The Australian Library Journal 64*(1), 3–10.

Pazooki, F. and S. Keshavarzian (2020). Bibframe: a new bibliographic framework for linked data environment. *Library and Information Science Research 9*(2), 226–241.

Pringle, G., L. Allison, and D. L. Dowe (1998). What is a tall poppy among web pages? *Computer Networks and ISDN Systems 30*(1-7), 369–377.

Schuurman, I., M. Windhouwer, O. Ohren, and D. Zeman (2016). Clarin concept registry: the new semantic registry. In *Selected Papers from the CLARIN Annual Conference 2015, October 14–16, 2015, Wroclaw, Poland*, Number 123, pp. 62–70. Linköping University Electronic Press.

Sporny, M. (2014.). Json-ld and why i hate the semantic web. `https://slacker.ro/2014/01/22/json-ld-and-why-i-hate-the-semantic-web/`, Last accessed on 2021-08-22.

Sporny, M., D. Longley, G. Kellogg, M. Lanthaler, and N. Lindström (2014). Json-ld 1.0. *W3C recommendation 16*, 41.

Swartz, A. (2013). Aaron swartz's a programmable web: An unfinished work. *Synthesis lectures on the semantic web: Theory and Technology 3*(2), 1–64.

Target, S. (2018). Why the semantic web has failed. `https://twobithistory.org/2018/05/27/semantic-web.html#`, Last accessed on 2021-08-22.

Trippel, T. and C. Zinn (2016, 10). Just for the record, cmdi should be about semantic interoperability.

van Kleef, P. (2018). Dbpedia usage report (as of 2018–01–01). `https://medium.com/virtuoso-blog/dbpedia-usage-report-as-of-2018-01-01-8cae1b81ca71`, Last accessed on 2021-08-22.

Van Uytvanck, D., H. Stehouwer, and L. Lampen (2012). Semantic metadata mapping in practice: the virtual language observatory. In *LREC 2012: 8th International Conference on Language Resources and Evaluation*, pp. 1029–1034. European Language Resources Association (ELRA).

Verborgh, R. and M. Vander Sande (2020). The semantic web identity crisis: in search of the trivialities that never were. *Semantic Web 11*(1), 19–27.

Vrandečić, D. (2012). Wikidata: A new platform for collaborative data collection. In *Proceedings of the 21st international conference on world wide web*, pp. 1063–1064.

Wang, X., Y. Jiang, N. Bach, T. Wang, Z. Huang, F. Huang, and K. Tu (2020). Automated concatenation of embeddings for structured prediction. *arXiv preprint arXiv:2010.05006*.

Windhouwer, M., E. Indarto, and D. Broeder (2017). Cmd2rdf: building a bridge from clarin to linked open data. *Data Archiving and Networked Services (DANS)*, 95–103.

Zinn, C., T. Trippel, S. Kaminski, and E. Dima (2016). Crosswalking from cmdi to dublin core and marc 21. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pp. 2489–2495.