SMC4LRT - Master Outline

Matej Durco

October 15, 2013

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation / problem statement

While in the Digital Libraries community a consolidation already took place and global federated networks of digital library repositories are set up, in the field of Language Resource and Technology the landscape is still scattered, although meanwhile looking back at a decade of standardization and integration efforts. One main reason seems to be the complexity and diversity of the metadata associated with the resources, stemming from the wide range of resource types combined with project-specific needs. (Chapter 3 analyses the disparity in the data domain.)

This situation has been identified by the community and numerous standardization initiatives had been undertaken. The process has gained a new momentum thanks to large framework programmes introduced by the European Commission aimed at fostering the development of common large-scale international research infrastructures. One key player in this development is the project CLARIN (see section 4.1). The main objective of this initiative is to make language resources and technologies (LRT) more easily available to scholars by means of a common harmonized architecture. One core pillar of this architecture is the *Component Metadata Infrastructure* (cf. 4.2) – a distributed system consisting of multiple interconnected modules aimed at creating and providing metadata for LRT in a coherent harmonized way.

This work discusses one module within the Component Metadata Infrastructure – the *Semantic Mapping Component* – dedicated to overcome or at least ease the semantic interoperability problem stemming from the heterogeneity of the resource descriptions, without the reductionist approach of imposing one common description schema for all resources.

1.2 Main Goal

The primary goal of this work is to ***enhance search functionality*** *over a large heterogeneous collection of resource descriptions* in the field of LRT, henceforth referred to as semantic search, distincting it from the underlying processing, referred to as semantic mapping.

The – notoriously polysemic – term "mapping" can have three different meanings within this work, that also translate into three corresponding subgoals:

crosswalk link related fields in different metadata formats

interpret translate string labels in field values to semantic entities

visualize provide appropriate means to explore the domain data.

The work can further be divided along the schema – instance duality. Figure 1.1 spans the conceptual space of this work and depicts the dependencies between individual subgoals.

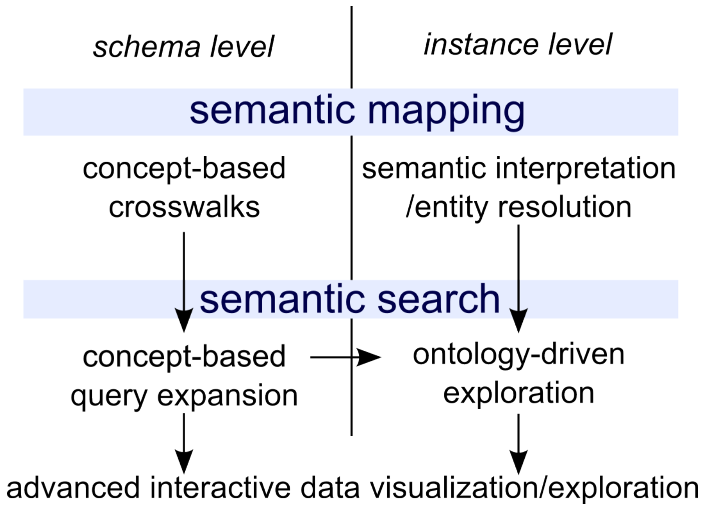

schema level
instance level
semantic mapping
concept-based crosswalks
semantic interpretation /entity resolution
semantic search
concept-based query expansion
ontology-driven exploration
advanced interactive data visualization/exploration

Figure 1.1: The conceptual space of this work

Crosswalk service

Semantic interoperability has been one of the main concerns addressed by the CMDI and appropriate provisions were weaved into the underlying meta-model as well as all the modules of the infrastructure. The task of the crosswalk service – the primary part of the SMC module – is to collect the relevant information maintained in the registries of the infrastructure and process it to generate mappings, i.e. *crosswalks* between fields in heterogeneous metadata schemas that can serve as basis for concept-based search.

Thus, the goal is not primarily to produce the crosswalks but rather to develop the service serving existing ones.

Concept-based query expansion

Once the crosswalks are available, they can be used to rewrite user queries, so that they match equivalent or similar fields across heterogeneous metadata schemas resulting in higher recall when searching.

Example Confronted with a user query searching in the notorious dublincore:title the query has to be *expanded* to all the semantically near fields (*concept cluster*), that are however labelled (or even structured) differently in other schemas like:

resourceTitle, BookTitle, tei:titleStmt, Corpus/GeneralInfo/Name

The expansion cannot be solved by simple string matching, as there are other fields labeled with the same (sub)strings but with different semantics, that shouldn't be considered:

Project/Title, Organisation/Name, Country/Name, LanguageName

Semantic interpretation

The problem of different labels for semantically similar or even identical entities is even more so virulent on the level of individual values in the fields of the instance data. A number of metadata fields (like organization or resource type) have a constrained value domain that yet cannot be explicitly exhaustively enumerated. This leads to a chronically inconsistent use of labels for referring to entities. (As the instance data shows, some organizations are referred to by more than 20 different labels.) Thus, one goal of this work is to propose a mechanism to map (string) values in selected fields to entities defined in corresponding vocabularies.

Ontology-driven data exploration

Based on the results of the previous parts of the work – crosswalks and semantic interpretation – the discussed dataset can be expressed as one big ontology. Consequently, semantic web technologies can be applied giving the user new means of *exploring the dataset.*

Example Ontology-driven search – Starting from a list of topics the user can browse an ontology to find institutions concerned with those topics and retrieve a union of resources for the resulting cluster. Thus in general the user is enabled to work with the data based on information that is not present in the original dataset, but rather in external interlinked semantic resources.

Visualization

Given the large, heterogeneous and complex dataset, it seems indispensable to equip the user with advanced means to explore and interact with it. Hence this subgoal aimed to propose ways of visualizing the data at hand.

1.3 Method

We start with examining the existing data and with the description of the existing infrastructure in which this work is embedded.

Building on this groundwork, in accordance with the first subgoal, we lay out the design of the service for handling crosswalks and concept-based query expansion. We describe the workflow, the central methods and the role of the module relative to other parts of the infrastructure.

Subsequently, we explore the ways of integrating this service into exploitation tools (metadata search engines), to enhance search/retrieval through the use of semantic relations between concepts or categories. This theoretical part will be accompanied by a prototypical implementation as proof of concept.

Note that in this work, the focus lies on the actual method to generate and apply the crosswalks – expressed in the specification and operationalized in the (prototypical) implementation of the service – rather than trying to establish final, accomplished crosswalks between the schemas. In fact, given the great diversity of resources and research tasks, a “final” complete alignment does not seem achievable at all. Therefore also the focus shall be on *dynamic mapping*, i.e. to enable the users to directly manipulate the level of use of the crosswalks or even apply custom crosswalks depending on their current task or research question being able to actively influence the recall/precision ratio of the search results, and essentially to modulate the semantic search space.

Serving the second subgoal – semantic interpretation on the instance level – we will propose the expression of all of the domain data (from meta-model specification to instances) in RDF, linking to corresponding entities in appropriate external semantic resources (controlled vocabularies, ontologies). Once the dataset is expressed in RDF, it can be exposed via a semantic web application and published as another nucleus of *Linked Open Data* in the global *Web Of Data*.

A separate evaluation of the usability of the proposed semantic search solution is indicated, examining the user interaction with and display of the relevant additional information in the user search interface, however this issue can only be tackled marginally and will have to be outsourced into future work.

1.4 Expected Results

The main result of this work will be the *specification* of the two modules concept-based search and the underlying crosswalk service. This theoretical part will be accompanied by a proof-of-concept *implementation* of the components and the sample results.

Another result of the work will be the original dataset expressed as RDF interlinked with existing external resources (ontologies, knowledge bases, vocabularies), effectively laying a foundation for providing this dataset as *Linked Open Data*[1].

Crosswalk service specification and a basic implementation of the service

Concept-based search design of the query expansion and prototypical integration with a search engine

Visualization tool design of an application for interactive exploration of the concerned dataset

LinkedData translation of the source dataset to RDF-based format with links into existing datasets, ontologies, knowledge bases

1.5 Structure of the work

The work starts with examining the state of the art work in the two fields language resources and technology and semantic web technologies in chapter 2, followed by administrative chapter A explaining the abbreviations and formatting conventions used throughout this work.

In chapter 3 we analyze the situation in the data domain of LRT metadata and in chapter 4 we discuss the individual software components of the infrastructure underlying this work.

The main part of the work is found in chapters 5 and 6 laying out the design of the software module and a proposal how to model the data in RDF respectively.

The results are discussed in chapter 7. Finally, in chapter 8 we summarize the findings of the work and lay out where it could develop in the future.

1.6 Keywords

semantic interoperability – crosswalks – schema mapping – metadata – language resources and technology – linked data – visualization

[1]http://linkeddata.org/

Chapter 2

State of the Art

In this chapter we give a short overview of the development of large research infrastructures (with focus on those for language resources and technology), then we examine in more detail the hoist of work (methods and systems) on schema/ontology matching and review Semantic Web principles and technologies.

Note though that substantial parts of state of the art coverage are outsourced into separate chapters: A broad analysis of the data is provided in separate chapter 3 and a detailed description of the underlying infrastructure is found in 4.

2.1 Research Infrastructures (for Language Resources and Technology)

In recent years, multiple large-scale initiatives have set out to combat the fragmented nature of the language resources landscape in general and the metadata interoperability problems in particular.

EAGLES/ISLE Meta Data Initiative (IMDI) [2] 2000 to 2003 proposed a standard for metadata descriptions of Multi-Media/Multi-Modal Language Resources aiming at easing access to Language Resources and thus increases their reusability.

FLaReNet[1] – Fostering Language Resources Network – running 2007 to 2010 concentrated rather on "community and consensus building" developing a common vision and mapping the field of LRT via survey.

CLARIN – Common Language Resources and Technology Infrastructure – large research infrastructure providing sustainable access for scholars in the humanities and social sciences to digital language data, and especially its technical core the Component Metadata Infrastructure (CMDI) – a comprehensive architecture for harmonized handling of metadata[3] – are the primary context of this work, therefore the description of this underlying infrastructure is detailed in separate chapter 4. Both above-mentioned projects can be seen as predecessors to CLARIN, the IMDI metadata model being one starting point for the development of CMDI.

More of a sister-project is the initiative DARIAH - Digital Research Infrastructure for the Arts and Humanities[2]. It has a broader scope, but has many personal ties as well as similar problems and similiar solutions as CLARIN. Therefore there are efforts to intensify the cooperation between these two research infrastructures for digital humanities.

META-SHARE is another multinational project aiming to build an infrastructure for language resource[4], however focusing more on Human Language Technologies domain.[3]

[1] http://www.flarenet.eu/
[2] http://dariah.eu
[3] http://meta-share.eu

> META-NET is designing and implementing META-SHARE, a sustainable network of repositories of language data, tools and related web services documented with high-quality metadata, aggregated in central inventories allowing for uniform search and access to resources. Data and tools can be both open and with restricted access rights, free and for-a-fee.

See 3.2.5 for more details about META-SHARE's catalog and metadata format.

Digital Libraries

In a broader view we should also regard the activities in the domain of libraries and information sciences (LIS). Starting already in 1970's with connecting, exchanging and harmonizing their bibliographic catalogs, libraries were the early adopters and driving force in the field of search federation even before the era of internet, starting collaborative efforts in mid 70s (e.g. Linked Systems Project [5])

, they certainly have a long tradition, wealth of experience and stable solutions.

Driven mainly by national libraries still bigger aggregations of the bibliographic data are being set up. The biggest one is the Worldcat[4] (totalling 273.7 million records [6]) powered by OCLC, a cooperative of over 72.000 libraries worldwide.

In Europe, multiple recent initiatives have pursuit similar goals of pooling together the immense wealth of information sheltered in the many libraries: The European Library[5] offers a search interface over more than 18 million digital items and almost 120 million bibliographic records from 48 National Libraries and leading European Research Libraries.

Europeana[6] [7] is a cultural heritage initiative with even broader scope, serving as "meta-aggregator and portal for European digitised works", encompassing material not just from libraries, but also museums, archives and all other kinds of collections (In fact, The European Library is the *library aggregator* for Europeana).

A large number of projects contribute(d) to Europeana. E.g. the auxiliary project EuropeanaConnect[7] (2009-2011) delivered the core technical components for Europeana as well as further services reusable in other contexts, one of them being the spatio-temporal browser GeoTemCo[8] [8]. Most recently, with Europeana Cloud[9] (2013 to 2015) another initiative in the realm of Europeana has been started, a Best Practice Network, coordinated by The European Library, designed to "establish a cloud-based system for Europeana and its aggregators, providing new content, new metadata, a new linked storage system, new tools and services for researchers and a new platform - Europeana Research".

The related catalogs and formats are described in the section **??**

2.2 Existing crosswalks (services)

Crosswalks as list of equivalent fields from two schemas have been around already for a long time, in the world of enterprise systems, e.g. to bridge to legacy systems as well as in the LIS domain. [9] lists a number of mappings between metadata formats, mostly betweeen Dublin Core and MARC families of formats.[10]

[4] http://www.worldcat.org/
[5] http://www.theeuropeanlibrary.org/tel4/
[6] http://www.europeana.eu/
[7] http://www.europeanaconnect.eu/
[8] https://github.com/stjaenicke/GeoTemCo
[9] http://pro.europeana.eu/web/europeana-cloud
[10] http://loc.gov/marc/marc2dc.html, http://www.loc.gov/marc/dccross.html

However, besides being restricted in terms of covered formats, these crosswalks are just static correspondence lists, often just available as documents and only limited coverage of formats. One effort, that comes nearer to our idea of a service delivering crosswalks dynamically is the Metadata Crosswalk Service[11] offered by OCLC as part of Metadata Schema Transformation Services[12]

> a self-contained crosswalk utility that can be called by any application that must translate metadata records. In our implementation, the translation logic is executed by a dedicated XML application called the Semantic Equivalence Expression Language, or Seel, a language specification and a corresponding interpreter that transcribes the information in a crosswalk into an executable format.

Although the website states "Crosswalk Web Service is now a production system that has been incorporated into OCLC products and services", the demo service[13] is not accessible. Also, this service only offers crosswalks between formats relevant for the LIS community: Dublin Core, MARCXML, MARC-2709, MODS. So, altogether the service does not seem suitable to be used as is for the purposes of this work. But it certainly can serve as inspiration as for the specification of the planned service.

2.3 Schema/Ontology Mapping/Matching

As Shvaiko[10] states "*Ontology matching* is a solution to the semantic heterogeneity problem. It finds correspondences between semantically related entities of ontologies." As such, it provides a very suitable methodical foundation for the problem at hand – the *semantic mapping*. (In sections 5.6 and 6.2 we elaborate on the possible ways to apply these methods to the described problem.)

There is a plethora of work on methods and technology in the field of *schema and ontology matching* as witnessed by a sizable number of publications providing overviews, surveys and classifications of existing work [11, 12, 13, 14, 15] and most recently [10, 16].

Shvaiko and Euzenat also run the web page http://www.ontologymatching.org/ dedicated to this topic and the related OAEI[14], an ongoing effort to evaulate alignment tools based on various alignment tasks from different domains.

Interestingly, [10] somewhat self-critically asks if after years of research"the field of ontology matching [is] still making progress?".

Method

There are slight differences in use of the terms between [17, 18], [19] and [16], especially one has to be aware if in given context the term denotes the task in general, the process, the actual operation/function or the result of the function.

[19] formalizes the problem as "ontology matching operation":

> The matching operation determines an alignment A' for a pair of ontologies O1 and O2. Hence, given a pair of ontologies (which can be very simple and contain one entity each), the matching task is that of finding an alignment between these ontologies. [. . .]

[11] http://www.oclc.org/developer/services/metadata-crosswalk-service, http://www.oclc.org/research/activities/xwalk.html, (SOAP based)

[12] http://www.oclc.org/research/activities/schematrans.html?urlm=160118

[13] http://errol.oclc.org/schemaTrans.oclc.org.search

[14] Ontology Alignment Evalution Intiative - http://oaei.ontologymatching.org/

But basically the different authors broadly agree on the definition of *ontology alignment* in the meaning `task` is "to identify relations between individual elements of mulitple ontologies", or as `result` "a set of correspondences between entities belonging to the matched ontologies".

More formally [18] formulates ontology alignment as "a partial function based on the set E of all entities $e \in E$ and based on the set of possible ontologies O. [...] Once an alignment is established we say entity e is aligned with entity f when $align(e) = f$." Also, "alignment is a one-to-one equality relation." (although this is relativized further in the work, and also in [17])

Definition 2.1: *align* function

$$align : E \times O \times O \rightarrow E$$

[17] and [16] instead introduce *ontology mapping* when applying the task on individual entities, in the meaning as a function that "for each concept (node) in ontology A [tries to] find a corresponding concept (node), which has the same or similar semantics, in ontology B and vice verse". In the meaning as result it is "formal expression describing a semantic relationship between two (or more) concepts belonging to two (or more) different ontologies".

[17] further specify the mapping function as based on a similarity function, that for a pair of entities from two (or more) ontologies computes a ratio indicating the semantic proximity of the two entities.

Definition 2.2: *map* function for single entities and underlying *similarity* function

$$\begin{aligned}
&map : O_{i1} \rightarrow O_{i2} \\
&map(e_{i_1 j_1}) = e_{i_2 j_2}, \text{ if } sim(e_{i_1 j_1}, e_{i_2 j_2}) > t \text{ with } t \text{ being the threshold} \\
&sim : E \times E \times O \times O \rightarrow [0, 1]
\end{aligned}$$

This elegant abstraction introduced with the *similarity* function provides a general model that can accomodate a broad range of comparison relationships and corresponding similarity measures. And here, again, we encounter a broad range of possible approaches.

[20] lists a number of basic features and corresponding similarity measures: Starting from primitive data types, next to value equality, string similarity, edit distance or in general relative distance can be computed. For concepts, next to the directly applicable unambiguous `sameAs` statements, label similarity can be determined (again either as string similarity, but also broaded by employing external taxonomies and other semantic resources like WordNet - *extensional* methods), equal (shared) class instances, shared superclasses, subclasses, properties.

Element-level (terminological) vs structure-level (structural) [15]

based on background knowledge...

subclass–superclass relationships, domains and ranges of properties, analysis of the graph structure of the ontology.

For properties the degree of the super an subproperties equality, overlapping domain and/or range. Additionally to these measures applicable on individual ontology items, there are approaches (like the *Similarity Flooding algorithm* [21]) to propagate computed similarities across the graph defined by relations between entities (primarily subsumption hierarchy).

[22] classifies, reviews, and experimentally compares major methods of element similarity measures and their combinations. [10] comparing a number of recent systems finds that "semantic and extensional methods are still rarely employed. In fact, most of the approaches are quite often based only on terminological and structural methods.

[18] employs this *similarity* function over single entities to derive the notion of *ontology similarity* as "based on similarity of pairs of single entities from the different ontologies". This is operationalized as some kind of aggregating function[20], that combines all similiarity measures (mostly modulated by custom weighting) computed for pairs of single entities again into one value (from the *[0,1]* range) expressing the similarity ratio of the two ontologies being compared. (The employment of weights allows to apply machine learning approaches for optimization of the results.)

Thus, *ontology similarity* is a much weaker assertion, than *ontology alignment*, in fact, the computed similarity is interpreted to assert ontology alignment: the aggregated similarity above a defined threshold indicates an alignment.

As to the alignment process, [18] distinguishes following steps:

1. Feature Engineering
2. Search Step Selection
3. Similarity Assessment
4. Interpretation
5. Iteration

In contrast, [23] in their system LogMap2 reduce the process into just two steps: computation of mapping candidates (maximise recall) and assessment of the candidates (maximize precision), that however correspond to the steps 2 and 3 of the above procedure and in fact the other steps are implicitly present in the described system.

Systems

A number of existing systems for schema/ontology matching/alignment is collected in the above-mentioned overview publications:

IF-Map [24], QOM [20], FOAM [25], Similarity Flooding (SF) [?], S-Match [26], the Prompt tools [27] integrating with Protégé or COMA++ [28], Chimaera. Additionally, [10] lists and evaluates some more recent contributions: SAMBO, Falcon, RiMOM, ASMOV, Anchor-Flood, AgreementMaker.

All of the tools use multiple methods as described in the previous section, exploiting both element as well as structural features and applying some kind of composition or aggregation of the computed atomic measures, to arrive to a alignment assertion.

Next to OWL as input format supported by all the systems some also accept XML Schemas (COMA++, SF, Cupid, SMatch), some provide a GUI (COMA++, Chimaera, PROMPT, SAMBO, AgreementMaker).

Scalability is one factor to be considered, given that in a baseline scenario (before considering efficiency optimisations in candidate generation) the space of possible candidate mappings is the cartesian product of entities from the two ontologies being aligned. Authors of the (refurbished) ontology matching system LogMap 2 [23] hold that it implements scalable reasoning and diagnosis algorithms, performant enough, to be integrated with the provided user interaction.

2.4 Semantic Web – Linked Open Data

Linked Data paradigm[29] for publishing data on the web is increasingly been taken up by data providers across many disciplines [30]. [31] gives comprehensive overview of the principles of Linked Data with practical examples and current applications.

Semantic Web - Technical solutions / Server applications

The provision of the produced semantic resources on the web requires technical solutions to store the RDF triples, query them efficiently via SPARQL[?] and *idealiter* expose them via a web interface to the users.

Meanwhile a number of RDF triple store solutions relying both on native, DBMS-backed or hybrid persistence layer are available, open-source solutions like Jena, Sesame or BigData as well as a number of commercial solutions AllegroGraph, OWLIM, Virtuoso.

A qualitative and quantitative study[32] in the context of Europeana evaluated a number of RDF stores (using the whole Europeana EDM data set = 382,629,063 triples as data load) and came to the conclusion, that "certain RDF stores, notably OpenLink Virtuoso and 4Store" can handle the large test dataset.

OpenLink Virtuoso Universal Server[15] is hybrid storage solution for a range of data models, including relational data, RDF and XML, and free text documents.[33, 32] Virtuoso is used to host many important Linked Data sets, e.g., DBpedia[16] [34]. Virtuoso is offered both as commercial and open-source version license models exist.

Another solution worth examining is the Linked Media Framework[17] – "easy-to-setup server application that bundles together three Apache open source projects to offer some advanced services for linked media management": publishing legacy data as linked data, semantic search by enriching data with content from the Linked Data Cloud, using SKOS thesaurus for information extraction.

One more specific work is that of Noah et. al [35] developing a semantic digital library for an academic institution. The scope is limited to document collections, but nevertheless many aspects seem very relevant for this work, like operating on document metadata, ontology population or sophisticated querying and searching. Another solution in a related, more specialized domain and in already in productive use is rechercheisidore[18] [36], a french portal for digital humanities resources.

Ontology Visualization

Landscape, Treemap, SOM

AlViz - Multiple-View Visualization for Semi-Automatic Alignment of Ontologies

AlViz is a research prototype for visual ontology alignment implemented as multiple-view plug-in for Protege using J-Trees and Graphs. Based on similarity measures of an ontology matching algorithm AlViz helps to assess and optimize the alignment results

Protege

check Ontology Mapping and Alignement / saiks/Ontology4 4auf1.pdf

2.5 Language and Ontologies

There are two different relation links betwee language or linguistics and ontologies: a) 'linguistic ontologies' domain ontologies conceptualizing the linguistic domain, capturing

[15] http://virtuoso.openlinksw.com
[16] http://dbpedia.org
[17] http://code.google.com/p/lmf/
[18] http://rechercheisidore.fr

aspects of linguistic resources; b) 'lexicalized' ontologies, where ontology entities are enriched with linguistic, lexical information.

Linguistic ontologies

One prominent instance of a linguistic ontology is General Ontology for Linguistic Description or GOLD[37][19], that "gives a formalized account of the most basic categories and relations (the "atoms") used in the scientific description of human language, attempting to codify the general knowledge of the field. The motivation is to" facilite automated reasoning over linguistic data and help establish the basic concepts through which intelligent search can be carried out".

In line with the aspiration "to be compatible with the general goals of the Semantic Web", the dataset is provided via a web application as well as a dump in OWL format[20] [38].

Founded in 1934, SIL International[21] (originally known as the Summer Institute of Linguistics, Inc) is a leader in the identification and documentation of the world's languages. Results of this research are published in Ethnologue: Languages of the World[22] [39], a comprehensive catalog of the world's nearly 7,000 living languages. SIL also maintains Language & Culture Archives a large collection of all kinds resources in the ethnolinguistic domain [23].

World Atlas of Language Structures (WALS) [24] [40] is "a large database of structural (phonological, grammatical, lexical) properties of languages gathered from descriptive materials (such as reference grammars) ". First appeared 2005, current online version published in 2011 provides a compendium of detailed expert definitions of individual linguistic features, accompanied by a sophisticated web interface integrating the information on linguistic features with their occurrence in the world languages and their geographical distribution.

Simons [41] developed a Semantic Interpretation Language (SIL) that is used to define the meaning of the elements and attributes in an XML markup schema in terms of abstract concepts defined in a formal semantic schema Extending on this work, Simons et al. [42] propose a method for mapping linguistic descriptions in plain XML into semantically rich RDF/OWL, employing the GOLD ontology as the target semantic schema.

These ontologies can be used by ("ontologized") Lexicons refer to them to describe linguistic properties of the Lexical Entries, as opposed to linking to Domain Ontologies to anchor Senses/Meanings.

Work on Semantic Interpretation Language as well as the GOLD ontology can be seen as conceptual predecessor of the Data Category Registry a ISO-standardized procedure for defining and standardizing "widely accepted linguistic concepts", that is at the core of the CLARIN's metadata infrastructure (cf. 4.2.1). Although not exactly an ontology in the common sense of Although (by design) this registry does not contain any relations between concepts, the central entities are concepts and not lexical items, thus it can be seen as a proto-ontology. Another indication of the heritage is the fact that concepts of the GOLD ontology were migrated into ISOcat (495 items) in 2010.

Notice that although this work is concerned with language resources, it is primarily on the metadata level, thus the overlap with linguistic ontologies codifying the disci-

[19] http://linguistics-ontology.org
[20] http://linguistics-ontology.org/gold-2010.owl
[21] http://www.sil.org/about-sil
[22] http://www.ethnologue.com/
[23] http://www.sil.org/resources/language-culture-archives
[24] http://WALS.info

pline specific linguistic terminology is rather marginal (perhaps on level of description of specific linguistic aspects of given resources).

Lexicalised ontologies, "ontologized" lexicons

The other type of relation between ontologies and linguistics or language are lexicalised ontologies. Hirst [43] elaborates on the differences between ontology and lexicon and the possibility to reuse lexicons for development of ontologies.

In a number of works Buitelaar, McCrae et. al [44, 45, 46, 47, 48] argues for "associating linguistic information with ontologies" or "ontology lexicalisation" and draws attention to lexical and linguistic issues in knowledge representation in general. This basic idea lies behind the series of proposed models LingInfo, LexOnto, LexInfo and, most recently, lemon aimed at allowing complex lexical information for such ontologies and for describing the relationship between the lexicon and the ontology. The most recent in this line, lemon or lexicon model for ontologies defines "a formal model for the proper representation of the continuum between: i) ontology semantics; ii) terminology that is used to convey this in natural language; and iii) linguistic information on these terms and their constituent lexical units", in essence enabling the creation of a lexicon for a given ontology, adopting the principle of "semantics by reference", no complex semantic in- formation needs to be stated in the lexicon. a clear separation of the lexical layer and the ontological layer.

Lemon builds on existing work, next to the LexInfo and LIR ontology-lexicon models. and in particular on global standards: W3C standard: SKOS (Simple Knowledge Organization System) [49] and ISO standards the Lexical Markup Framework (ISO 24613:2008 [50]) and and Specification of Data Categories, Data Category Registry (ISO 12620:2009 [1])

Lexical Markup Framework LMF [51, 50] defines a metamodel for representing data in lexical databases used with monolingual and multilingual computer applications, provides a RDF serialization (?!?!).

An overview of current developments in application of the linked data paradigm for linguistic data collections was given at the workshop Linked Data in Linguistics[25] 2012 [52].

The primary motivation for linguistic ontologies like lemon are the tasks ontology-based information extraction, ontology learning and population from text, where the entities are often referred to by non-nominal word forms and with ambiguous semantics. Given, that the discussed collection contains mainly highly structured data referencing entities in their nominal form, linguistic ontologies are not directly relevant for this work.

2.6 Summary

This chapter concentrated on the current affairs/developments regarding the infrastructures for Language Resources and Technology and on the other hand gave an overview of the state of the art regarding methods to be applied in this work: Semantic Web Technologies, Ontology Mapping and Ontology Visualization.

[25] http://ldl2012.lod2.eu/

Chapter 3

Analysis of the data landscape

This section gives an overview of existing standards and formats for metadata in the field of Language Resources and Technology together with a description of their characteristics and their respective usage in the initiatives and data collections. Special attention is paid to the Component Metadata Framework representing the base data model for the infrastructure this work is part of.

3.1 Component Metadata Framework

The *Component Metadata Framework* (CMD) is the data model of the CLARIN Component Metadata Infrastructure. (See 4.2 for information about the infrastructure. The XML-schema defining CMD – the general-component-schema – is featured in appendix B.2.) CMD is used to define the so-called *profiles* being constructed out of reusable *components* – collections of metadata fields. The components can contain other components and they can be reused in multiple profiles. Profile itself is just a special kind of a component (a sub class), with some additional administrative information. The actual core provision for semantic interoperability is the requirement, that each CMD element (i.e. metadata field) refers "via a PID to exactly one data category[1] (cf. 4.2.1), thus indicating unambiguously how the content of the field in a metadata description should be interpreted" [53].

While the primary registry for data categories used in CMD is the ISOcat Data Category Registry (cf. 4.2.1), other authoritative sources are accepted (so-called "trusted registries"), especially the set of terms maintained by the Dublin Core Metadata Initiative [54].

Once the profiles are defined they are transformed into a XML-Schema, that prescribes the structure of the instance records. The generated schema also conveys as annotation the information about the referenced data categories.

3.1.1 CMD Profiles

In the CR 124[2] public Profiles and 696 Components are defined. Table 3.1 shows the development of the CR and DCR population over time.

Next to the 'native' CMD profiles a number of profiles have been created that implement existing metadata formats, like OLAC/DCMI-terms, TEI Header or the META-SHARE schema. The resulting profiles proof the flexibility/expressivity of the CMD metamodel. The individual profiles differ also very much in their structure – next to

[1]persistently referenceable concept definition

[2]All numbers are as of 2013-06 if not stated otherwise

Table 3.1: The development of defined profiles and DCs over time

date	2011-01	2012-06	2013-01	2013-06
Profiles	40	53	87	124
Distinct Components	164	298	542	828
Expanded Components	1055	1536	2904	5757
Distinct Elements	511	893	1505	2399
Expanded Elements	1971	3030	5754	13232
Distinct data categories	203	266	436	499
Data categories in the Metadata profile	277	712	774	791
Ratio of elements without DCs	24,7%	17,6%	21,5%	26,5%
Components with DCs	28	67	115	140

flat profiles with just one level of components or elements with 5 to 20 fields (*dublincore*, *collection*, the set of *Bamdes*-profiles) there are complex profiles with up to 10 levels (*ExperimentProfile*, profiles for describing Web Services) and a few hundred elements. The biggest single profile is currently the remodelled maximum schema from the META-SHARE project [55] for describing corpora, with 419 components and 1587 elements (when expanded[3]).

3.1.2 Instance Data

The main CLARIN OAI-PMH harvester[4] collects records from 69 providers on daily basis. The complete dataset amounts to 540.065 records. 16 of the providers offer CMDI records, the other 53 provide OLAC/DC records, that are being converted into the corresponding CMD profile after harvesting. Next to these 81.226 original OLAC records, there a few providers offering their OLAC or DCMI-terms records already converted into CMDI, thus all in all OLAC, DCMI-terms records amount to 139.152. On the other hand, some of the comparatively few providers of 'native' CMD records expose multiple profiles (e.g. Meertens Institute uses 12 different profiles.) So we encounter both situations: one profile being used by many providers and one provider using many profiles.

We can also observe a large disparity on the amount of records between individual providers and profiles. Almost half of all records is provided by the Meertens Institute (*Liederenbank* and *Soundbites* collections), another 25% by MPI for Psycholinguistics (*corpus* + *Session* records from the *The Language Archive*). On the other hand there are 25 profiles that have less than 10 instances. This can be owing both to the state of the respective project (resources and records still being prepared) and the modelled granularity level (collection vs. individual resource).

3.2 Other LRT Metadata Formats and Collections

Next to CLARIN and CMDI, there is a hoist of related previous and concurrent work. In the following, we briefly introduce some formats and data collections established in the field and, where applicable, we also sketch the ties with CMDI and existing integration efforts.

[3]The reusability of components results in an element expansion, i.e., elements of a component (e.g. *Contact*) included by three other components (*Project*, *Institution*, *Access*) will appear three times in the instantiated record.

[4]http://catalog.clarin.eu/oai-harvester/

Table 3.2: Top 20 CMD profiles, with the respective number of records

# records	*profile*
155.403	Song
138.257	Session
92.996	OLAC-DcmiTerms
46.156	DcmiTerms
28.448	SongScan
21.256	SourceScan
19.059	LiteraryCorpusProfile
16519	Source
13626	imdi-corpus
10610	media-session-profile
7961	SongAudio
7557	SymbolicMusicNotation
4485	LCC DataProviderProfile
4485	SourceProfile
4417	Text
1982	Soundbites-recording
1530	Performer
1475	ArthurianFiction
939	LrtInventoryResource
873	teiHeader

Table 3.3: Top 20 CMD collections, with the respective number of records

# records	*colleciton*
243.129	Meertens collection: Liederenbank
46.658	DK-CLARIN Repository
46.156	Nederlands Instituut voor Beeld en Geluid Academia collectie
29.266	childes
24.583	DoBeS archive
23.185	Language and Cognition
14.593	talkbank
14.363	Acquisition
14.320	Institut für Deutsche Sprache, CLARIN-D Zentrum, Mannheim
12.893	MPI CGN
10.628	Bavarian Archive for Speech Signals (BAS)
7.964	Pacific And Regional Archive for Digital Sources in Endangered Cultures
7.348	WALS RefDB
5.689	Lund Corpora
4.640	Oxford Text Archive
4.492	Leipzig Corpora Collection
3.539	Institut für Deutsche Sprache, CLARIN-D Zentrum, Mannheim
3.280	A Digital Archive of Research Papers in Computational Linguistics
3.147	CLARIN NL
3.081	MPI für Bildungsforschung

Some overview/survey works regarding existing formats are: The CLARIN deliverable *Interoperability and Standards* [56] provides overview of standards, vocabularies and other normative/standardization work in the field of Language Resources and Technology. And *Seeing standards: a visualization of the metadata universe* by Riley and Becker [57] putting the overwhelming amount of existing metadata standards into a systematic comprehensive overview analyzing the use of standards from four aspects: community, domain, function, and purpose. Though despite its aspiration on comprehensiveness it leaves out some of the formats relevant in the context of this work: IMDI, EDM, ESE, TEI???

3.2.1 Dublin Core metadata terms

The work on this metadata format started in 1995 at Metadata Workshop[5] organized by OCLC/NCSA in Dublin, Ohio, USA. Nowadays maintained by Dublin Core Metadata Initiative.

It is a fixed set of terms for a basic generic description of a range of resources (both virtual and physical) coming in two version[6]:

Dublin Core Metadata Element Set (DCMES) namespace: `/elements/1.1/`
the original set 15 terms, standardized as IETF RFC 5013, ISO Standard 15836-2009 and NISO Standard Z39.85-2007

Dublin Core metadata terms namespace: `/terms/`
the extended 'Qualified' set of 55 terms, extending the original 15 ones (replicating them in the new namespace for consistency)

The DCMI terms format is very widely spread nowadays. Thanks to its simplicity it is used as the common denominator in many applications, content management systems integrate Dublin Core to use in `meta` tags of served pages (`<meta name="DC.Publisher" content="publisher-name" >`), it is default minimal description in content repositories (Fedora-commons, DSpace). It is also the obligatory base format in the OAI-PMH protocol. The OpenArchives register[7] lists more than 2100 data providers.

There are multiple possible serializations, in particular a mapping t RDF is specified[8]. Worth noting is Dublin Core's take on classification of resources[9].

The simplicity of the format is also it's main drawback when considered as metadata format in the research communities. It it too general to capture all specific details, individual research groups need to describe different kinds of resources with.

3.2.2 OLAC

OLAC Metadata[10]format [58] is a application profile[59], of the Dublin Core metadata terms, adapted to the needs of the linguistic community. It is developed and maintained by the Open Language Archives Community providing a common platform and an infrastructure for "creating a worldwide virtual library of language resources" [60].

The OLAC schema [11] extends the dcterms schema mainly by adding attributes with controlled vocabularies, for domain specific semantic annotation (`linguistic-field, linguistic-type, language, role, discourse-type`)

[5]http://dublincore.org/workshops/dc1/
[6]http://dublincore.org/documents/dcmi-terms/
[7]http://www.openarchives.org/Register/BrowseSites
[8]http://dublincore.org/documents/dcq-rdf-xml/
[9]http://dublincore.org/documents/resource-typelist/
[10]http://www.language-archives.org/
[11]http://www.language-archives.org/OLAC/1.1/olac.xsd

> Uniform description across archives is ensured by limiting the values of certain metadata elements to the use of terms from agreed-upon controlled vocabularies. [...] OLAC adds encoding schemes that are designed specifically for describing language resources, such as subject language and linguistic data type.

Listing 3.1: Sample OLAC record

```
<olac:olac>
   <creator>Bloomfield, Leonard</creator>
   <date>1933</date>
   <title>Language</title>
   <publisher>New York: Holt</publisher>
</olac:olac>
```

OLAC provides a "search over 100,000 records collected from 44 archives[12], covering resources in half of the world's living languages".

Note, that OLAC archives are being harvested by CLARIN harvester and OLAC records are part of the CMDI joint metadata domain (cf. 3.2, 7.3.2).

3.2.3 TEI / teiHeader

> The Text Encoding Initiative (TEI) is a consortium which collectively develops and maintains a standard for the representation of texts in digital form ... [Next to] its chief deliverable is a set of Guidelines which specify encoding methods for machine-readable texts, chiefly in the humanities, social sciences and linguistics, ... the Consortium provides a variety of TEI-related resources, training events and software. [abgridged]

TEI is a de-facto standard for encoding any kind of digital textual resources being developed by a large community since 1994. It defines a set of elements to annotate individual aspects of the text being encoded. For the purposes of text description, metadata encoding (of main concern for us) the complex top-level element `teiHeader` is foreseen. TEI is not prescriptive, but rather descriptive, it does not provide just one fixed schema, but allows for a certain flexibility wrt to elements used and inner structure, allowing to generate custom schemas adopted to projects' needs. Thus there is also not just one fixed `teiHeader`.

Some of the data collections encoded in TEI are die Korpora des DWDS[13], Deutsches Textarchiv[14] [61], Oxford Text Archives[15]

There has been an intense cooperation between the TEI and CMDI community on the issue of interoperability and multiple efforts to express teiHeader in CMDI were undertaken (cf. 7.3.2) as a starting point for integrating TEI-based data into the CLARIN infrastructure.

3.2.4 ISLE/IMDI – The Language Archive

IMDI[16] (EAGLES/ISLE Meta Data Initiative) is an elaborate format for detailed descriptions of multi-media/multi-modal language resouces developed within the corresponding project[2] 2000 to 2003.

[12] http://www.language-archives.org/archives
[13] http://www.dwds.de
[14] http://www.dwds.de/dta
[15] http://ota.oucs.ox.ac.uk/
[16] http://www.mpi.nl/imdi/

To serve the main goal of the project, easing access to language resources fostering the reuse, resource description in this new format were created for a number of collections and were made available via a dedicated IMDI browser[17], that allowed browsing the collection structure as well as complex advanced search over the deeply structured metadata. Also a metadata editor was developed for generating records in this format, with provisions for offline field-work and synchronization with the repository.

The project lead and responsible for running the repository and whole infrastructure was the Technical Group at MPI for Psycholinguistics, who has engaged in a number of projects aimed at building a stable technical infrastructure for long-term archiving and work with language resources since its foundation (together with the Institute itself) in 1970s[18]. Recently, the group and the established infrastructure has been renamed to TLA – The Language Archive[19] "Your partner for language data, tools and archiving", where on one platform both the hoist of language resources and their description are preserved and provided as well as tools for working with this data is offered. The archive is also an aggregator itself, offering various collection from different (also external) projects (like DOBES, CGN, RELISH, etc.).

IMDI can be seen as predecessor of CMDI, the team of the TG being the driving force behind the development of both. A imdi-session profile, the corresponding IMDI to CMDI conversion as well as the transformed records were among the first to be added to the new CMD Infrastructure in 2010. The statistics of CMDI records list round 138.000 Session records and round 13.000 imdi-corpus records, modelling the collections for the sessions. Also, the metadata editor Arbil was refactored to work with the new data model.

3.2.5 META-SHARE

META-SHARE was the subproject (2010-2013) of META-NET, a Network of Excellence consisting of 60 research centres from 34 countries, that covered the technical aspects.

> META-SHARE is an open, integrated, secure and interoperable sharing and exchange facility for LRs (datasets and tools) for the Human Language Technologies domain and other applicative domains where language plays a critical role.

Within the project META-SHARE a new metadata format was developed[55]. Although inspired by the Component Metadata, META-SHARE metadata imposes a single large schema for all resource types with a subset of core obligatory elements and with many optional components.

The original META-SHARE schema actually accomodates four models for different resource types. Consequently, the model has been expressed as 4 CMD profiles each for a distinct resource type however all four sharing most of the components, as can be seen in figure 7.6. The biggest single profile is currently the remodelled maximum schema from the META-SHARE project for describing corpora, with 117 distinct components and 337 elements. When expanded, this translates to 419 components and 1587 elements. However, many of the components and elements are optional (and conditional), thus a specific instance will never use all the possible elements. (See 7.3.2 for more details about the format based on its integration into CMDI)

[17] http://corpus1.mpi.nl/ds/imdi_browser/
[18] http://tla.mpi.nl/home/history/
[19] http://tla.mpi.nl/

The technical infrastructure of META-SHARE represents a distributed network of repositories consists of a number of member repositories, that offer their own subset of resource[20].

Selected member repositories[21] play the role of managing nodes providing "a core set of services critical to the whole of the META-SHARE network"[4], especially collecting the resource descriptions from other members and exposing the aggregated information to the users. The whole network offers approximately 2.000 resources (the numbers differ even across individual managing nodes).

One point of criticism from the community was, the fact, that META-SHARE infrastructure does not provide any interface to the outer world, such as a OAI-PMH endpoint.

3.2.6 ELRA

European Language Resources Association[22] ELRA, offers a large collection of language resources, mostly under license for a fee, although some resources are available for free as well. The available datasets can be search for via ELRA Catalog[23] Additionally ELRA runs the so-called Universal Catalog – a repository comprising information regarding Language Resources (LRs) identified all over the world.

> ELRA's missions are to promote language resources for the Human Language Technology (HLT) sector, and to evaluate language engineering technologies.
>
> ELDA[24] - Evaluations and Language resources Distribution Agency – is ELRA's operational body, set up to identify, classify, collect, validate and produce the language resources which may be needed by the HLT – Human Language Technology – community.
>
> ELDA handles the practical and legal issues related to the distribution of language resources, provides legal advice in the field of HLT, and drafts and concludes distribution agreements on behalf of ELRA.

3.2.7 LDC

Linguistic Data Consortium[25] hosted by University of Pennsylvania is another provider/aggregator of high quality curated language resources. The data is provided for a fee, more than 650 resources have been made available since 1993. The catalog is freely accessible. The metadata is additionally aggregated by OLAC archives.

3.3 Formats and Collections in the World of Libraries

There are at least two reasons to concern ourselves with the developments in the world of Libraries and Information Systems (LIS): the long tradition implying rich experience and the fact, that almost all of the resources in the libraries are language resources. This argument gets even more relevant in the light of the efforts to digitize large portions of the material pursued in many (national) libraries in the last years (cf. discussion on Libraries partnering with Google). And given the amounts of data, even only the bibliographic records constitute sizable language resources in they own right.

[20]http://www.meta-share.eu/
[21]7 as of 2013-07
[22]http://elra.info
[23]http://catalog.elra.info/
[24]http://www.elda.org/
[25]http://www.ldc.upenn.edu/

3.3.1 Formats – MARC, METS, MODS

There is a long tradition of standardized metadata formats in the world of Libraries and Information Systems (LIS), major role in the standardization being assumed for decades by the Library of Congress[26].

The MARC[27] set of formats (being used since 1970s) "are standards for the representation and communication of bibliographic and related information in machine-readable form". A number of variants developed over the years, the most widely spread is MARC 21 since 1999 – is the standard format used for communication among libraries around the world.

MARC 21 consists of 5 "communication formats" for specific types of data (Bibliographic, Authority Data, Holdings Data, Classification, and Community Information), are widely used standards for the representation and exchange of bibliographic, authority, holdings, classification, and community information data in machine-readable form. In 2002, the Library of Congress developed the MARCXML schema for representing MARC records in XML;

METS – Metadata Encoding and Transmission Standard - a format from the family of Library of Congress standards (since 2001) for encoding descriptive, administrative, and structural metadata regarding objects within a digital library. It is dedicated primarily to capture the structure of the digital objects, "record the various relationships that exist between pieces of content, and between the content and metadata that compose a digital library object" [62]. A METS record acts as a flexible container that accomodates other pieces of data (different levels of metadata and encoded objects themselves or references to those) in external formats[28].

Number of tools have been developed to author and process METS format[29] and numerous projects (online editions, DAM systems) use METS for structuring and recording the data[30] among others also austrian literature online[31]

Metadata Object Description Schema - "is a schema for a bibliographic element set that may be used for a variety of purposes, and particularly for library applications". It is a simplified subset of MARC 21 using language-based tags rather than numeric ones, more than Dublin Core. One of endorsed schemas to extend (be used inside) METS.

There were efforts to create a conceptually more sound base for the bibliographic data – in 1998 Functional Requirements for Bibliographic Records (FRBR) [63] was published, an abstract model for the data expressed as an Entity Relationship Model and a standard based on FRBR, the Resource Description and Access (RDA) has been proposed as an comprehensive standard for resource description and discovery, that however it was confronted with opposition from the LIS community, questioning the need of abandoning established cataloging practices [**?**]. And although there is still work on RDA, among others by the Library of Congress, there has been no wider adoption of the standard by the LIS community until now.

3.3.2 ESE, Europeana Data Model - EDM

Within the big european initiative Europeana (cf. 2.1) information about digitised objects are collected from a great number of cultural institutions from all of Europe, currently

[26] http://www.loc.gov/standards/
[27] www.loc.gov/marc/
[28] http://www.loc.gov/standards/mets/mets-extenders.html
[29] http://www.loc.gov/standards/mets/mets-tools.html
[30] http://www.loc.gov/standards/mets/mets-registry.html though seems rather outdated
[31] http://www.loc.gov/standards/mets/mets-registry.html

originally developed and advised the common format ESE Europeana Semantic Elements[32] a Dublin Core-based application profile[33]. Soon it became obvious, that this format is very limiting and work started on a Semantic Web compatible RDF-based format – the Europeana Data Model EDM[34] [64, 65, 66]. EDM is fully compatible with ESE, which is (and will be) accepted from the providers. There is also already a SPARQL endpoint[35] to explore the Europeana data in the new format.

3.4 Controlled Vocabularies, Reference Data, Ontologies

One goal of this work being the groundwork for exposing the discussed dataset in the Semantic Web one preparatory task is to identify external semantic resources like controlled vocabularies or ontologies that the dataset could be linked with[36].

Conceptually, we want to partition these resources in two types. On the one hand abstract concepts constituting all kinds of classifications, typologies, taxonomies. On the other hand named entities that exist(ed) in real world, like persons, organizations or geographical places. Main motivation for this distinction is the insight, that while for named entities there is (mostly) "something" in the (physical) world that gives a solid ground for equivalence relations between references from different sources (sameAs), for concepts we need to accept a plurality of existing conceptualizations and while we can (and have to) try to identify relations between them, the equivalence relation is inherently much weaker. This insight entails a partly different approach – simply put, while we can aspire to create one large list/index encompassing all named entities, we have to maintain a forest of conceptual trees.

In the following we inventarize such resources, covering the domains expected to be needed for linking the original dataset. (Information about size of the dataset is meant rather as a rough indication of the "general weight" of the dataset, not necessarily a precise up to date information.) The acronyms in the tables are resolved in the subsequent glossary. How this resources will be employed is discussed in 6.2. Additionally, some verbose commentary follows.

The largest controlled vocabularies of named entities are the authority files of (national) libraries. These are further aggregated into the so-called Virtual International Authority File, a huge resource, with entries from different authority files referring to the same entity being merged. This resource can be explored via a search interface and there is also a search service for applications. Other general large-scale resources are the vocabularies curated and provided by Getty Research Institute[37], however there is only a limited free access and licensed and fee for full access. But recently there work was announced to publish the vocabularies as LOD[38]

Regarding existing domain-specific semantic resources `LT-World`[39], the ontology-based portal covering primarily Language Technology being developed at DFKI[40], is a prominent resource providing information about the entities (Institutions, Persons, Projects, Tools, etc.) in this field of study. [67]

[32]http://pro.europeana.eu/ese-documentation
[33]www.europeana.eu/schemas/ese/ESE-V3.4.xsd
[34]http://pro.europeana.eu/edm-documentation
[35]http://europeana.ontotext.com/sparql
[36]Similar activity of inventarizing vocabularies and thesauri was conducted in the context of the Europeana initiative http://europeanalabs.eu/wiki/WP12Vocabularieshttps://europeanalabs.eu/wiki/DesignSemanticThesauri
[37]http://www.getty.edu/research/tools/vocabularies/index.html
[38]http://www.getty.edu/research/tools/vocabularies/lod/index.html
[39]http://www.lt-world.org/
[40]Deutsches Forschungszentrum für Künstliche Intelligenz, http://www.dfki.de

Also to mention Yago, a large knowledge base created by MPI informatik integrating dbpedia, geonames and wordnet[41] [68].

So we witness a strong general trend towards Semantic Web and Linked Open Data.

[41]http://www.mpi-inf.mpg.de/yago-naga/yago/

Table 3.4: Controlled vocabularies of named entities – Persons, Organizations, Works, Language Names, Geographica

name	*provider*	*size (items / facts)*	*description*	*access*
VIAF	OCLC + NatLibs	$\gg$ 1E7	union of national authority files	search service, search app
GND/p	DNB	4.6E6	Persons, universal, lang:de	GND ontology
GND/k	”	1.2E6	Organizations, universal, lang:de	
GND/w	”	193,000	Works, lang:de	
GND/g	”	293.000	Geographica, lang:de	
ULAN	Getty	202,720 / 638,900	persons, artists	
TGN	Getty	992.310 / 1.7E6	also historical place names	web search
dbpedia	Wikipedia	$\sim$ 4E6	all kinds of entities in up to 111 langs	data dumps, live SPARQL endpoint
		764,000 persons; 333,000 works; 192,000 organizations; 639,000 geographica		
Yago [68]	MPI Informatik	1E7 / 1.2E8	huge semantic KB (aggregated from Wikipedia, Wordnet, Geonames)	data dumps
LT-World	DFKI	3.300 persons, 4.600 organizations	ontology-based portal for Language Technology	portal
Geonames	Geonames	>1E7 (2.8E6 / 5.5E6)	”modern” place names	data dump + web service
PKND	prometheus	>37,000	persons, artists	XML dump
iDAI.gazetteer	DAI		archaeologically relevant places	search interface
Pleiades		34.000	A community-built gazetteer and graph of ancient places	CSV, KML and RDF data dumps
LCCN	LoC	>1.2E7	identifier for bibliographic records	search service, search app
ISO 3166	ISO	249	Official country codes, lang: en, fr	
ISO-639-1	ISO	185	basic language codes	static list
ISO-639-3	SIL	$\sim$ 7.679	3-letter code for every human language	view/download
CLAVAS	CLARIN	2.500	organization names extracted from CMD records	OpenSKOS – search service

Table 3.5: Taxonomies, Classifications, Thesauri

name	*provider*	*size (items / facts)*	*description*	*access*
AAT	Getty	34,880 / 245,530	subjects in art and architecture	
LCSH	LoC		subjects, universal	FAST (Faceted Application of Subject Terminology), Linked Data FAST
LCC	LoC		universal hierarchical classification	web app: classification web
GND/s	DNB	202.000	subjects (Schlagwörter), universal, lang:de	
GTAA	NISL	3.800	Subjects, describing TV programs	(RDF) data dumps, OpenSKOS – search service
DDC	OCLC		universal classification by field of study, translated in multiple languages	dewey.info
UDC				
Wiki Categories	Wikipedia	995,911	classification of Wiki articles as skos:Concepts	SKOS Vocabulary, SPARQL
DBpedia Ontology	Wikipedia	529 / 2333	general classification of Wiki articles as ontology	RDF data, SPARQL
ISOcat	(CLARIN)	>6,500	data categories defining (linguistic) concepts in a number of thematic groups (Metadata, Lexical Resources, ...)	web-app, service
Object Names Thesaurus	British Museum		classification of objects in the collection	
Material Thesaurus	British Museum		classification of material	
Thesaurus of Monument Types	British Museum		types of monuments	
Hornbostel-Sachs-Systematik		300 categories	classification of musical instruments	web page
Oberbegriffsdatei	DMB		a set of vocabularies for museums, lang:de	`museumsvokabular.de`, PDF, XML dumps
Iconclass	RKD	28,000	taxonomy of subject of an image	RDF dump
DiRT	Project Bamboo	32 categories	taxonomy of research tools (1,200 tools)	

AAT international Architecture and Arts Thesaurus, Getty

CONA Cultural Objects Name Authority

DAI Deutsches Archäologisches Institut

DDC Dewey Decimal Classification

DFKI Deutsches Forschungszentrum für Künstliche Intellligenz

DMB Deutscher Museumsbund

DNB Deutsche National Bibliothek

FAST Faceted Application of Subject Terminology

Getty Getty Research Institute curating the vocabularies[42], part of Getty Trust

GND *Gemeinsame Norm Datei* - Integrated authority Files of the German National Library

GTAA Gemeenschappelijke Thesaurus Audiovisuele Archieven (Common Thesaurus [for] Audiovisual Archives)

> The thesaurus consists of several facets for describing TV programs: subjects; people mentioned; named entities (Corporation names, music bands etc); locations; genres; makers and presentators.

ISO International Standardization Organization

LCCN Library of Congress Control Number

LCC Library of Congress Classification

LCSH Library of Congress Subject Headings

LoC Library of Congress[43]

OCLC Online Computer Library Center[44] – world's biggest library federation

PKND prometheus KünstlerNamensansetzungsDatei[45]

RKD Rijksbureau voor Kunsthistorische Documentatie – Netherlands Institute for Art History

TGN Getty Thesaurus of Geographic Names

UDC Universal Decimal Classification

ULAN Union List of Artist Names

VIAF Virtual International Authority File – union of the authority files of >20 national (and prominent research) libraries

[42] http://www.getty.edu/research/tools/vocabularies/index.html
[43] http://loc.gov
[44] http://www.oclc.org
[45] http://prometheus-bildarchiv.de/de/tools/pknd

3.5 Summary

In this chapter, we gave an overview of the existing formats and datasets in the broad context of Language Resources and Technology. We also gave an overview of main formats and collections in the domain of Library and Information Services and a inventory of existing controlled vocabularies for named entities and concepts (taxonomies, classifications), needed as input in section 6.2 about mapping values to entities.

Chapter 4

Underlying infrastructure

In this chapter, we present the infrastructure, in which this work is embedded. We start with a short general introduction about the large research infrastructure initiative CLARIN, followed by a close examination of its technical infrastructure for creating and publishing metadata. In section 4.3, we discuss the services for managing controlled vocabularies and their role in the context of metadata creation.

4.1 CLARIN

CLARIN - Common Language Resource and Technology Infrastructure[69] - is one of the large research infrastructure initiatives as envisaged by the European Stategy Forum on Research Infrastructures (ESFRI) and fostered by the framework programmes of the European Commission. The mission of this project is to provide

> ...easy and sustainable access for scholars in the humanities and social sciences to digital language data (in written, spoken, video or multimodal form) and advanced tools to discover, explore, exploit, annotate, analyse or combine them, independent of where they are located.[70]

The initiative foresees a federated network of centres providing resources and services in a harmonized, interoperable manner to the academic community in all participating countries.

In the preparation phase of the project 2008 - 2011 over 180 institutions from 38 countries participated. In the construction phase, the action impetus moved, as projected, more to the individual national initiatives of this federated endeavour, while kept together by the common principles set up during the preparation phase and established processes and administrative decision bodies ensuring the flow of information and coherent action on European level.

Since 2013, CLARIN also became an *European Research Infrastructure Consortium* (ERIC), which is a new type of legal entity established within EU, especially designed to give the research infrastructure initiatives a more stable status and better means to act independently. This is an important step to ensure a continuity of the endeavour, the chronic problem of (international) projects.

4.2 Component Metadata Infrastructure – CMDI

One core pillar of CLARIN is the *Component Metadata Infrastructure* (CMDI)[1] – a distributed system consisting of multiple interconnected modules aimed at creating and

[1] http://www.clarin.eu/cmdi

providing metadata for LRT in a coherent harmonized way. The conceptual foundation of CMDI is the *Component Metadata Framework*[53], a flexible meta model that supports creation of metadata schemas also allowing to accommodate existing schemas (cf. 3.1).

The SMC is part of CMDI and depends on multiple modules on the production side of the infrastructure. Before we describe the SMC and its interaction with these modules in detail in chapter 5, we introduce the latter and the type of data they provide in 4.2.1:

- Data Category Registry
- Component Registry
- Relation Registry

All these modules are running services, that this work shall directly build upon.

In contrast, SMC is meant as provider for the modules on the exploitation side of the infrastructure, i.e. search and exploration services used by the end users. These are briefly introduced in 4.2.3.

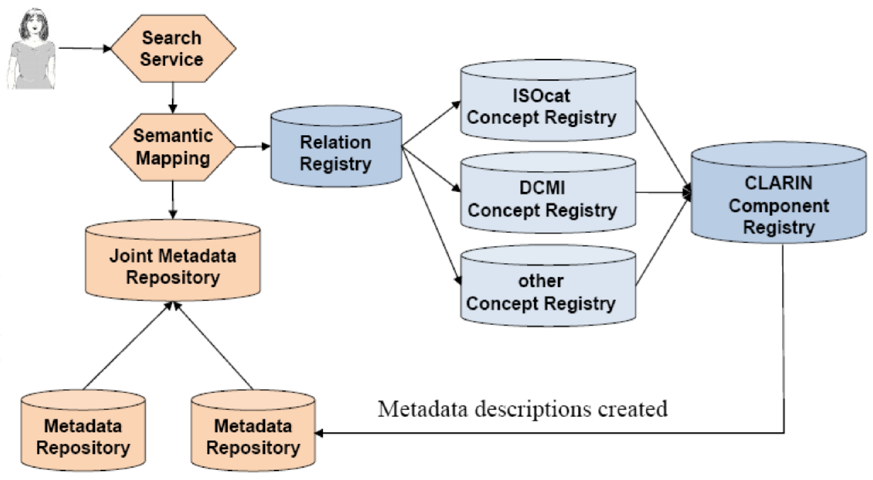


Figure 4.1: The diagram [from early CLARIN/CMDI presentations] shows individual modules of the CMDI and their interrelations as envisaged in the initial phase of the CLARIN project

Next to the above-mentioned services SMC is in direct interaction with, some other services and applications are part of the CMDI ecosystem that are briefly introduced in 4.2.2 for completeness:

- metadata editors
- Schema Registry
- SchemaParser

Finally, the Vocabulary Alignment Service, a module playing crucial role in metadata curation, is treated separately in section 4.3.

4.2.1 CMDI registries

The CMD framework as data model (cf. 3.1) together with the two registries the *Data Category Registry* ISOcat and the *Component Registry* build the backbone of the CMD

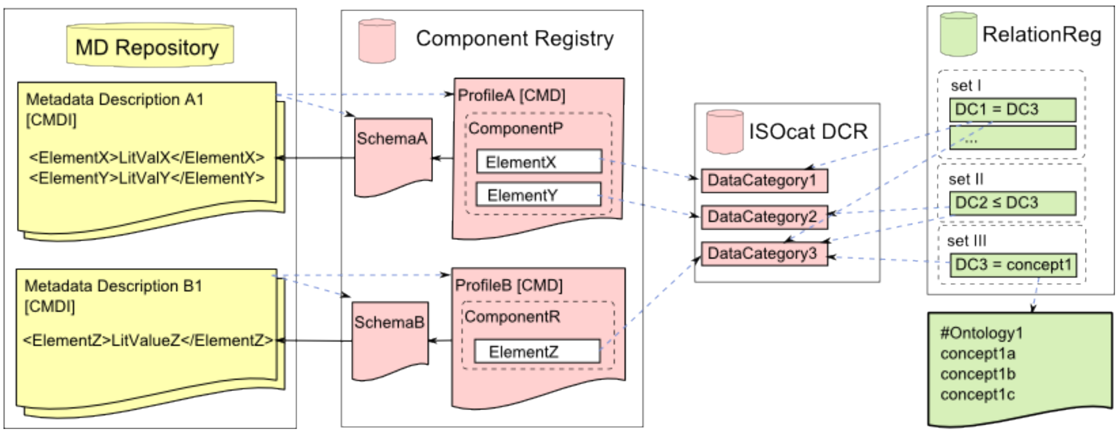


Figure 4.2: The diagram depicts the links between pieces of data in the individual registries that serve as basis for semantic mapping

Infrastructure. See figure 4.1 with the rather naïve initial vision of the system contrasted with the figure 4.2 detailing the actual linkage between the data in the individual registries. In the following, we explain briefly their role and interaction.

Data Category Registry – ISOcat

The *Data Category Registry* (DCR) is a central registry that enables the community to collectively define and maintain a set of relevant linguistic data categories (DC). The resulting shared controlled vocabulary is the cornerstone for grounding the semantic interpretation within the CMD framework (among others – DCR is not specific to CMDI, it is meant to be used as common concept registry in many applications).

The data model and the procedures of the DCR are defined by the ISO standard [1]. ISOcat[2] is an implementation of this standard framework developed by MPI for Psycholinguistics, Nijmegen in collaboration with the ISO technical committee ISO TC 37 Terminology and Other Language and Content Resources. Next to a web interface for users to browse and manage the data categories, ISOcat provides a REST-style webservice allowing applications to retrieve the data category specifications. By default, it is provided in the Data Category Interchange Format - DCIF, the standardized XML-serialization of the data model, but a RDF and HTML representation is available as well.

The core data model defining the data category specification is rather complex, consisting of administrative, linguistic and description part, containing language-specific versions of definitions, value domains, examples and other attributes (cf. B.1 for the diagram of the full data model). Following types of data categories are recognized (cf. figure 4.3): *simple, complex*: (*closed, open* or *constrained*), *container*. One fundamental aspect to emphasize is, that the data categories are assigned a persistent identifier, making them globally and permanently referable.

Component Registry

Component Registry[3] (CR) implements the CMD data model (cf. 3.1) and fulfills two functions. For one, it is the actual registry that persistently stores and exposes published CMD profiles via a web interface allowing to browse and search in them and view their structure accompanied by a REST webservice to allows client applications to retrieve

[2] http://www.isocat.org/
[3] http://catalog.clarin.eu/ds/ComponentRegistry/

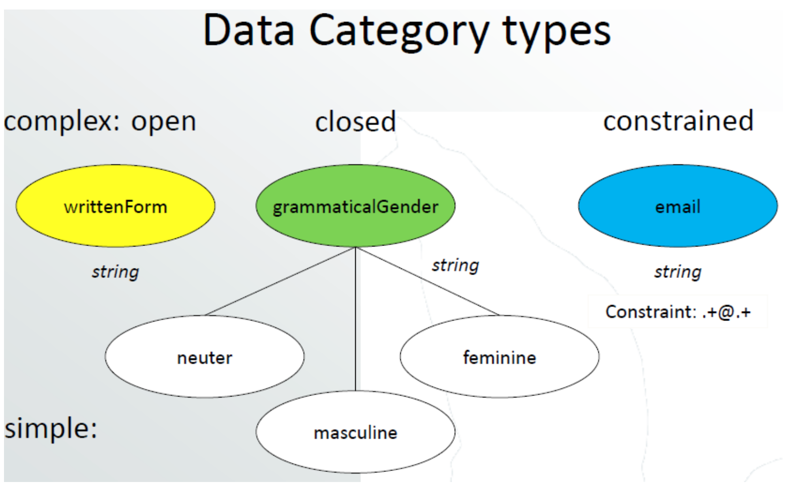


Figure 4.3: Data Category types[**?**]

the profile definitions. At the same time the web interface serves as an editor for creating and editing new CMD components and profiles.

The primary user of the CR is the metadata modeller with the task to create a dedicated metadata profile for a given resource type. She can browse and search the CR for components and profiles that are suitable or come close. The registry already contains many general components, e.g., for contact persons, language and geographical information. In general many of these can be reused as they are or have to be only slightly adapted, i.e., have some metadata elements and/or components added or removed. Also new components can be created if needed to model the unique aspects of the resources under consideration.[71]

Let us reiterate, that the actual core provision for semantic interoperability is the requirement that the elements (and as far as possible also components and values) should be linked "via a PID to exactly one data category (cf. 4.2.1), thus indicating unambiguously how the content of the field in a metadata description should be interpreted"[53], or *to make its semantics explicit*.

As dictated by the CMD model, all components needed for the modelled resource description are compiled into one profile. Once a profile is created, the Component Registry provides automatically the corresponding XML schema in the `cmd` target namespace `http://www.clarin.eu/cmd`, that can be used as base for creating and validating metadata records.

Ontological Relations – Relation Registry

The framework as described so far provides a sound mechanism for binding the semantic interpretation of the metadata descriptions. However there needs to be an additional means to capture information about relations between data categories. This information was deliberately not included in the DCR, because relations often depend on the context in which they are used, making global agreement unfeasible. CMDI proposes a separate module – the *Relation Registry* (RR) [72] –, where arbitrary relations between data categories can be stored and maintained. This design decision is based upon the assumption that the relations be under control of the metadata user whereas the data categories are under control of the metadata modeller.

The relations don't need to pass a standardization process, but rather separate research teams may define their own sets of relations according to the specific needs of the project. That is not to say that every researcher has to create her own set of relations – some basic recommended sets will be defined right from the start. But new – even contradictory – ones can be created when needed.

There is a prototypical implementation of such a relation registry called RELcat being developed at MPI, Nijmegen[73, 74], that already hosts a few relation sets. There is no user interface to it yet, but it is accessible as a REST-webservice[4]. This implementation stores the individual relations as RDF triples allowing typed relations, like equivalency (`rel:sameAs`) and subsumption (`rel:subClassOf`). The relations are grouped into relation sets that can be used independently. The relations are deliberately defined in a separate namespace, instead of reusing existing ones (`skos:exactMatch, owl:sameAs`) with the aim to avoid introducing too specific semantics. These relations can be mapped to appropriate other predicates when integrating the relation sets in concrete applications.

Definition 4.1: The relation triples as stored by the Relation Registry

$$< subjectDatcat\ relationPredicate\ objectDatcat>$$

4.2.2 Further parts of the infrastructure

Schema Registry

SCHEMAcat[5] is a registry for schemas of all kinds (not just the CMD-based, in fact not even just XML-based) semantically annotated with data categories.

> RELcat and SCHEMAcat will provide the means to harvest and specify this information in the form of relationships and allow (search) algorithms to traverse the semantic graph thus made explicit[75].

Schema Parser

Schema Parser is a service developed at the Meertens Institute, Amsterdam, that processes XML Schemas to generate all possible paths in the instance data. It is used primarily as auxiliary service to the search engine developed at the same institute, presented in the following subsection.

Metadata editors

Metadata creation, i.e. the authoring of actual metadata records is undisputably the fundamental task in the whole system. Though not directly interacting with SMC, metadata editors need to be mentioned, i. e. tools that the human metadata editors is using for authoring metadata.

Given that the Component Registry generates a XML schema for every profile, basically any generic XML editor with schema validation can be used (e.g. the widespread oXygen). However, there have been efforts within the CLARIN community to develop dedicated tools, tailor-made for creation of CMD records. Two examples being the stand-alone application Arbil[76][6] being developed at Max Planck Institute for Psycholinguistics, Nijmegen and the web-based application developed within the project NaLiDa[77][7] at the Seminar für Sprachwissenschaft University Tübingen.

[4]sample relation set: http://lux13.mpi.nl/relcat/rest/set/cmdi
[5]http://lux13.mpi.nl/schemacat/site/index.html
[6]http://tla.mpi.nl/tools/tla-tools/arbil/
[7]http://www.sfs.uni-tuebingen.de/nalida/en/

4.2.3 CMDI exploitation side

Metadata complying with the CMD data model is being created by a growing number of institutions by various means – automatic transformation from legacy data or authoring of new metadata records with the help of one of the metadata editors (cf. 4.2.2). The CMD infrastructure requires the content providers to publish their metadata via the OAI-PMH protocol and announce the OAI-PMH endpoints. These are being collected daily by a dedicated CLARIN harvester[8]. The harvested data is validated against the corresponding schemas (every profile implies a separate schema). In the future a subsequent normalization step will play a bigger role, currently only minimal ad-hoc label normalization is performed for a few organization names. Finally, the data is made (publicly) available as compressed archive files. These are being fetched by the exploitation side applications, that ingest the metadata records, index them and make them available for searching and browsing (cf. figure 4.4).

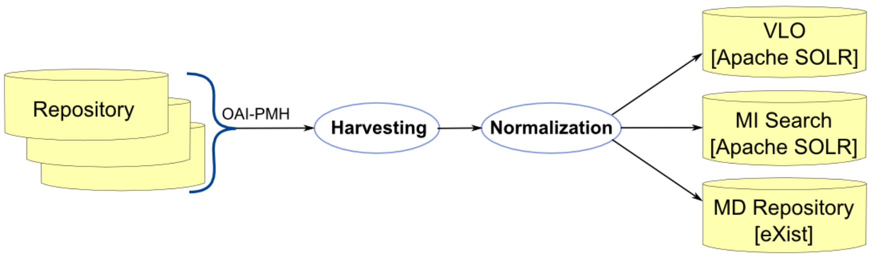


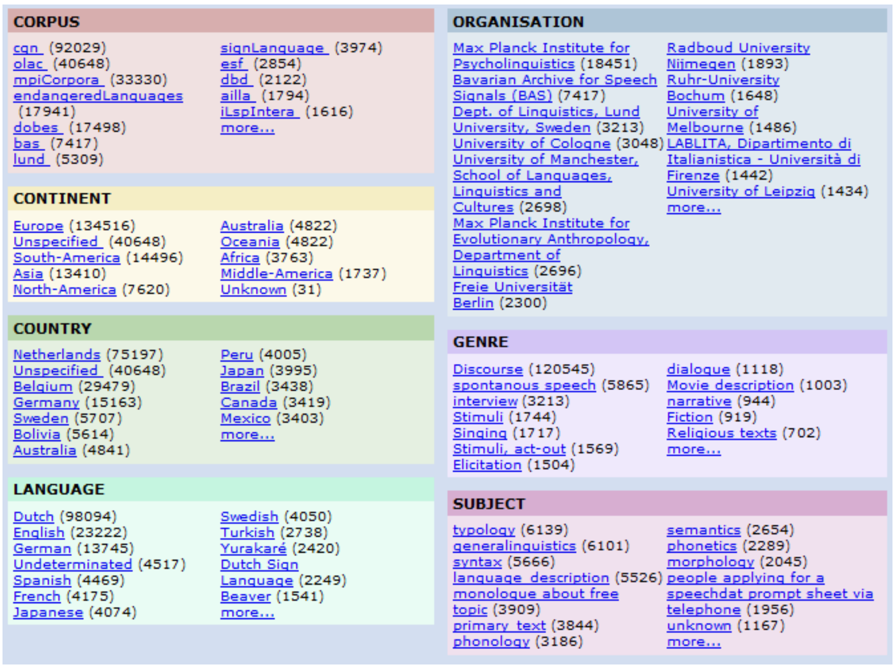
Figure 4.4: Within CMDI, metadata is harvested from content providers via OAI-PMH and made available to consumers/users by search applications

The first stable and publicly available application providing access to the collected metadata of CMDI has been the VLO - Virtual Language Observatory[9][78], developed by the Technical Group at the MPI for Psycholinguistics, Nijmegen, based on the widespread full-text search engine Apache Solr[10]. The application employs a faceted search with 10 fixed facets (figure 4.5). As the processed metadata records are instances of different CMD profiles and thus have very differing structures, to map the fields in the records onto the facets the application relies on the data category references in the underlying schemas, effectively making use of this basic layer of semantic interoperability provided by the infrastructure.

More recently, the team at Meertens Institute developed a similar application the `MI Search Engine`[11]. It too is based on the Apache Solr and provides a faceted search, but with a substantially more sophisticated indexing process and search interface [**?**]. Instead of reducing the data into a fixed number of indexes or facets, the application employs the aforementioned Schema Parser to dynamically generate an index configuration that covers all data, again relying on the data categories to merge information from semantically equivalent metadata fields in the different schemas into a common index. The application also offers some innovative solutions on the user interface, like search by similarity, content-first search or specialized contextual widgets visualizing the time dimension, the geographic information and other derived data.

And finally, there is the Metadata Repository, being developed by the author as a XQuery application in the XML database eXist, originally (in the initial blueprints of

[8] http://catalog.clarin.eu/oai-harvester/
[9] http://www.clarin.eu/vlo/
[10] http://lucene.apache.org/solr/
[11] http://www.meertens.knaw.nl/cmdi/search/



CORPUS

cqn (92029)
olac (40648)
mpiCorpora (33330)
endangeredLanguages (17941)
dobes (17498)
bas (7417)
lund (5309)
signLanguage (3974)
esf (2854)
dbd (2122)
ailla (1794)
iLspIntera (1616)
more...

CONTINENT

Europe (134516)
Unspecified (40648)
South-America (14496)
Asia (13410)
North-America (7620)
Australia (4822)
Oceania (4822)
Africa (3763)
Middle-America (1737)
Unknown (31)

COUNTRY

Netherlands (75197)
Unspecified (40648)
Belgium (29479)
Germany (15163)
Sweden (5707)
Bolivia (5614)
Australia (4841)
Peru (4005)
Japan (3995)
Brazil (3438)
Canada (3419)
Mexico (3403)
more...

LANGUAGE

Dutch (98094)
English (23222)
German (13745)
Undeterminated (4517)
Spanish (4469)
French (4175)
Japanese (4074)
Swedish (4050)
Turkish (2738)
Yurakaré (2420)
Dutch Sign Language (2249)
Beaver (1541)
more...

ORGANISATION

Max Planck Institute for Psycholinguistics (18451)
Bavarian Archive for Speech Signals (BAS) (7417)
Dept. of Linguistics, Lund University, Sweden (3213)
University of Cologne (3048)
University of Manchester, School of Languages, Linguistics and Cultures (2698)
Max Planck Institute for Evolutionary Anthropology, Department of Linguistics (2696)
Freie Universität Berlin (2300)
Radboud University Nijmegen (1893)
Ruhr-University Bochum (1648)
University of Melbourne (1486)
LABLITA, Dipartimento di Italianistica - Università di Firenze (1442)
University of Leipzig (1434)
more...

GENRE

Discourse (120545)
spontanous speech (5865)
interview (3213)
Stimuli (1744)
Singing (1717)
Stimuli, act-out (1569)
Elicitation (1504)
dialogue (1118)
Movie description (1003)
narrative (944)
Fiction (919)
Religious texts (702)
more...

SUBJECT

typology (6139)
generalinguistics (6101)
syntax (5666)
language description (5526)
monologue about free topic (3909)
primary text (3844)
phonology (3186)
semantics (2654)
phonetics (2289)
morphology (2045)
people applying for a speechdat prompt sheet via telephone (1956)
unknown (1167)
more...

Figure 4.5: screenshot of the faceted browser of the VLO

the infrastructure) foreseen as main storage of the collected metadata with the Metadata Service on top providing search access to the data optionally applying Semantic Mapping to expand user queries (cf. figure 4.1). [79] However the application still did not reach production quality, and is used rather as experimenting field for the author. Meanwhile the functionality of the Metadata Service had been integrated directly into the Metadata Repository together with the auxiliary use of Semantic Mapping, making it the implementation of the semantic search module as proposed in this work (cf. 5.4).

4.3 Vocabulary Service / Reference Data Registries

4.3.1 Motivation & broader context

The provisions for data harmonization and semantic interoperability as presented until now pertain mostly to the schema level. However the problem of incoherent labeling and nomenclature is even more virulent in the actual metadata fields on the instance level. While for a number of fields the value domain can be enforced through schema validation, many fields (e.g. organization or resource type) have a constrained value domain that yet cannot be explicitly exhaustively enumerated. This leads to a chronically inconsistent use of labels for referring to entities (as the instance data shows, some organizations are referred to by more than 20 different labels, or spelling variants.) prompting an urgent need for better means for harmonizing the constrained-field values.

This issue is to be seen in a broader context of a general need for reliable community-shared registry services for concepts, controlled vocabularies and reference data in both the LRT and Digital Humanities community, applicable in a range of applications and tasks like data enrichment and annotation, metadata generation and curation, data analysis, etc. Moreover, by using global semantic identifiers instead of strings, such a service enables the harmonization of metadata descriptions and annotations and is an indispensable step towards transformation of this data into *Linked Open Data*.

Consequently, activities with regard to controlled vocabularies are ongoing not only in CLARIN, but also within the sister ESFRI project DARIAH. As there is a substantial overlap in the vocabularies relevant for the various communities and even more so a high potential for reusability on the technical level, there is a strong case for tight synergic cooperation between individual initiatives.

It has to be also kept in mind, that a hoist of work on controlled vocabularies has already been done and a large body of data is present in individual specialized communities (taxonomies) as well as – with more general scope – in the libraries world (authority files).

4.3.2 Implementation – OpenSKOS/CLAVAS

In the context of CLARIN (primarily CLARIN-NL), a concrete initiative has been conducted – Vocabulary Alignment Service for CLARIN or CLAVAS – with the objective to reuse and enhance for CLARIN needs a SKOS-based vocabulary repository and editor OpenSKOS[12], developed and run within the dutch program CATCHplus[13].

The basic idea of this repository is to serve as a project independent manager and provider of controlled vocabularies, as an exchange platform for data in SKOS format. One important feature of the OpenSKOS system is its distributed architecture. Multiple instances can be set up, that can synchronize the maintained vocabularies among each other via OAI-PMH protocol. This caters for a reliable redundant system, in which multiple instances provide identical synchronized data, with organizations behind individual instances assuming the primary responsibility for individual vocabularies based on their specialization or field of expertise.

Currently, the Meertens Institute[14] of the Dutch Royal Academy of Sciences (KNAW), Netherlands Institute for Sound and Vision[15], as well as Austrian Centre for Digital Humanities at the Austrian Academy of Sciences are running a instance of the OpenSKOS system.

As the work on this vocabulary repository started in the context of a cultural heritage program, originally it served vocabularies not directly relevant for the LRT-community GTAA - Gemeenschappelijke Thesaurus Audiovisuele Archieven or AAT - Art & Architecture Thesaurus[16]. Within the CLAVAS, a number of vocabularies relevant for the CLARIN and LRT-community were identified, that will be gradually integrated into the vocabulary repository. (See 3.4 for a more complete list of required reference data together with candidate existing vocabularies.) Following vocabularies were already integrated into the CLAVAS instance of OpenSKOS:

- the list of language codes[80]
- organization names for the domain of language resources
- a number of data categories from ISOcat (see 4.3.3 for details of the process)

4.3.3 Export DCR to SKOS

Based on the premise, that the data in DCR also represents a kind of a controlled vocabularies, there is an effort to export data categories in SKOS format and import them into the Vocabulary Service.

Note, that there are two interaction paths between the ISOcat and the Vocabulary Service. The first, importing certain data categories from ISOcat into the Vocabulary

[12]http://openskos.org
[13]*Continuous Access To Cultural Heritage* - http://www.catchplus.nl/en/
[14]http://meertens.knaw.nl/
[15]http://www.beeldengeluid.nl/
[16]http://openskos.org/api/collections

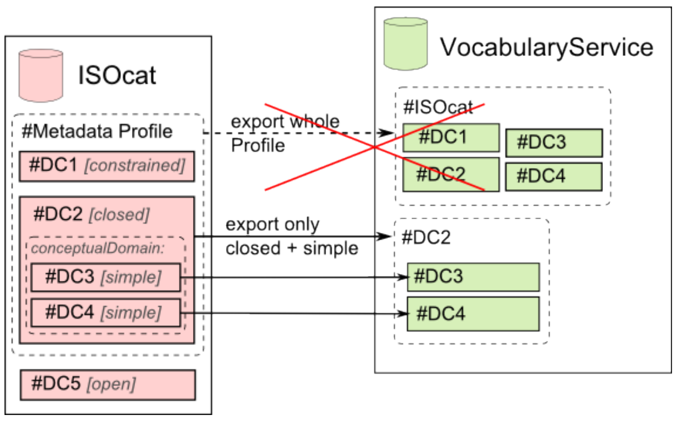


Figure 4.6: The wrong and correct variant of exporting ISOcat data categories in SKOS format to the Vocabulary Service

Service, is described in this section. The second aspect (described in next section 4.3.4) is, that the value domains of certain data categories are defined by reference to a vocabulary maintained in the Vocabulary Service.

The fact that data categories are basically definitions of concepts may mislead to a naïve approach to mapping DCR data to SKOS, namely mapping every data category to a `skos:Concept` all of them belonging to the `ISOcat:ConceptScheme`. However the data in ISOcat as whole is too disparate in scope for such a vocabulary to be useful.

A more sensible approach is to export only closed DCs (with explicitely defined value domain, cf. 4.2.1) as separate `skos:ConceptSchemes` and their respective simple DCs as `skos:Concepts` within that scheme.

> The rationale is, that if we see a vocabulary as a set of possible values for a field/element/attribute, complex DCs in ISOcat are the users of such vocabularies and simple DCs the DCR equivalence of values in such a vocabulary.[81]

Another aspect is, that a simple DC can be in value domains of multiple closed DCs. Also a `skos:Concept` can belong to multiple `skos:ConceptSchemes`[17]. So there could a 1:1 mapping [complex closed DCs] to [skos:ConceptSchemes] and [simple DCS] to [skos:Concepts]. That would automatically convey also the possibly multiplicate membership of simple DCs / skos:Concepts in closed DCs / skos:ConceptSchemes.

Alternatively, for each value domain a SKOS concept scheme with SKOS concepts can be created, i.e., a SKOS concept always belongs to one concept schema, but multiple SKOS concepts refer to the same simple DC using `<dcr:datcat/>` (and `<dcterms:source/>`). This is, how the export for CLAVAS currently works.[18][19]

4.3.4 Linking to vocabularies in data categories and schemas – interaction between ISOcat, CLAVAS and client applications

In the following, we elaborate on the possible ways to model references to vocabularies in data category specification and to convey that information to the client application.

[17] http://www.w3.org/TR/skos-primer/#secscheme
[18] http://www.isocat.org/rest/profile/5.clavas
[19] https://trac.clarin.eu/browser/cats/ISOcat/trunk/mod-ISOcat-interface-rest/representations/dcs2/clavas.xsl

As of the writing, this is work in progress with some design decision yet to be made.[20]

Providing vocabularies for constrained but large and complex conceptual domains is the main motivation for the vocabulary repository:

> Originally, the vocabulary repository has been conceived to manage rather large and complex value domains, that do not fit easily in the DCR data model. Where the value domains are big (ISO 639-3) or can only be partially enumerated (organization names) ISOcat can't/shouldn't contain the value domains but just refer to CLAVAS, i.e., ISOcat wouldn't be a provider.[81]

Currently, the only possibility to constrain the value domain of a data category is by the means a XML Schema provides, like enumeration or regular expression. So for the data category languageID#DC-2482 the rule looks like:

```
<dcif:conceptualDomain type="constrained">
       <dcif:dataType>string</dcif:dataType>
       <dcif:ruleType>XML Schema regular expression</dcif:ruleType>
       <dcif:rule>[a-z]{3}</dcif:rule>
</dcif:conceptualDomain>
```

A proposal by Windhouwer[81] for integration with CLAVAS foresees following extension:

```
<clavas:vocabulary href="http://my.openskos.org/vocab/ISO-639" type="closed"/>
```

> `@href` points to the vocabulary. Actually a PID should be used in the context of ISOcat, but it is not clear how persistent are the vocabularies. This may pose a problem as part of DC specification may now have a different persistency then the core.
>
> `@type` could be `closed` or `open`. `closed`: only values in the vocabulary are valid. `open`: the values in the vocabulary are hints/preferred values. Basically the DC itself is then open.

This yields a definition of the value domain for the data category, where the new rule pointing to the vocabulary is *added* (cf. listing 4.1), so that – once the information from the DC specification gets into the schema – tools that don't support vocabulary lookup but are capable of XSD/RNG validation, can still use the regular expression based definition.

Listing 4.1: definition of conceptualDomain for the data category languageID employing the proposed extension for pointing to a vocabulary

```
<dcif:conceptualDomain type="constrained">
   <dcif:dataType>string</dcif:dataType>
   <dcif:ruleType>XML Schema regular expression</dcif:ruleType>
   <dcif:rule>[a-z]{3}</dcif:rule>
</dcif:conceptualDomain>
<dcif:conceptualDomain type="constrained">
   <dcif:dataType>string</dcif:dataType>
   <dcif:ruleType>CLAVAS vocabulary</dcif:ruleType>
    <dcif:rule>
       <clavas:vocabulary href="http://my.openskos.org/vocab/ISO-639" type="closed"/>
    </dcif:rule>
</dcif:conceptualDomain>
```

[20] Large parts of this subsection come from email correspondence with M. Windhouwer in spring 2013.[81]

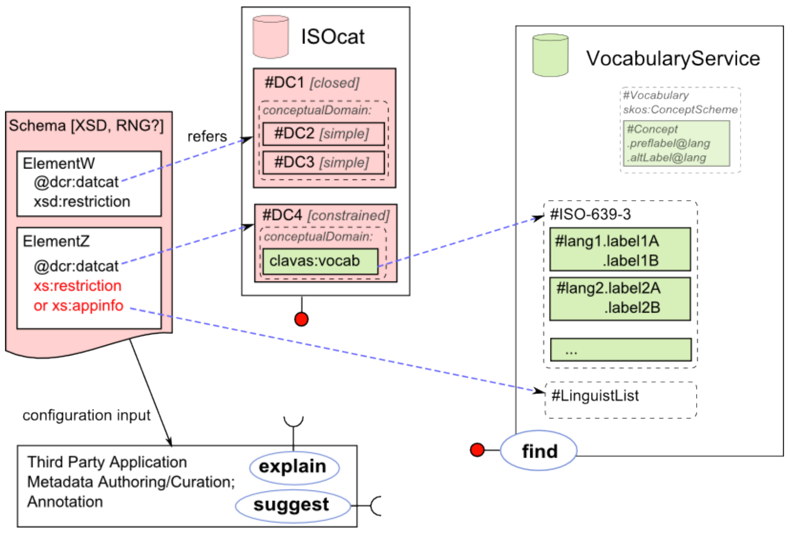


Figure 4.7: The linking between schemas, data categories and vocabularies

It is important to emphasize, that anything stated in the DC specification is not binding (even if the DC is of type *closed*), but rather a non-normative hint or recommendation. The authoritative source is the schema. A schema modeller binding an element in a schema to a data category can still decide to have other restriction for the values domain of that element then the ones suggested in the DC specification. This applies equally to the proposed vocabulary reference mechanism: The author of the data category suggests a vocabulary to be used for values of given data category, but the metadata modeller decides, if and how this vocabulary will be integrated into the modelled schema.

There are basically two options, how the vocabulary can be integrated into the schema. One approach is to explicitly enumerate all the values from the vocabulary. Within CMD this has been done in the component for language-codes[21]. This method allows to strictly validate given metadata field, however there is clearly a limit to this approach in terms of a) size of the vocabulary[22], b) completeness – most of the vocabularies cannot be seen as closed, i.e. they represent only a partial enumeration just providing a recommended label for an entity, and c) stability or change rate – even the supposedly fixed list of language-codes ISO-639-* undergoes regular changes – it is being updated semi-annually, with entries being added, deleted, merged and split.[23]

The other "soft" alternative is to convey the information about data category and vocabulary in the schema as annotation, either in `<xs:app-info>` element or by some attribute in dedicated namespace. This method is already being employed in the Component Registry indicating data category of a generated element with the `@dcr:datcat` attribute.

Once the data category and vocabulary reference end up in the specification of the CMD profile and the derived XSD, the information can finally be used by client applications (like metadata editor)[24]. The tool can use the reference to the data category to fetch explanations (semantic information) (and translations) from ISOcat and it can access the autocomplete/search interface of the Vocabulary Service to offer the user

[21] http://catalog.clarin.eu/ds/ComponentRegistry/?item=clarin.eu:cr1:c_1271859438110
[22] e.g. ISO-639 contains 7.679 items (language codes) adding some 2MB to each schema referencing it
[23] http://www-01.sil.org/iso639-3/changes.asp
[24] Note though, that this is not a standard mechanism but rather a convention. The client application must implement it in order to be able to make use of it.

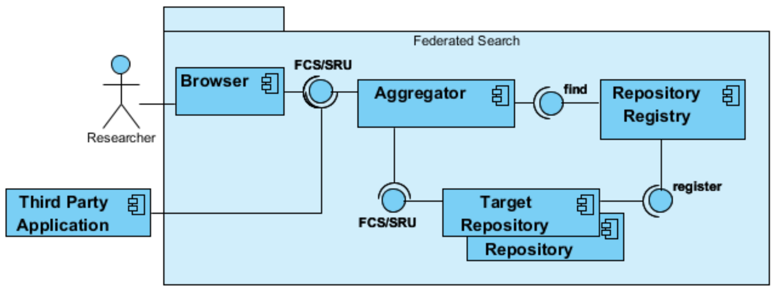


Figure 4.8: Components of the Federated Content Search and their interdependencies

suggestions from the recommended vocabulary (cf. figure 4.7).

The drawback of this variant is, that we gave up the validation. This isn't a problem if the vocabulary is of `@type=open`, e.g. organisation names, but it is when the value domain is closed, e.g. languageId. In the latter case, the XSD generation could support both modes: a lax (smaller) version which doesn't contain the closed vocabulary as an enumeration and leaves it to the tool, and a strict version which does contain the vocabulary as an enumeration. Probably the latter should stay the default, but the client application could request the lax version leading to smaller and quicker XSD validation inside the tool.

4.4 Other aspects of the infrastructure

While this work concentrates solely on the metadata, it needs to be recognized, that it is only aspect of the infrastructure and its actual purpose the availability of resources. Metadata is a necessary first step to announce and describe the resources. However it is of little value, if the resources themselves are not accessible. Consequently, another pillar of the CLARIN infrastructure are the centres[25]:

> CLARIN's distributed network is made out of centres. These units, often a university or an academic institute, offer the scientific community access to services on a sustainable basis.

CLARIN imposes a number of criteria, that each centre needs to fulfill to become a CLARIN Centre[26][82]. CLARIN also maintains a central registry, the Centre Registry[27], maintaining structured information about every centre, meant as primary entry point into the CLARIN network of centres.

One core service of such centres are the content repositories, systems meant for long-term preservation and online provision of research data and resources. A number of centres have been identified that provide Depositing Services[28], i.e. allow third parties researchers (not just the home users) to store research data.

Another aspect of the availability of resources is, that while metadata can be harvested and indexed locally in one repository, this is not possible with the content itself, both due to the size of the data, but mainly due to legal obligations (licenses, copyright), restricting the access to and availability of the resources. CLARIN's answer to this problem is the task force *Federated Content Search*[29] [83] aiming at establishing

[25]http://www.clarin.eu/node/3812
[26]http://www.clarin.eu/node/3767
[27]https://centerregistry-clarin.esc.rzg.mpg.de/
[28]http://clarin.eu/3773
[29]http://www.clarin.eu/fcs

an architecture allowing to search simultaneously (via an aggregator) across a number of resources hosted by different content providers via a harmonized interface adhering to a common protocol. The agreed upon protocol is a compatible extension of the SRU/CQL protocol developed and endorsed by the Library of Congress as the XML- (and web)based successor of the Z39.50 [85].

Note that in practice the line between metadata and content data is not so clear – usually there is a need to filter by metadata even when searching in content. Therefore also most content search engines feature some kind of metadata filters.

4.5 Summary

In this chapter we presented individual parts of the infrastructure, next to the core registries: ISOcat Data Category Registry, Component Registry and Relation Registry, that this work directly builds upon, a number of other services and application forming the CLARIN ecosystem were briefly introduced. A separate consideration was dedicated to the issue of controlled vocabularies together with a related module the Vocabulary Alignment Service (and its implementation OpenSKOS) that allows to manage vocabularies and use them in client application. Finally a few other aspects of the infrastructure, that are equally important, however not pertaining to the metadata level, were briefly tackled.

Chapter 5

System design – concept-based mapping on schema level

In this chapter, we define the main function of the proposed system – the **concept-based crosswalk and search functionality** – the tasks that the Semantic Mapping Component was originally conceived for within the larger CMD Infrastructure (cf. 4.2). Additionally we explore the related aspect of analytic visualization of the processed data.

We start by drawing an overall view of the system, introducing its individual components and the dependencies among them. In the next section, the internal data model is presented and explained. In section 5.3 the design of the actual main service for serving crosswalks is described, divided into the interface specification and notes on the actual implementation. In section 5.4 we elaborate on a search functionality that builds upon the aforementioned service in terms of appropriate query language, a search engine to integrate the search in and the peculiarities of the user interface that could support this enhanced search possibilities. Finally, in section 5.5 an advanced interactive user interface for exploring the CMD data domain is proposed.

5.1 System Architecture

The SMC module is part of the CMD Infrastructure. It is a consumer of data from the production-side registries and serves search services on the exploitation side of the infrastructure, as well as third party applications accessing the joint CLARIN metadata domain.

The SMC module can be broken down into following components:

crosswalk service the basic service translating between fields (or indexes), detailed in 5.3.1

concept-based query expansion a module for query expansion based on the crosswalks

smc-xsl set of xslt-stylesheets (governed by a build-file) for pre- and post-processing the data

SMC Browser a web application to explore the CMD data domain consisting of the two modules: smc-stats and smc-graph

smc-stats a module of the SMC Browser providing human-readable statistical summaries of the CMD data domain

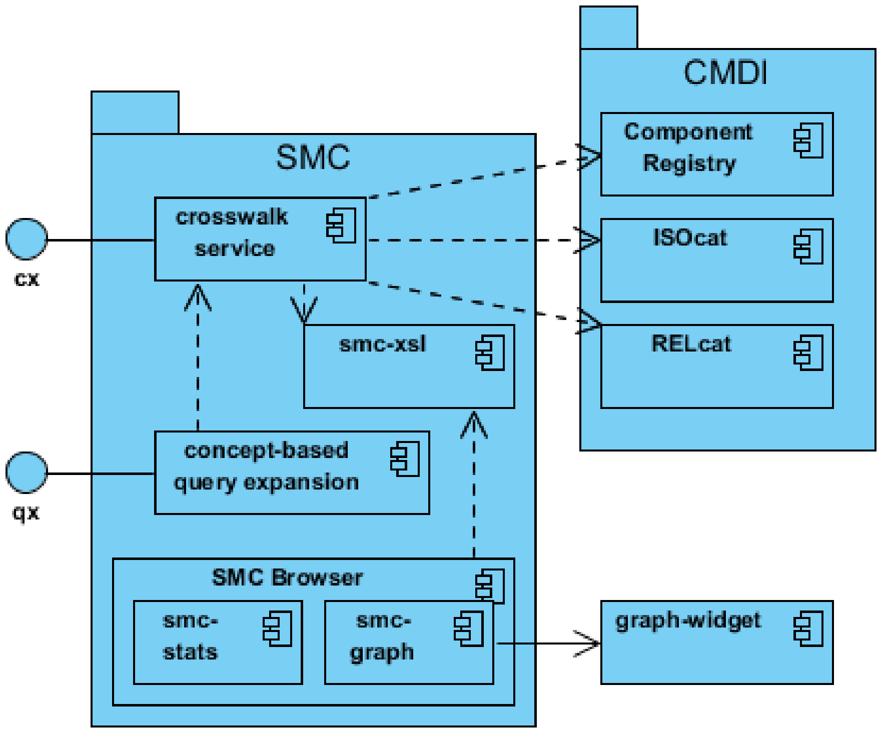


Figure 5.1: The component view on the SMC - modules and their inter-dependencies

smc-graph a module of the SMC Browser providing advanced interactive graph-based user interface for exploring the CMD data domain

The component diagram in 5.1 depicts the dependencies between the components of the system. The crosswalk service uses the set of XSL-stylesheets smc-xsl and accesses the CMDI registries: Component Registry, ISOcat DCR and RELcat to retrieve the data. It exposes an interface cx to be used by third party applications. The query expansion module uses the crosswalk service to rewrite queries, also exposing a corresponding API qx.

SMC Browser consists of two parts the smc-stats and smc-graph and also uses the set of stylesheets for processing the data. smc-graph is build on top of a library for interactive visualization of graphs.

For broader context see the reference architecture diagram in Figure B.2.

5.2 Data model

Before we get to the definition of the actual service, we define the internal data model, divided into of two parts:

smcIndex a data type for denoting indexes in a human-readable way used internally and as input and output format of the service

Terms.xsd the schema for internal representation of the processed data

5.2.1 smcIndex

In this section, we describe *smcIndex* – the data type to denote indexes used by the components of the system internally, as well as input and output on the interfaces.

Definition 5.1: Grammar of *smcIndex*

$$
\begin{aligned}
smcIndex &::= dcrIndex \mid cmdIndex \\
dcrIndex &::= dcrID\ contextSep\ datcatLabel \\
&\quad \mid [\ dcrID\ contextSep\ ]\ datcatID \\
cmdIndex &::= profile \\
&\quad \mid cmdEntityId \\
&\quad \mid [\ profile\ contextSep\ ]\ dotPath \\
profile &::= profileName\ [\ \texttt{\#}\ profileID\ ] \\
dotPath &::= [\ dotPath\ pathSep\ ]\ elemName \\
cmdEntityId &::= componentId\ [\ \texttt{\#}\ elemName\ ] \\
contextSep &::= \text{`}\texttt{.}\text{`} \mid \text{`}\texttt{:}\text{`} \\
pathSep &::= \text{`}\texttt{.}\text{`} \\
dcrId &::= \text{`}\texttt{isocat}\text{`} \mid \text{`}\texttt{dc}\text{`}
\end{aligned}
$$

An *smcIndex* is a human-readable string adhering to a specific syntax, denoting a search index. The syntax is based on two main ideas drawn from existing work: a) denoting a context by a prefix is derived from the way indices are referenced in CQL-syntax[1] (analogous to the XML-namespace mechanism, cf. 5.4.1), e.g. dc.title and b) on the dot-notation used in IMDI-browser[2] to denote paths into structured data (analogous to XPath), e.g. Session.Location.Country. The grammar generates only single terms, that may not contain whitespaces.

The grammar distinguishes two main types of *smcIndex*: a) *dcrIndex* referring to data categories and b) *cmdIndex* denoting a specific "CMD entity", i.e. an element (metadata field), component or whole profile defined within CMD (cf. 3.1 for description of the CMD data model). These two types of *smcIndex* follow different construction patterns. *cmdIndex* has a recursive path-like structure and can be interpreted as a XPath-expression into the instances of CMD profiles. In contrast to it, *dcrIndex* consists of just one-level term and is generally not directly applicable on existing data. It can be understood as abstract index referring to well-defined concepts – the data categories – and for actual search it needs to be resolved to the set of CMD elements it is referred by. In return, one can expect to match more metadata fields from multiple profiles, all referring to the same data category.

It is important to note that in general *smcIndex* can be ambiguous, meaning it can refer to multiple concepts, or CMD entities. This is due to the fact that the labels of the data categories and CMD entities are not guaranteed unique. Although it may seem problematic and undesirable to have an ambiguous reference, this is an intentional design decision. The labels are needed for human-readability and ambiguity can be useful, as long as one is aware of it. However there needs to be also the possibility to refer to data categories or CMD entities unambiguously. Therefore, the syntax also allows to reference indexes by the corresponding identifier. Following are some explanations to the individual constituents of the grammar:

dcrID is a shortcut referring to a data category registry. Next to ISOcat, other registries can function as a DCR, in particular, the dublincore set of metadata terms.

[1]Context Query Language, http://www.loc.gov/standards/sru/specs/cql.html
[2]http://www.lat-mpi.eu/tools/imdi

datcatLabel is the human-readable name of given data category (e.g. telephoneNumber). In the case of ISOcat data categories the verbose descriptor `mnemonicIdentifier` is used. However despite its name, it is not guaranteed unique. Therefore, *datcatID* has to be used if a data category shall be referenced unambiguously. For dublincore terms no such distinct identifier and label exist, the concepts are denoted by the lexical term itself, which is unique within the dublincore namespace.

profile is reference to a CMD profile. Again, it can be either the name of the profile *profileName* or – for guaranteed unambiguous reference – its identifier *profileId* as issued by the Component Registry (e.g. *clarin.eu:cr1:p_1272022528363* for LexicalResourceProfile). Even if a profile is referenced by its identifier it may and should be prefixed by its name to still ensure human-readability. Or, seen the other way round, the name is disambiguated by suffixing it with the identifier:

```
LexicalResourceProfile#clarin.eu:cr1:p_1272022528363
LexicalResourceProfile#clarin.eu:cr1:p_1290431694579
```

dotPath allows to address a leaf element (Session.Actor.Role), or any intermediary XML element corresponding to a CMD component (Session.Actor) within a metadata description. This allows to easily express search in whole components, instead of having to list all individual fields. The paths don't need to start from the root entity (the profile), they can reference any subtree structure. However longer paths are often needed for more specific references, e.g. instead of Name one could say Actor.Name vs. Project.Name or even Session.Actor.Name vs. Drama.Actor.Name. Still this mechanism does not guarantee unique references, it only allows to specify context and thus narrow down the semantic ambiguity.

5.2.2 Terms

Here we describe the XML schema for internal representation of the processed data. In abstract terms, the internal format is basically a table with information about indexes collected from the upstream registries or created during preprocessing. `Term` is main entity that represents either a label of a data category, or a CMD entity (a CMD component or element). `Termset` represents a logical collection of `Terms` (one profile or data categories of one type). `Concept` represents a data category and groups all corresponding terms. `Relation` is used to express relation between two `Concepts`. In the following, we explain the data model of these entities and their use in more detail. For a full Terms.xsd XML schema see listing B.1.

Type `Term`

`Term` is a polymorph data type, that can have different sets of attributes depending on the type of data it represents.

Table 5.1: Attributes of `Term` when encoding data category

attribute	allowed values	sample value
concept-id	PID given by DCR	`isocat:DC-2522`
set	identifier of the DCR *dcrID*	`isocat`
type	one of ['id', 'label', 'mnemonic']	`id`, `label`
xml:lang	two-letter language code (only for ISOcat)	`en`, `si`

Table 5.2: Attributes of `Term` when encoding CMD entity

attribute	allowed values	sample value
id	*cmdEntityId* as defined in 5.2.1	`clarin.eu:cr1:c_1290431694487#Url`
type	`CMD_Element` or `CMD_Component`	`CMD_Element`
datcat	reference to the data category, URL or *dcrIndex*	`isocat:DC-2546`
name	name of the component or element	`Url`
path	*dotPath* (cf. 5.2.1)	`SpeechCorpus.Access.Contact.Url`
parent	name of the parent component	`Contact`

Table 5.3: Attributes of `Term` when encoding a CMD entity in the inverted index

attribute	allowed values	sample value
id	*cmdEntityId* cf. 5.2.1	`clarin.eu:cr1:c_1359626292113 #ResourceTitle`
set	denotion of the containing termset	`cmd`
type	one of `full-path` or `min-path`	`full-path`
schema	*profileID*	`clarin.eu:cr1:p_1357720977520`
node-value	*dotPath*	`SpeechCorpus.Access.Contact.Url`

Listing 5.1: sample `Term` element encoding an ISOcat data category

```
<Term concept-id="http://www.isocat.org/datcat/DC-2544" set="isocat"
         type="label" xml:lang="fr">nom de ressource</Term>
```

Listing 5.2: sample `Term` element encoding a CMD element

```
<Term type="CMD_Element" name="Url" datcat="http://www.isocat.org/datcat/DC-2546"
         id="clarin.eu:cr1:c_1290431694487#Url" parent="Contact"
         path="SpeechCorpus.Access.Contact.Url"/>
```

Listing 5.3: sample `Term` element encoding a term in the inverted index

```
<Term set="cmd" type="full-path" schema="clarin.eu:cr1:p_1357720977520"
         id="clarin.eu:cr1:c_1359626292113#ResourceTitle"
         concept-id="http://www.isocat.org/datcat/DC-2545" >
     AnnotatedCorpusProfile.GeneralInfo.ResourceTitle
</Term>
```

Type Concept

`Concept` represents a data category. Identifier is the PID issued by the DCR encoded in the *id* attribute. It groups all terms belonging to given data category. The content model is a sequence of `Terms` followed by a sequence of `info` elements. Initially, after loading from DCR, a `Concept` contains only `Terms` of type: `id, mnemonic, label` (in multiple languages) encoding the corresponding attributes of the data category, followed by `info` elements holding the definition (also potentially in different languages). In the inverted index, the `Concept` is enriched with the `Terms` representing corresponding CMD entities (cf. Listing 5.6).

Listing 5.4: sample `Concept` element representing the data category resourceTitle

```
<Concept id="http://www.isocat.org/datcat/DC-2545" type="datcat">
  <Term set="isocat" type="mnemonic">resourceTitle</Term>
  <Term set="isocat" type="id">DC-2545</Term>
  <Term set="isocat" type="label" xml:lang="en">resource title</Term>
```

```
    <Term set="isocat" type="label" xml:lang="fi">resurssin otsikko</Term>
    ...
    <info xml:lang="en">The title is the complete title
               of the resource without any abbreviations.</info>
     ...
  </Concept>
```

Listing 5.5: Sample of the inverted index `Concept` ↦ `Term`

```
<Concept id="http://www.isocat.org/datcat/DC-2545" type="datcat">
   <Term set="isocat" type="mnemonic">resourceTitle</Term>
   <Term set="isocat" type="id">DC-2545</Term>
   <Term set="isocat" type="label" xml:lang="en">resource title</Term>
   <Term set="isocat" type="label" xml:lang="hr">naslov resursa</Term>
   <Term set="isocat" type="label" xml:lang="lv">resursa nosaukums</Term>
   ...
   <Term set="cmd" type="full-path" schema="clarin.eu:cr1:p_1357720977520"
          id="clarin.eu:cr1:c_1359626292113#ResourceTitle">
                 AnnotatedCorpusProfile.GeneralInfo.ResourceTitle</Term>
   <Term set="cmd" type="full-path" schema="clarin.eu:cr1:p_1297242111880"
          id="clarin.eu:cr1:c_1271859438123#Title">
                 AnnotationTool.GeneralInfo.Title</Term>
   <Term set="cmd" type="full-path" schema="clarin.eu:cr1:p_1274880881885"
          id="clarin.eu:cr1:c_1274880881884#Title">
                 imdi-corpus.Corpus.Title</Term>
   <Term set="cmd" type="full-path" schema="clarin.eu:cr1:p_1271859438204"
          id="clarin.eu:cr1:c_1271859438201#Title">
                 Session.Title</Term>
   ...
</Concept>
```

Type `Relation`

As explained in 4.2.1, the framework allows to express relations between concepts or data categories. These are maintained in the Relation Registry and fetched from there by SMC upon initialization. Type `Relation` is the internal representation of this information. It has attribute *type* indicating the type of the relation as delivered by RR (currently only `sameAs`). The relations of one relation set are enclosed in `Termset` element carrying the identifier of the relation set. The content of `Relation` is a sequence of at least two `Concepts`. Currently, it is always exactly two `Concepts` corresponding to the pairs delivered from RR, but by traversing the equivalence relation concept clusters (or "cliques") could be generated, that contain more than two equivalent concepts.

Listing 5.6: Internal representation of the relation between concepts

```
<Relation type="sameAs">
  <Concept type="datcat" id="http://www.isocat.org/datcat/DC-2484" role="about"/>
  <Concept type="datcat" id="http://purl.org/dc/elements/1.1/language"/>
</Relation>
```

Type `Termsets/Termset`

`Termset` groups a set of terms as outlined in 5.4. It is identified by the `@set` attribute. For example all french labels of isocat data categories under the identifier `isocat-fr` build a termset, as well as all the full-paths of one profile. The content of the `Termset` can optionally begin with an `info` element (conveying information as provided by the source registry, like definition, creation date or author) followed by a flat or nested list of `Term` elements.

Finally, `Termsets` is a root element grouping `Termset` elements.

Listing 5.7: `Termset` element representing a CMD profile

```
<Termset name="AnnotatedCorpusProfile" id="clarin.eu:cr1:p_1357720977520"
         type="CMD_Profile">
  <info>
    <id>clarin.eu:cr1:p_1357720977520</id>
    <description>A CMDI profile for annotated text corpus resources.</description>
    <name>AnnotatedCorpusProfile</name>
    <registrationDate>2013-01-31T11:57:12+00:00</registrationDate>
    <creatorName>nalida</creatorName>
    ...
 </info>
 <Term type="CMD_Component" name="GeneralInfo" datcat=""
         id="clarin.eu:cr1:c_1359626292113"
         parent="AnnotatedCorpusProfile"
         path="AnnotatedCorpusProfile.GeneralInfo">
     <Term ...
   </Term>
   ...
</Termset>
```

5.3 cx – crosswalk service

The crosswalk service offers the functionality, that was understood under the term *Semantic Mapping* as conceived in the original plans of the Component Metadata Infrastructure. Semantic interoperability has been one of the main concerns addressed by the CMDI and appropriate provisions were weaved into the underlying meta-model as well as all the modules of the infrastructure. Consequently, the infrastructure has also foreseen this dedicated module, *Semantic Mapping*, that exploits this mechanism to find **corresponding fields in different metadata schemas**.

The task of the crosswalk service is to collect the relevant information maintained in the registries of the infrastructure and process it to generate the mappings, or **crosswalks** between fields in heterogeneous metadata schemas. These crosswalks can be used by other applications representing the base for concept-based search in the heterogeneous data collection of the joint CLARIN metadata domain (cf. 5.4).

The core means for semantic interoperability in CMDI are the *data categories* (cf. 4.2.1), well-defined atomic concepts, that are supposed to be referenced in schemas by annotating fields to unambiguously indicate their intended semantics. Drawing upon this system, the crosswalks are not generated directly between the fields of individual schemas by some kind of matching algorithm (cf. 2.3), but rather the data categories are used as reliable bridges for translation. This results in clusters of semantically equivalent metadata fields (with data categories serving as pivotal points) instead of a collection of pair-wise links between fields.

5.3.1 Interface Specification

In this section, we define the abstract interface of the proposed service, in terms of the input parameters and output data format.

Method *list*

Method *list* lists available items for given context or type. This allows the client applications to configure the query input and provide autocompletion functionality. Table 5.4 lists the accepted values for the *$context* parameter and the corresponding types of returned data.

Definition 5.2: URI-pattern of the *list* method

$$/smc/cx/list/\$context$$

Table 5.4: Allowed values for parameters of the `list`-method and corresponding return values

$context	returns a list of
`*,top`	available termsets
{*termset*}	terms (CMD components and elements) of given termset
`dcr`	available data category registries (isocat, dublincore)
`isocat`	ISOcat data categories referenced in CMD data
`languages`	available languages (only for isocat data categories)
`cmd-profiles`	all available CMD profiles
`cmd-full-paths`	all complete (starting from Profile) *dotPaths* to CMD components and elements
`cmd-minimal-paths`	reduced but still unique paths to CMD components and elements
`relsets`	available relation sets (defined in the Relation Registry)

Method *explain*

The service also has to deliver additional information about the indexes like description and a link to the definition of the entity in the source registry.

Definition 5.3: URI-pattern of the `explain` method

$$/smc/cx/explain/\{\$context\}\ [\ /\{\$term\}\ ]\ [\ ?format = \$format\ ]\ [\ ?lang = \$lang\ ]$$

```
/smc/cx/explain/cmd/clarin.eu:cr1:p_1357720977520
/smc/cx/explain/isocat/DC-2506?lang=sv,et
```

Listing 5.8: Sample output of the *explain* function for a data category

```
<Concept type="datcat" id="http://www.isocat.org/datcat/DC-2506">
  <Term set="isocat" type="mnemonic">annotationMode</Term>
  <Term set="isocat" type="id">DC-2506</Term>
  <Term set="isocat" type="label" xml:lang="et">m\"{a}rgendusviis</Term>
  <Term set="isocat" type="label" xml:lang="sv">annoteringsl\"{a}ge</Term>
  <info xml:lang="et">N\"{a}itab, kas ressurss m\"{a}rgendati k\"{a}sitsi v\~{o}i automaatselt.</info>
  <info xml:lang="sv">Flagga som indikerar om resursen skapades manuellt eller automatiskt.</info>
</Concept>
```

Method *map*

Method *map* performs the actual translations: it accepts any index (adhering to the *smcIndex* datatype, cf. 5.2.1) and returns a list of corresponding indexes.
Parameter definition:

Definition 5.4: General function definition

$$smcIndex \mapsto smcIndex*$$

Definition 5.5: URI-pattern of the *map* method

$$/smc/cx/map/\{\$context\}/\{\$term\} \ [\ ?format = \{\$format\}\ ]\ [\ \&relset = \{\$relset\}\ ]$$

$context identifies the context to search in for the *$term*, primarily this is one of `[*, isocat, dc, cmd]`, in extended mode any of terms listed in table 5.4 is accepted

$term *smcIndex* term (without the context prefix); the term is used to lookup a concept, to deliver the list of equivalent indexes; case-insensitive

$format the desired result format can be indicated explicitely, alternatively to default content negotiation; one of `[json, rdf, xml]`; `xml` is default

$relset optional; reference to a relation set to be combined with the identified concept to expand the cluster of matching concepts; allows multiple values from `list/relsets`; if multiple sets are listed they are all applied in the expansion

Possible return formats:

default internal XML format with all attributes (Terms.xsd, cf. listing 5.9)

schema distinct schemas (`Termset`) referencing given data category or string

```
<Termset schema="clarin.eu:cr1:p_1295178776924" name="serviceDescription"/>
```

datcat distinct data categories, by grouping the `Term@datcat` attribute of the matching terms

```
<Term concept-id="http://www.isocat.org/datcat/DC-2512"
         set="isocat" type="datcat">creatorFullName</Term>
```

cmdid, id distinct cmd entities grouped by `@id`

```
<Term type="CMD_Element" name="Name" elem="Name" parent="Session"
      datcat="http://www.isocat.org/datcat/DC-2544"
      id="clarin.eu:cr1:c_1349361150645#Name" path="DBD.Session.Name"/>
```

Sample request

```
/smc/cx/map/isocat/resourceTitle
```

Listing 5.9: Corresponding sample output

```
<Terms >
   <Term set="cmd" type="full-path" schema="clarin.eu:cr1:p_1297242111880"
      id="clarin.eu:cr1:c_1271859438123#Title">
            AnnotationTool.GeneralInfo.Title</Term>
   <Term set="cmd" type="full-path" schema="clarin.eu:cr1:p_1288172614014"
      id="clarin.eu:cr1:c_1288172614011#resourceTitle">
            BamdesLexicalResource.BamdesCommonFields.resourceTitle
```

```
    </Term>
  <Term set="cmd" type="full-path" schema="clarin.eu:cr1:p_1274880881885"
      id="clarin.eu:cr1:c_1274880881884#Title">
             imdi-corpus.Corpus.Title</Term>
  <Term set="cmd" type="full-path" schema="clarin.eu:cr1:p_1271859438204"
      id="clarin.eu:cr1:c_1271859438201#Title">
             Session.Title</Term>
  <Term set="cmd" type="full-path" schema="clarin.eu:cr1:p_1272022528363"
      id="clarin.eu:cr1:c_1271859438123#Title">
             LexicalResourceProfile.LexicalResource.GeneralInfo.Title</Term>
   <Term set="cmd" type="full-path" schema="clarin.eu:cr1:p_1284723009187"
      id="clarin.eu:cr1:c_1271859438123#Title">collection.GeneralInfo.Title</Term>
```

We can distinguish following levels for the mapping function:
(1) *data category identity* – for the resolution only the basic data category map derived from Component Registry is employed. Accordingly, only indexes denoting CMD elements (*cmdIndex)* bound to a given data category are returned:

```
isocat.size ↦                     [teiHeader.extent, TextCorpusProfile.Number]
```

cmdIndex as input is also possible. It is translated to a corresponding data category, proceeding as above:

```
imdi-corpus.Name ↦
(isocat.resourceName) ↦           TextCorpusProfile.GeneralInfo.Name
```

(2) *relations between data categories* – employing also information from the Relation Registry, related (equivalent) data categories are retrieved and subsequently both the input and the related data categories resolved to a list of *cmdIndexes*:

```
isocat.resourceTitle ↦
(+ dc.title) ↦                    [GeneralInfo.Title, Text.TextTitle,
                                  collection.CollectionInfo.Title, resourceInfo.
                                  identificationInfo.  resourceName,
                                  teiHeader.titleStmt.title, teiHeader.monogr.title]
```

(3) *container data categories* – further expansions will be possible once the *container data categories* [74] will be used.[3] The idea is to set a concept link also for the components, meaning that besides the “atomic” data category for actorName, there would be also a data category for the complex concept Actor. Having concept links also on components will require a compositional approach for the mapping function, resulting in:

```
Actor.Name ↦                      [Actor.Name, Actor.FullName,
                                  Person.Name, Person.FullName]
```

5.3.2 Implementation

The core functionality of the SMC is implemented as a set of XSL-stylesheets

At the core of the described module is a set of XSL-stylesheets, governed by an ant-build file and a configuration file holding the information about individual source registries. The documentation of XSLT stylesheets is found in appendix D.1.

[3]Although metadata modellers are encouraged to indicate data categories for both components and elements, this is taking up only slowly and currently only around 14 per cent of the components have a data category specified.

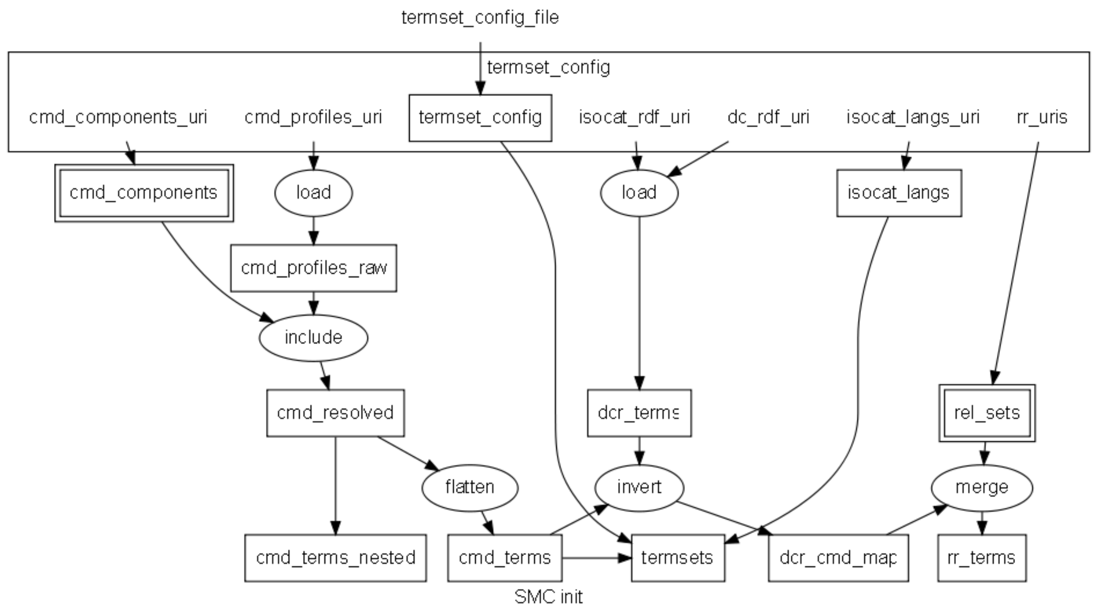


Figure 5.2: The various stages of the data flow during the initialization

The service is implemented as a RESTful service, however only supporting the GET operation, as it operates on a data set, that the users cannot change directly. (The changes have to be performed in the upstream registries.)

Initialization

During initialization the application fetches the information from the source modules (cf. 4.2) and transforms it into the internal Terms format (cf. 5.2.2). All profiles and components from the Component Registry are read and all the URIs to data categories are extracted to construct an inverted map of data categories.5.1

Definition 5.6: Principal structure of the inverted index

$$datcatPID \mapsto profile.component.element*$$

The collected data categories are enriched with information from corresponding registries (DCRs), adding the label, the description and available translations into other working languages. Finally, relation sets defined in the Relation Registry are fetched and matched with the data categories in the map to create sets of semantically equivalent (or otherwise related) data categories.

Following datasets are available, after the initialization sequence has finished (cf. figure 5.2):

termets a list of all available Termsets compiled from the CMD profiles, and available DCRs; for ISOcat a termset is generated for every available language

cmd-terms a flat list of `Term` elements representing all components and elements in all known profiles; grouped in `Termset` elements representing the profiles

cmd-terms-nested as above, however the `Term` elements are nested reflecting the component structure in the profile

dcr-terms a list of `Concept` elements representing the data categories with nested `Term` elements encoding its properties (`id, label`

dcr-cmd-map the main inverted index – a list of concepts as in dcr-terms, but with additional `Term` elements included in the `Concept` elements representing the CMD components or elements corresponding to given data category (cf. listing 5.6)

rr-terms Additional index generated based on the relations between data categories as defined in the Relation Registry; the `Concept` elements representing the pair of related data categories are wrapped with a `Relation` element (with a `@type` attribute

Operation

For the actual service operation a minimal application has been implemented, that accesses the cached internal datasets and optionally applies XSL stylesheets for post-processing depending on requested format. The application implements the interface as defined in 5.3.1 as a XQuery module based on the restxq library within an eXist XML database.

5.3.3 Extensions

Once there will be overlapping[4] user-defined relation sets in the Relation Registry an additional input parameter will be required to *explicitly restrict the selection of relation sets* to apply in the mapping function.

Also, use of *other than equivalence* relations will necessitate more complex logic in the query expansion and accordingly also more complex response of the crosswalk service, either returning the relation types themselves as well or equip the list of indexes with some kind of similarity ratio.

5.4 qx – concept-based search

To recall, the main goal of this work is to enhance the search capabilities of the search engines serving the metadata. In this section we want to explore how this shall be accomplished, i.e. how to bring the enhanced capabilities to the user.

The emphasis lies on the query language and the corresponding query input interface. Crucial aspect is the question how to integrate the additional processing, i.e. how to deal with the even greater amount of information in a user-friendly way without overwhelming the user, while still being verbose about the applied processing on demand for the user to understand how the result came about and even more important, to allow the user to manipulate the processing easily.

Note, that this chapter deals only with the schema level, i.e. the expansion here pertains only to the indexes to be searched in, not to the search terms. The instance level is dealt with in **??**.

Note, also that *query expansion* yet needs to be distinguished from *query translation*, a task to express input query in another query language (e.g. CQL query expressed as XPath).

[4]i.e. different relations may be defined for one data category in different relation sets

5.4.1 Query language

As base query language to build upon the *Context Query Language* (CQL) is used, a well-established standard, designed with extensibility in mind. CQL is the query language defined as part of SRU/CQL – the communication protocol introduced by the Library of Congress. SRU is a simplified, XML- and HTTP-based successor to Z39.50[85], which is very widely spread in the library networks. It was introduced 2002 [86]. In recent years OASIS took over the management of the standard and a first version of the refurbished specification under OASIS rule was published 2012 [84]. The maintenance of SRU/CQL has been transfered from LoC to OASIS in 2012, and OASIS released a first version of the protocol as Committee Specification in April 2012[84].)

Coming from the libraries world, the protocol has a certain bias in favor of bibliographic metadata. However, the protocol is defined in a very generic way, with a strong focus on extensibility. It is equally suitable for content search.

The protocol part (SRU) defines three major operations: 1) *explain*: in which the target repository announces its particular configuration (e.g. available indices), 2) *scan*: informing about terms available in/for given index, and 3) *searchRetrieve*: returning a search result based on a CQL query.

The query language part (CQL - Context Query Language) defines a relatively complex and complete query language. The decisive feature of the query language is its inherent extensibility allowing to define own indexes and operators. In particular, CQL introduces so-called *context sets* – a kind of application profiles that allow to define new indexes or even relation operators in own namespaces. This feature can be employed to integrate the dynamic indexes adhering to the *smcIndex* syntax as defined in 5.2.1.

The SRU/CQL protocol has been adopted by the CLARIN community as base for a protocol for federated content search[5] (FCS) [83].

5.4.2 Query Expansion

As long as the indexes to expand with are equivalent the query expansion is simply disjunction, returning a union of matching records. Thus `isocat.resourceTitle any "elephant"` would translate into

```
GeneralInfo.Title any "elephant"
OR resourceInfo.resourceName any "elephant"
OR CollectionInfo.Title any "elephant"
OR teiHeader.titleStmt.title any "elephant"
```

Alternatively to the – potentially costly – on the fly expansion, the concept-based equivalence clusters could be applied already during the indexing of the data. That means that "virtual" search indexes are defined for individual data categories in which values from all metadata fields annotated with given data category are indexed. Indeed, this approach is already being applied in the search applications VLO and Meertens Institute Search Engine (cf. 4.2.3).

5.4.3 SMC as module for Metadata Repository

As a concrete proof of concept the functionality of SMC has been integrated into the Metadata Repository, another module of the CMDI providing all the metadata records harvested within the CLARIN joint metadata domain (cf. 4.2.3).

[5] http://clarin.eu/fcs

Metadata repository itself is implemented as custom project within cr-xq, a generic web application developed in XQuery running within the eXist XML-database. cr-xq is developed by the author as part of a larger publication framework corpus_shell. As can be seen in figure 5.3 within cr-xq the crosswalk service – implemented as the smc-xq module – is used by the search module fcs, which is in turn used by the query_input module, that provides a user interface widget for formulating the query.

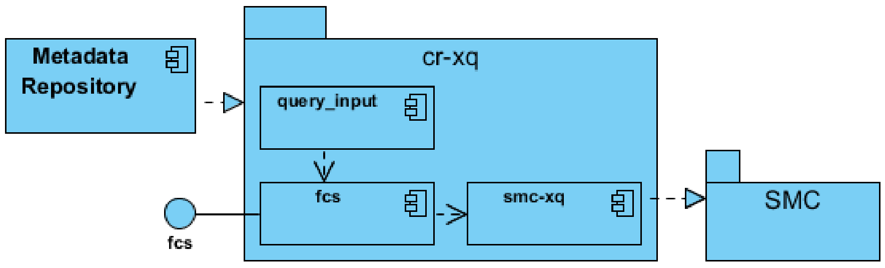


Figure 5.3: The component view on the SMC - modules and their inter-dependencies

5.4.4 User Interface

A starting point for our considerations is the traditional structure found in many ("advanced") search interfaces, which is basically a an array of tuples of index, comparison operator, terms combined by a boolean operator. This is reflected in the CQL syntax with the basic *searchClause* and the boolean operators to formulate more complex queries.

Definition 5.7: Generic data format for structured queries

$$< index, operation, term, boolean > +$$

Definition 5.8: The basic `searchClause` of the CQL syntax

$$searchClause \ ::= \ index\ relation\ searchTerm$$

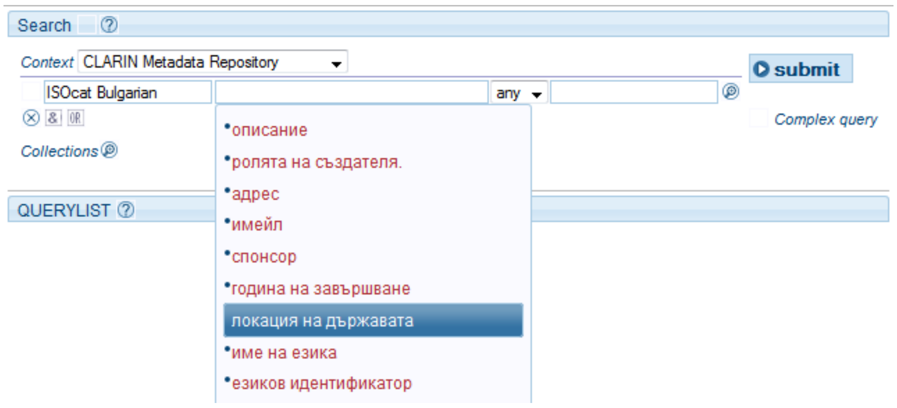


Figure 5.4: A proposed query input interface offering concepts as search indexes

Using data categories from ISOcat as search indexes brings about – next to solid semantic grounding – the advantage of multilingual labels and descriptions/definitions. Although we concentrate on query input, the use of indexes has to be consistent across the user interface, be it in labeling the fields of the results, or when providing facets to drill down the search.

A fundamentally different approach is the "content first" paradigm, that, similiar to the notorious simple search fields found in general search engines, provides suggestions via autocompletion on the fly, when the user starts typing any string. The difference is, that the suggestions are typed, so that the user is informed from which index given term comes (person, place, etc.)

Combining the two approaches, we could arrive at a “smart” widget a input field with on the fly query parsing and contextual autocomplete. Though even such a widget would still share the underlying data model of CQL + smcIndexes.

5.5 SMC Browser

As the CMD dataset keeps growing both in numbers and in complexity, the call from the community to provide enhanced ways for its exploration gets stronger. In the following, some design considerations for an application to answer this need are proposed.

While the Component Registry (cf. 4.2.1) allows to browse, search and view existing profiles and components, it is not possible to easily find out, which components are reused in which profiles and also which data categories are referenced by which elements. However this kind of information is crucial during profile creation as well as for curation of the existing profiles, as it enables the data modeller to recognize a) which components and data categories are those most often used, indicating their adoption and popularity within the community and b) the thematic contexts in which individual components are used, providing a hint about their appropriateness for given research data.

5.5.1 Design

In the following, we elaborate on the basic idea of the proposed application, the source data, requirements and proposed application UI-layout.

Basic concept

If we consider the CMD data model (cf. 3.1) we recognize that every profile can be expressed as a tree with the profile component as the root node, the used components as intermediate nodes and elements or data categories as leaf nodes, parent-child relationship being defined by *inclusion* and *reference*.

Definition 5.9: *inclusion* and *reference* relationship

$$\begin{aligned} cmds : Component &\xrightarrow{includes} \quad cmds : Component \\ cmds : Component &\xrightarrow{includes} \quad cmds : Element \\ cmds : Element &\xrightarrow{refersTo} \quad DatCat \end{aligned}$$

The reuse of components in multiple profiles and especially also the referencing of the same data categories in multiple CMD elements leads to a blending of the individual profile trees into a graph (acyclic directed, but not necessarily connected). The main idea for the SMC Browser is to **visualize this graph inherent in the CMD data**.

Requirements

Given the size of the data set (currently more than 4.000 nodes and growing) it is obvious, that it is not possible to overview the whole of the graph in one view. Thus, a general essential requirement is to be able to select and view subgraphs by various means.

In a basic scenario, user looks for possibly reusable profiles or components, based on some common terms associated with the type of data to be described (e.g. `"corpus"`). If the search yields matching profiles or components, the user should be able to view the whole structure of the profiles, explore the definitions for individual components and see which data categories are being referenced for semantic grounding. Furthermore, it has to be possible to view multiple profiles concurrently, in particular to be able to see the components or data categories they share and, vice versa, in which profiles a given data category is referenced.

This scenario implies a few requirements on the user interface:

- select nodes from a list of all available nodes (ideally grouped by type)
- filter the node list
- select an arbitrary number of nodes of any type (be it profiles, components, elements, data categories)
- traverse the graph starting from selected nodes into arbitrary depth
- traverse the graph backwards (meaning against the direction of the edges, i.e. e.g. from data categories towards the profiles)
- maintain the identity of the nodes, meaning one component or one data category used in two profiles has to be represented by one node (for displaying the reuse)
- show auxiliary information about the nodes on demand

Application layout

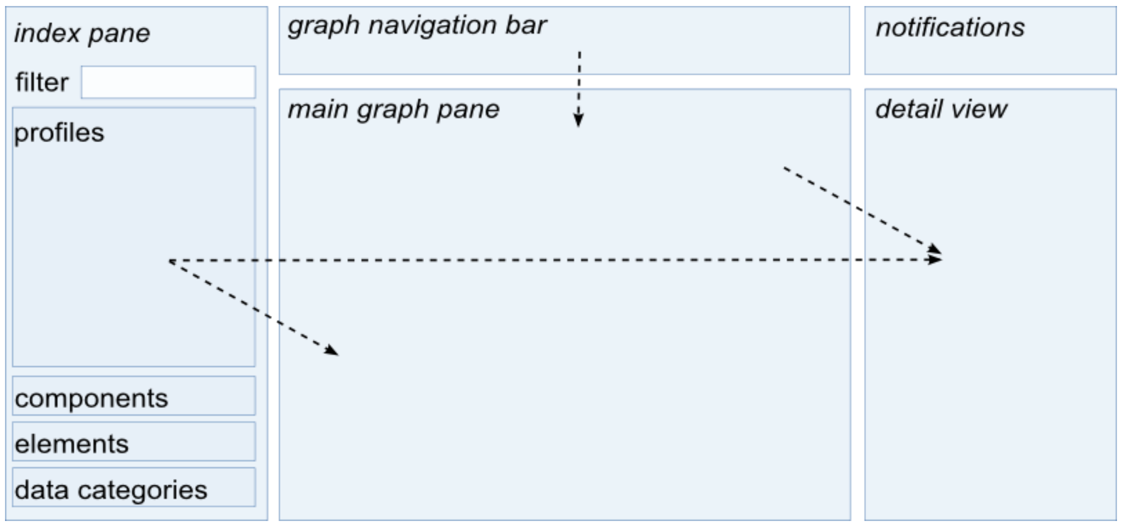


Figure 5.5: A sketch of a possible layout for the SMC Browser – individual parts of the user interface and the update dependencies

Prospective parts of the application layout (cf. figure 5.5):

index pane list of all available nodes (profiles, components, elements, data categories); allows to select nodes to be displayed in the graph pane

main graph pane displays the selected subgraph, needs as much space as possible

graph navigation bar for manipulation of the displayed graph by various means

detail view displaying definition and statistical information for selected nodes

statistics a separate view on the data listing the statistical information for whole dataset in tables

notifications a widget to provide feedback about the system status to the user

5.5.2 Implementation

The application is implemented in javascript based on a generic visualization js-library d3[6]. The library allows for data-driven visualization (hence the name d3 = data-driven documents), attributes of data items being dynamically bound to attributes of the SVG objects representing them. This caters for high flexibility, fast development and consistent data views. The library also delivers the base graph layout algorithm: *force-directed graph layout*[7]:

> A flexible force-directed graph layout implementation using position Verlet integration to allow simple constraints. [. . .] In addition to the repulsive charge force, a pseudo-gravity force keeps nodes centered in the visible area and avoids expulsion of disconnected subgraphs, while links are fixed-distance geometric constraints. Additional custom forces and constraints may be applied on the "tick" event, simply by updating the x and y attributes of nodes.

Especially remarkable feature is the possibility to add custom constraints, that are accomodated with the constraints imposed by the base algorithm. This enables flexible customization of the layout, still harnessing the power of the underlying layout algorithm. At the same time this is a quite challenging feature to master, as with different constraint affecting the layout algorithm, it is at times difficult to understand the impact of a specific constraint on the layout.

Data preprocessing

The application operates on a set of static XHTML and JSON data files, that are created in a preprocessing step and deployed with the application. The preprocessing consists of a series of XSLT transformations (cf. figure 5.6), starting from the internal datasets generated during the initialization (cf. 5.3.2). The HTML output for smc-stats is generated in two steps (*track S*) via an intermediate internal generic XML format for representing tabular data. The JSON data for the smc-graph as expected by the d3 library is also generated in two steps (*track G*). First, a XML representation of the graph is generated from the data (terms2graph.xsl), on which a generic XSLT-transformation is applied (graph_json.xsl) transforming the XML graph into required JSON format. In fact, this track is run multiple times generating different variants of the graph, featuring different aspects of the dataset:

[6] https://github.com/mbostock/d3/
[7] https://github.com/mbostock/d3/wiki/Force-Layout#wiki-force

SMC graph basic the basic graph contains *profiles* $\mapsto$ *components* $\mapsto$ *elements* $\mapsto$ *datcats*; processing 155 profiles yields a graph with over 4.500 nodes and over 7.500 edges

SMC graph all additionally rendering the new profile-groups and relations between data categories (from Relation Registry)

only profiles + datcats just profiles and data categories are rendered (with direct links between those, skipping all components and elements)

profiles + datcats + datcats + groups + rr as above but again with profile-groups and relations

profiles similarity just profiles with links between them representing the degree of similarity based on the reuse of components and data categories

Additionally, a detour pass (*track D*) is executed, in which the graph is also transformed into the DOT format and run through the Graphviz dot tool to get a SVG representation of the graph. In an early stage of development, this was actually the only processing path. However soon it became obvious, that the graph is getting to huge to be displayed in its entirety. Figure D.1 displays an old version of such a dot generated graph visualization. Currently, the `dot` output is only used as input for the final graph data, providing initialization coordinates for the nodes in the `dot`-layout.

To The graph is constructed from all profiles defined in the Component Registry and related datasets. To resolve (multilingual) name and description of data categories referenced in the CMD elements definitions of referenced data categories from DublinCore and ISOcat are fetched.

User interface

As proposed in the design section, the starting point when using the SMC browser is the node list on the left, listing all nodes grouped by type (profiles, components, elements, data categories) and sorted alphabetically. This list can be filtered by a simple substring search which is important, as already now there are more than 4.000 nodes in the graph. Individual nodes are selected and deselected by a simple click. All selected nodes are displayed in the main graph pane represented by a circle with a label. The representation is styled by type. Based on the settings in the navigation bar (cf. figure 5.7), next to the selected nodes also related nodes are displayed. The `depth-before` and `depth-after` options govern how many levels in each direction are traversed and displayed starting from the set of selected nodes. Option `layout` allows to select from one of available layouts – next to the basic `force` layout there are also directed layouts, that are often better suited for displaying the directed graph. Other options influence the layouting algorithm (`link-distance`, `charge`, `friction`) and the visual representation of the nodes and edges (`node-size, labels, curve`).

One special option is `graph`, that allows to switch between different graphs as listed in 5.5.2.

There is user documentation deployed with the application and featured in the appendix D.2, where all aspects of interaction with the application (D.2.3) and the options in the navigation bar (D.2.4) are described.

5.5.3 Extensions

Next to the basic setup described above, there is a number of possible additional features, that could enhance the functionality and usefulness of the discussed tool.

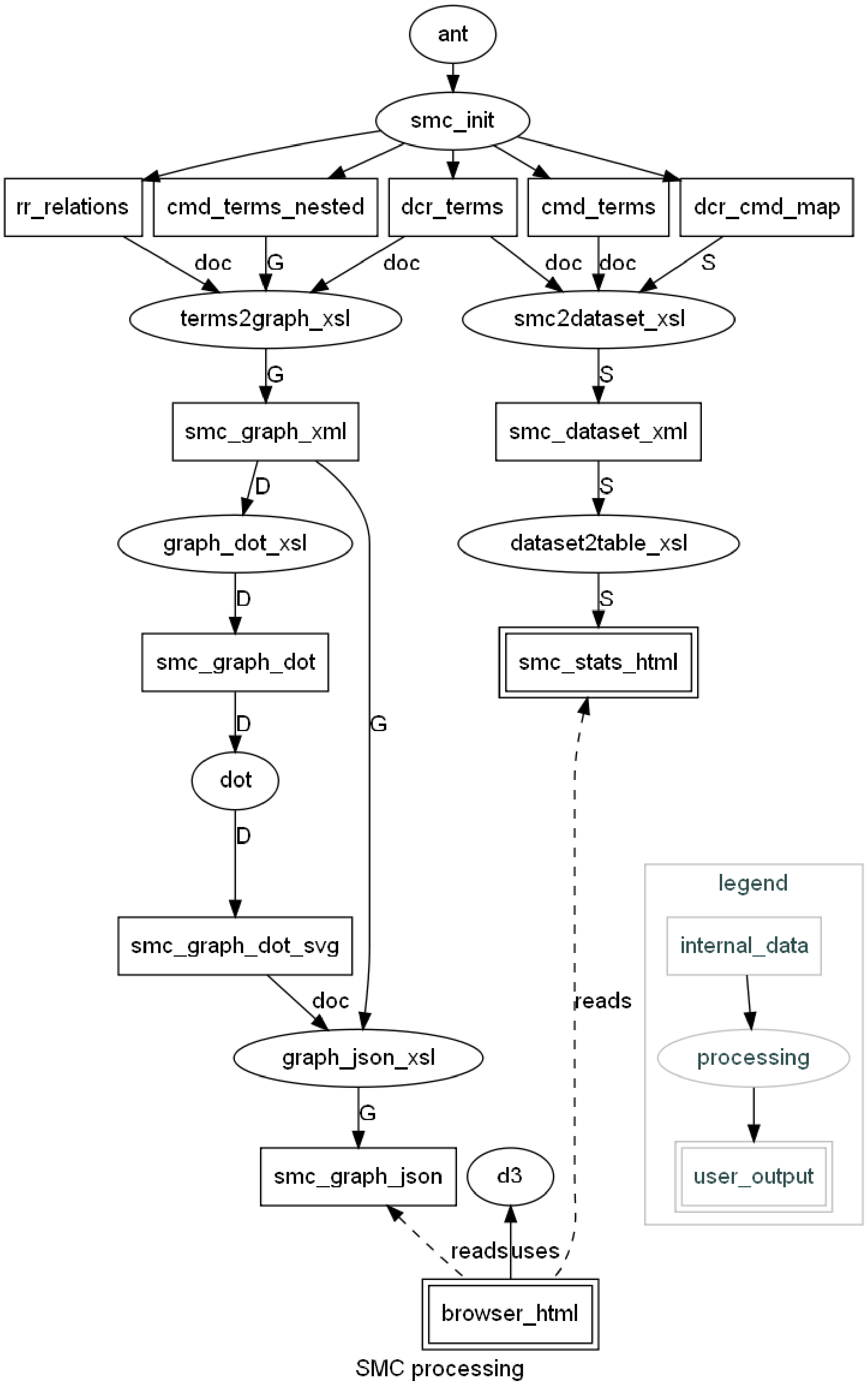


Figure 5.6: The data flow in process of precomputing data for the SMC browser

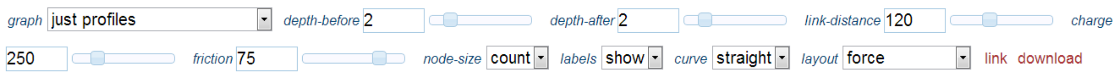

graph just profiles depth-before 2 depth-after 2 link-distance 120 charge 250 friction 75 node-size count labels show curve straight layout force link download

Figure 5.7: Navigation bar of the SMC Browser with a number of options to manipulate the visible graph

Graph operations – differential views

An important feature would be to be able to apply set operations on selected (sub)graphs, especially *intersection* and *difference*. This would enable the user to easily extract components (nodes) that are shared (or not shared) among given schemas (subgraphs).

Generalization

There is a high potential to broaden the scope of application for the discussed tool, provided some generalizations are taken into account. Equipped with a more flexible or modular matching algorithm (additionally to the initially foreseen identity match), the tool could visualize matches between any given schemas, not only CMD-based ones.

Also, the input format being a graph, with appropriate preprocessing the tool could visualize any structural information, that is suited to be expressed as graph, like cooccurrence analysis, dependency networks, RDF data in general etc.

Viewer for external data

The above feature would be even more useful if the application would be enabled to ingest and process external data. The data can be passed either via upload or via a parameter with a URL of the data. This is especially attractive also to providers of other data and applications, who could provide a simple link in their user interface (with the data-parameter appropriately set), that would allow to visualize their data in the SMC browser.

One prominent visualization application offering this feature is the geobrowser e4D[8] (currently GeoTemCo[9], developed in the context of the europeana connect initiative), accepting data in KML format.

Integrate with instance data

The usefulness and information gain of the application could be greatly increased by integrating the instance data. I.e. generate and display a variant of the graph which contains only profiles for which there is actually instance data present in the CLARIN joint metadata domain. Obviously, in such a visualization the size of data could be incorporated, in the most simple case number of records being mapped on the radius of the nodes, but there are a number of other metrics that could be applied in the visualizations.

Also such a visualization could feature direct search links from individual nodes into the dataset, i.e. from a profile node a link could lead into a search interface listing metadata records of given profile.

[8]http://www.informatik.uni-leipzig.de:8080/e4D/
[9]https://github.com/stjaenicke/GeoTemCo

5.6 Application of *schema matching* techniques in SMC

Even though the described module is about "semantic mapping", until now we did not directly make use of the traditional ontology/schema mapping/alignment methods and tools as summarized in 2.3. This is due to the fact that in this work we can harness the mechanisms of the semantic interoperability layer built into the core of the CMD Infrastructure, which integrates the task of identifying semantic correspondences directly into the process of schema creation, to a high degree obsoleting the need for a posteriori complex schema matching/mapping techniques. Or put in terms of the schema matching methodology, the system relies on explicitly set concept equivalences as base for mapping between schema entities. By referencing a data category in a CMD element, the modeller binds this element to a concept, making two elements linked to the same data category trivially equivalent.

However this is only holds for schemas already created within the CMD framework (and even for these only to a certain degree, as will be explained later). Given the growing universe of definitions (data categories and components) in the CMD framework the metadata modeller could very well profit from applying schema mapping techniques as pre-processing step in the task of integrating existing external schemas into the infrastructure. (User involvement is identified by [10] as one of promising future challenges to ontology matching.) Already now, we witness a growing proliferation of components in the Component Registry and of data categories in the Data Category Registry.

Let us restate the problem of integrating existing external schemas as an application of *schema matching* method: The data modeller starts off with existing schema S_x. The system accomodates a set of schemas[10] $S_{1..n}$. It is very improbable, that there is a $S_y \in S_{1..n}$ that fully matches S_x. Given the heterogeneity of the schemas present in the field of research, full alignments are not achievable at all. However thanks to the compositional nature of the CMD data model, data modeller can reuse just parts of any of the schemas – the components c. Thus the task is to find for every entity $e_x \in S_x$ the set of semantically equivalent candidate components $\{c_y\}$, which corresponds to the definitions of mapping function for single entities as defined in [17]. Given, that the modeller does not have to reuse the components as they are, but can use existing components as base to create his own, she is helped even with candidates that are not equivalent, thus we can further relax the task and allow even candidates that are just similar to a certain degree, that can be operationalized as threshold t on the output of the *similarity* function Being only a pre-processing step meant to provide suggestions to the human modeller implies higher importance to recall than to precision.

Another requirement is that the matching entities should be maximal regarding the compositional tree:

Definition 5.10

$$\begin{aligned}
&map(e_{x1}) \rightarrow c_{y1}\\
&map(e_{x2}) \rightarrow c_{y2}\\
&candidates := \{(e_{xa}, c_{ya}) : c_{ya} \notin descendants(c_{yb})\}
\end{aligned}$$

Next to the usual features and measures that can be applied like label equality or string-similarity and structural equality, the mapping function could be enriched with

[10]We talk of schema even though the creation (and also remodelling) takes place in the component registry by creating CMD profiles and components, because every profile has an unambiguous expression in XML Schema.

extensional features based on the concept clusters as delivered by the crosswalk service 5.3. It would be also worthwhile to test, in how far the *smcIndex* paths as defined in 5.2.1 could be used as feature (compute the longest matching subpath).

Although we examplified on the case of integration of an external schema, the described approach could be applied also to the schemas already integrated in the system. Although there is already a high baseline given thanks to the mechanisms of reuse of components and data categories, there certainly still exist semantic proximities that are not explicitly expressed by these mechanisms. This deficiency is rooted in the collaborative creation of the CMD components and profiles, where individual modellers overlooked, deliberately ignored or only partially reused existing components or profiles. This can be seen on the case of multiple teiHeader profiles, that though they are modelling the same existing metadata format, are completely disconnected in terms of components and data category reuse (cf. 7.3.2).

Note, that in the case of reuse of components, in the normal scenario, the semantic equivalence is ensured even though the new component (and all its subcomponents) is a copy of the old one with new identity, because the references to data categories are copied as well. Thus, by default, the new component shares all data categories with the original one and the modeller has to deliberately change them if required. But even with reuse of components scenarios are thinkable, in which the semantic linking gets broken, or is not established, even though semantic equivalency pervails.

The question is, what to do with the new correspondences that would possibly be determined, when, as proposed, we would apply the schema matching on the integrated schemas. One possibility is to add a data category, if one of the pair is still one missing. However if both already are linked to a data category, the data category pair could be added to the relation set in Relation Registry (cf. 4.2.1).

Once all the equivalences (and other relations) between the profiles/schemas were found, simliarity ratios can be determined. This new simliarity ratios could be applied as alternative weights in the profiles-similarity graph 7.3.3.

In contrast to the task described here, that – restricted to matching XML schemas – can be seen as staying in the “XML World”, another aspect within this work is clearly situated in the Semantic Web domain and requires application of ontology matching methods – the mapping of field values to semantic entities described in 6.2.

5.7 Summary

In this core chapter, we layed out a design for a system dealing with concept-based crosswalks on schema level. The system consists of three main parts: the crosswalk service, the query expansion module and SMC Browser – a tool for visualizing and exploring the schemas and the corresponding crosswalks. In addition, we elaborated on the application of schema matching methods to infer mappings between schemas.

Chapter 6

Mapping on instance level, CMD as LOD

I do think that ISOcat, CLAVAS, RELcat and actual language resource all provide a part of the semantic network.

And if you can express these all in RDF, which we can for almost all of them (maybe except the actual language resource ... unless it has a schema adorned with ISOcat DC references ... <insert a SCHEMAcat plug ;-) >, but for metadata we have that in the CMDI profiles ...) you could load all the relevant parts in a triple store and do your SPARQL/reasoning on it. Well that's where I'm ultimately heading with all these registries related to semantic interoperability ... I hope ;-)[81]

As described in previous chapters (4,5), semantic interoperability is one of the main motivations for the CMD infrastructure. However, the established machinery pertains mostly to the schema level, the actual values in the fields of CMD instances remain "just strings". This is the case even though the problem of different labels for semantically equivalent or even identical entities is even more so virulent on the instance level. While for a number of metadata fields the value domain can be enforced through schema validation, some important fields (like organization or resource type) have a constrained value domain that yet cannot be explicitly exhaustively enumerated. This leads to a chronically inconsistent use of labels for referring to entities (as the instance data shows, some organizations are referred to by more than 20 different labels, or spelling variants.) prompting an urgent need for better means for harmonizing the constrained-field values.

One potential remedy is the use of reference datasets – controlled vocabularies, taxonomies, ontologies and such. In fact, this is a very common approach, be it the authority files in libraries world, or domain-specific reference vocabularies maintained by practically every research community. Not as strict as schema definitions, they cannot be used for validation, but still help to harmonize the data, by offering preferred labels and identifiers for entities.

In this chapter, we explore how this general approach can be employed for our specific problem of harmonizing the (literal) values in selected instance fields and mapping them to entities defined in corresponding vocabularies. This proposal is furthermore embedded in a more general effort to **express the whole of the CMD data domain (model and instances) in RDF** constituting one large ontology interlinked with existing external semantic resources (ontologies, knowledge bases, vocabularies). This result lays a foundation for providing the original dataset as a *Linked Open Data* nucleus within the *Web of Data*[29] as well as for real semantic (ontology-driven) search and exploration of the data.

The following section 6.1 lays out how individual parts of the CMD framework can be expressed in RDF. In 6.2 we investigate in further detail the abovementioned critical aspect of the effort, namely the task of translating the string values in metadata fields to corresponding semantic entities. Finally, the technical aspects of providing the resulting ontology as LOD and the implications for an ontology-driven semantic search are tackled briefly in 6.3 and **??** respectively.

6.1 CMD to RDF

In this section, RDF encoding is proposed for all levels of the CMD data domain:

- CMD meta model
- profile definitions
- the administrative and structural information of CMD records
- individual values in the fields of the CMD records

6.1.1 CMD specification

The main entity of the meta model is the CMD component and is typed as specialization of the `owl:Class`. CMD profile is basically a CMD component with some extra features, implying a specialization relation:

```
cmds:Component          subClassOf              owl:Class.
cmds:Profile            subClassOf              cmds:Component.
cmds:Element            subClassOf              rdf:Property.
```

This entities are used for typing the actual profiles, components and elements (as they are defined in the Component Registry):

```
cmd:collection          a                       cmds:Profile;
                        rdfs:label              "collection";
                        dcterms:identifier      cr:clarin.eu:cr1:p_1345561703620.
cmd:Actor               a                       cmds:Component.
cmd:LanguageName        a                       cmds:Element.
```

Should the ID assigned in the Component Registry for the CMD entities be used as identifier in RDF, or rather the verbose name? (if yes, how to ensure uniqueness – generate the name from the cmd-path?)

6.1.2 Data Categories

Windhouwer [87] proposes to use the data categories as annotation properties:

```
dcr:datcat              a                       owl:AnnotationProperty ;
                        rdfs:label              "data category"@en ;
                        rdfs:comment            "This resource is equivalent to this
                                                data category."@en ;
                        skos:note               "The data category should be
                                                identified by its PID."@en ;
```

That implies that the `@ConceptLink` attribute on CMD elements and components as used in the CMD profiles to reference the data category would be modelled as:

```
cmd:LanguageName        dcr:datcat              isocat:DC-2484.
```

Encoding data categories as annotation properties is in contrast to the common approach seen with dublincore terms used usually directly as data properties:

```
<lr1>                   dc:title                "Language Resource 1"
```

Analogously, we could model ISOcat data categories as data properties, i.e. metadata elements referencing ISOcat data categories could be encoded as follows:

```
<lr1>                   isocat:DC-2502          "19th century"
```

However, Windhouwer[87] argues against direct mapping of complex data categories to data properties and in favour of modelling data categories as annotation properties, so as to avoid too strong semantic implications.

This raises the vice-versa question, whether to rather handle all data categories uniformly, which would mean encoding dublincore terms also as annotation properties, but the pragmatic view dictates to encode the data in line with the prevailing approach, i.e. express dublincore terms directly as data properties.
The REST web service of ISOcat provides a RDF representation of the data categories:

```
isocat:languageName     dcr:datcat              isocat:DC-2484;
                        rdfs:label              "language name"@en;
                        rdfs:comment            "A human understandable..."@en;
                        ...
```

However this is only meant as template, as is stated in the explanatory comment of the exported data:

> By default the RDF export inserts `dcr:datcat` annotation properties to maintain the link between the generated RDF resources and the used Data Categories. However, it is possible to also maintain a stronger semantic link when the RDF resources will be used as OWL (2) classes, properties or individuals.

So in a specific (OWL 2) application the relation with the data categories can be expressed as `owl:equivalentClass` for classes, `owl:equivalentProperty` for properties or `owl:sameAs` for individuals:

```
#myPOS                  owl:equivalentClass     isocat:DC-1345.
#myPOS                  owl:equivalentProperty  isocat:DC-1345.
#myNoun                 owl:sameAs              isocat:DC-1333.
```

6.1.3 RELcat - Ontological relations

As described in 4.2.1 relations between data categories are not stored directly in the ISOcat DCR, but rather in a dedicated module the Relation Registry RELcat. The relations here are grouped into relation sets and stored as RDF triples[74]. A sample relation from the CMDI relation set expressing a number of equivalences between ISOcat data categories and dublincore terms:

```
isocat:DC-2538          rel:sameAs              dct:date
```

By design, the relations in Relation Registry are not expressed with predicates from known vocabularies like SKOS or OWL, again with the aim to avoid too strong semantic implications. This leaves leeway for further specialization of the relations in specific applications.

Does this mean, that I would say:

```
rel:sameAs              owl:equivalentProperty  owl:sameAs
```

to enable the inference of the equivalences?

Is this correct: ?? That means, that to be able to infer that a value in a CMD element also pertains to a given data category, e.g.:

```
  cmd:PublicationYear = 2012 →       dc:created = 2012
```

following facts need to be present in the ontology :

```
<lr1>                   cmd:PublicationYear     2012^^xs:year
cmd:PublicationYear     owl:equivalentProperty  isocat:DC-2538
isocat:DC-2538          rel:sameAs              dc:created
rel:sameAs              owl:equivalentProperty  owl:sameAs
→
<lr1>                   dc:created              2012^^xs:year
```

What about other relations we may want to express? (Do we need them and if yes, where to put them? – still in RR?) Examples:

```
cmd:MDCreator           owl:subClassOf          dcterms:Agent
clavas:Organization     owl:subClassOf          dcterms:Agent
<org1>                  a                       clavas:Organization
```

6.1.4 CMD instances

In the next step, we want to express the individual CMD instances, the metadata records, making use of the previously defined entities on the schema level, but also entities from external ontologies.

Resource Identifier

It seems natural to use the PID of a Language Resource (`<lr1>`) as the resource identifier for the subject in the RDF representation. While this seems semantically sound, not every resource has to have a PID. (This is especially the case for “virtual” resources like collections, that are solely defined by their constituents and don’t have any data on their own.) As a fall-back the PID of the MD record (`<lr1.cmd>` from `cmd:MdSelfLink` element) could be used as the resource identifier. If identifiers are present for both resource and metadata, the relationship between the resource and the metadata record can be expressed as an annotation using the OpenAnnotation vocabulary[1]:

```
_:anno1              a                    oa:Annotation;
                     oa:hasTarget         <lr1>;
                     oa:hasBody           <lr1.cmd>;
                     oa:motivatedBy       oa:describing
```

Provenance

The information from `cmd:Header` represents the provenance information about the modelled data:

```
<lr1.cmd>            dcterms:identifier   <lr1.cmd>;
                     dcterms:creator ??   "{cmd:MdCreator}";
                     dcterms:publisher    <http://clarin.eu>,
                                          <provider-oai-accesspoint>; ??
                     dcterms:created      "{cmd:MdCreated}" ??
                     /dcterms:modified?
```

Hierarchy (Resource Proxy – IsPartOf)

In CMD, the `cmd:ResourceProxyList` structure is used to express both collection hierarchy and point to resource(s) described by the MD record. This can be modelled as OAI-ORE Aggregation[2] :

```
<lr0.cmd>            a                    ore:ResourceMap
<lr0.cmd>            ore:describes        <lr0.agg>
<lr0.agg>            a                    ore:Aggregation
                     ore:aggregates       <lr1.cmd>, <lr2.cmd>;
```

?? Should both collection hierarchy and resource-pointers (collection and resource MD records) be encoded as ore:Aggregation? Additionally the flat header field `cmd:MdCollectionDisplayName` has been introduced to indicate by simple means the collection, of which given resource is part. This information can be used to generate a separate one-level grouping of the resources, in which the value from the `cmd:MdCollectionDisplayName` element would be used as the label of an otherwise undefined `ore:ResourceMap`. Even the identifier/ URI for this collections is not clear. Although this collections should match with the ResourceProxy hierarchy, there is no guarantee for this, thus a 1:1 mapping cannot be expected.

check consistency for MdCollectionDisplayName vs. IsPartOf in the instance data

[1] http://openannotation.org/spec/core/core.html#Motivations
[2] http://www.openarchives.org/ore/1.0/primer#Foundations

```
_:mdcoll            a                   ore:ResourceMap;
                    rdfs:label          "Collection 1";
_:mdcoll#aggreg     a                   ore:Aggregation
                    ore:aggregates      <lr1.cmd>, <lr2.cmd>;
```

Components – nested structures

There are two variants to express the tree structure of the CMD records, i.e. the containment relation between the components:

a) the components are encoded as object property

```
   <lr1>               cmd:Actor           _:Actor1
   <lr1>               cmd:Actor           _:Actor2
   _:Actor1            cmd:motherTongue    iso-639:aac
   _:Actor2            cmd:motherTongue    iso-639:deu
   _:Actor1            cmd:role            "Interviewer"
   _:Actor2            cmd:role            "Speaker"
```

b) a dedicated object property is used

```
   _:Actor1            a                   cmd:Actor
   <lr1>               cmd:contains        _:Actor1
```

6.1.5 Elements, Fields, Values

Finally, we want to integrate also the actual field values in the CMD records into the ontology.

Predicates

As explained before CMD elements are typed as `rdf:Property` with the corresponding data category expressed as annotation property:

```
cmd:timeCoverage    a                   cmds:Element
cmd:timeCoverage    dcr:datcat          isocat:DC-2502
<lr1>               cmd:timeCoverage    "19th century"
```

Literal values – data properties

To generate triples with literal values is straightforward:

Definition 6.1: Literal triples

$$lr : Resource \quad cmds : Property \quad xsd : string$$

```
<lr1>               cmd:Organisation    "MPI"
```

Mapping to entities – object properties

The more challenging but also more valuable aspect is to generate objectProperty triples with the literal values mapped to semantic entities:

Definition 6.2: new RDF triples

$$lr : Resource \quad cmd : Property \quad xsd : anyURI$$

```
<lr1>                 cmd:Organisation_?     <org1>
```

Don't we need a separate property (predicate) for the triples with object properties pointing to entities, i.e. `cmd:Organisation_` additionally to `cmd:Organisation`

The mapping process is detailed in 6.2

6.2 Mapping field values to semantic entities

This task is a prerequisite to be able to express also the CMD instance data in RDF. The main idea is to find entities in selected reference datasets (controlled vocabularies, ontologies) matching the literal values in the metadata records. The obtained entity identifiers are further used to generate new RDF triples. It involves following steps:

1. identify appropriate controlled vocabulares for individual metadata fields or data categories (manual task)
2. extract *distinct data category, value pairs* from the metadata records
3. actual **lookup** of the individual literal values in given reference data (as indicated by the data category) to retrieve candidate entities, concepts
4. assess the reliability of the match
5. generate new RDF triples with entity identifiers as object properties

This task is basically an application of ontology mapping method.

We don't try to achieve complete ontology alignment, we just want to find for our "anonymous" concepts semantically equivalent concepts from other ontologies. This is very near just other phrasing for the definition of ontology mapping function as given by [17, 16]: "for each concept (node) in ontology A [tries to] find a corresponding concept (node), which has the same or similar semantics, in ontology B and vice verse".

The first two points in the above enumeration represent the steps necessary to be able to apply the ontology mapping. The identification of appropriate vocabularies is discussed in the next subsection. In the operationalization, the identified vocabularies could be treated as one aggregated ontology to map all entities against. For the sake of higher precision, it may be sensible to perform the task separately for individual concepts, i.e. organisations, persons etc. and in every run consider only relevant vocabularies.

The transformation of the data has been partly described in previous section: It can be trivially automatically converted into RDF triples as :

```
<lr1>                 cmd:Organisation       "MPI"
```

However for the needs of the mapping task we propose to reduce and rewrite to retrieve distinct concept , value pairs:

```
_:1                   a                     cmd:Organisation;
                      skos:altLabel         "MPI";
```

lookup function is a customized version of the *(*map) function, that operates on this information pairs (concept, label).

The two steps *lookup* and *assess* correspond exactly to the two steps in [23] in their system LogMap2: 1) computation of mapping candidates (maximise recall) and b) assessment of the candidates (maximize precision)

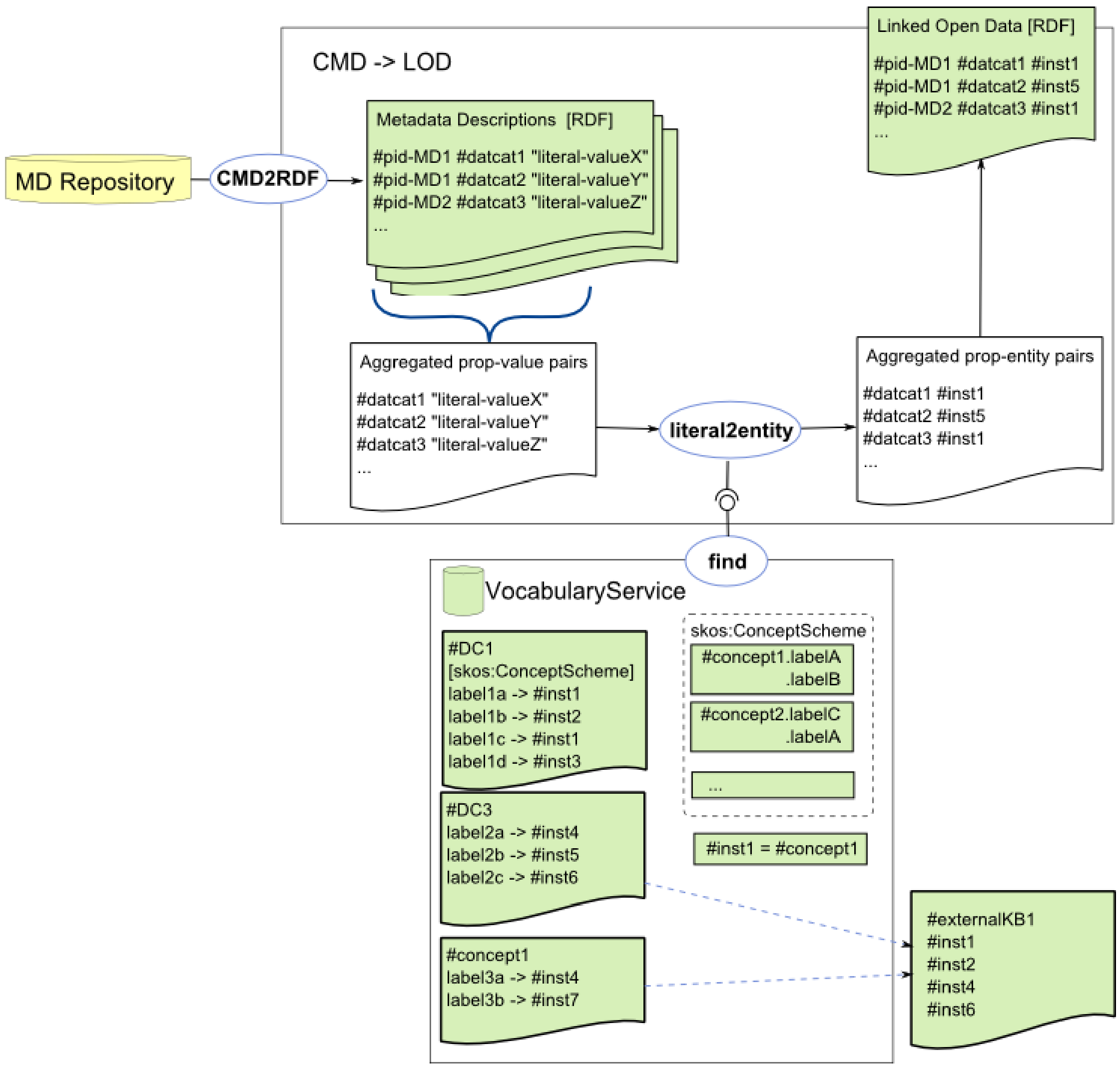


Figure 6.1: Sketch of the process of transforming the CMD metadata records to a RDF representation

Identify vocabularies

One generic way to indicate vocabularies for given metadata fields or data categories being discussed in the CMD community is to use dedicated annotation property (tentatively `@clavas:vocabulary`) in the schema or data category definition. For such a mechanism to work, the consuming applications (like metadata editor) need to be made aware of this convention and interpret it accordingly.

The primary provider of relevant vocabularies is ISOcat and CLAVAS – a service for managing and providing vocabularies in SKOS format (cf. 4.3.2). Closed and corresponding simple data categories are already being exported from ISOcat in SKOS format and imported into CLAVAS/OpenSKOS and also other relevant vocabularies shall be ingested into this system, so that we can assume OpenSKOS as a first source of vocabularies. However definitely not all of the existing reference data will be hosted by OpenSKOS, so in general we have to assume/consider a number of different sources (cf. 3.4).

Data in OpenSKOS is modelled purely in SKOS, so there is no more specific typing of the entities in the vocabularies, but rather all the entities are `skos:Concepts`:

```
<org1>                 a                    skos:Concept
```

We may want to add some more typing and introduce classes for entities from individual vocabularies like `clavas:Organization` or similar. As far as CLAVAS will also maintain mappings/links to other datasets

```
<org1>                 skos:exactMatch      <dbpedia/org1>, <lt-world/orgx>;
```

we could use it to expand the data with alternative identifiers, fostering the interlinking of data:

```
<org1>                 dcterms:identifier   <org1>, <dbpedia/org1>,
                                            <lt-world/orgx>;
```

Lookup

In abstract term, the lookup function takes as input the identifier of data category (or CMD element) and a literal string value and returns a list of potentially matching entities. Before actual lookup, there may have to be some string-normalizing preprocessing.

Definition 6.3: signature of the lookup function

$$lookup\ (\ DataCategory\ ,\ Literal\ )\quad \mapsto \quad (\ Concept \mid Entity\ )*$$

In the implementation there needs to be additional initial configuration input, identifying datasets for given data categories, which will be the result of the previous step.

Definition 6.4: Required configuration data indicating data category to available

$$DataCategory \quad \mapsto \quad Dataset+$$

As for the implementation, in the initial setup the system could resort to the `find`-interface provided by OpenSKOS. However, in the long term a more general solution is required, a kind of hybrid *vocabulary proxy service* that allows to search in a number of datasets, many of them distributed and available via different interfaces. Figure 6.2 sketches the general setup. The service has to be able to a) proxy search requests to a number of search interfaces (SRU, SPARQL), b) fetch, cache and search in datasets.

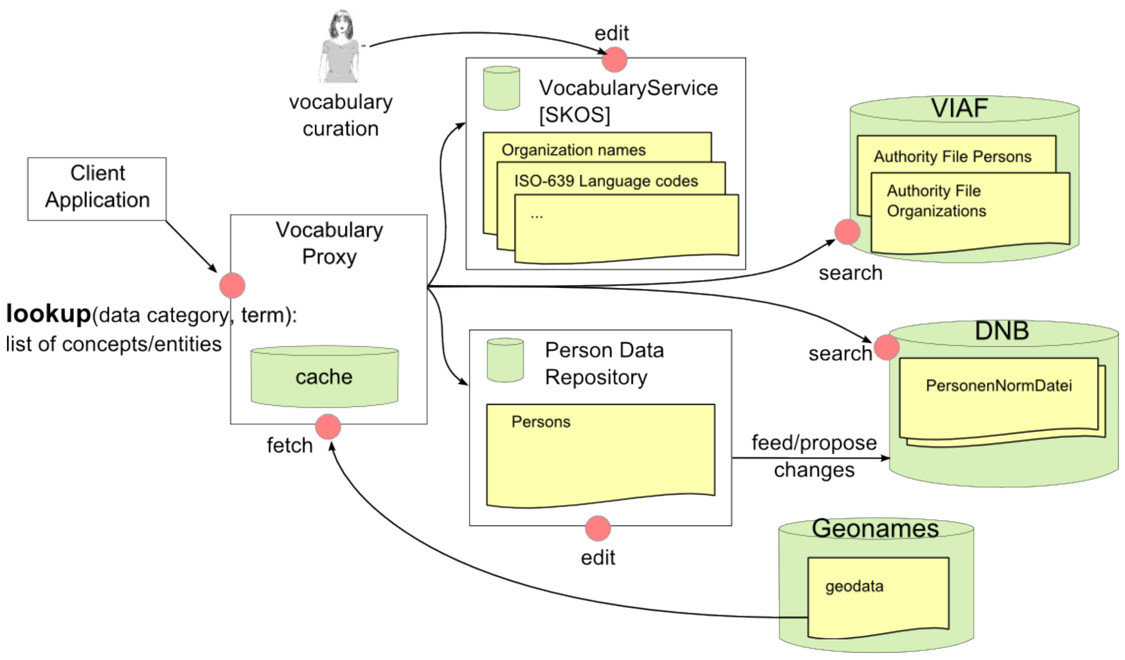


Figure 6.2: Sketch of a general setup for vocabulary lookup via a VocabularyProxy service

Candidate evaluation

The lookup is the most sensitive step in the process, as that is the gate between strings and semantic entities. In general, the resulting candidates cannot be seen as reliable and should undergo further scrutiny to ensure that the match is semantically correct.

One example: A lookup with the pair `<organization, "Academy of sciences">` would probably return a list of organizations, as there is a national Academy of Sciences, in a number of countries. It would require further heuristics, e.g. checking the corresponding department, contact or – less reliably – the language of the described resource, to determine which specific Academy of Sciences is meant in given resource description.

In some situation this ambiguities can be resolved algorithmically, but in the end in many cases it will require human curation of the generated data. In this respect, it is worth to note, that the CLARIN search engine VLO provides a feedback link, that allows even the normal user to report on problems or inconsistencies in CMD records.

6.3 SMC LOD - Semantic Web Application

With the new enhanced dataset, as detailed in section 6.1, the groundwork is laid for the full-blown semantic search as proposed in the original goals, i.e. the possibility of exploring the dataset using external semantic resources. The user can access the data indirectly by browsing external vocabularies/taxonomies, with which the data will be linked like vocabularies of organizations or taxonomies of resource types.

The technical base for a semantic web application is usually a RDF triple-store as discussed in 2.4. Given that our main concern is the data itself, their processing and display, we want to rely on stable, robust feature rich solution minimizing the effort to provide the data online. The most promising solution seems to be Virtuoso, a integrated feature-rich hybrid data store, able to deal with different types of data ("Universal Data Store").

Although the distributed nature of the data is one of the defining features of LOD and theoretically one should be able to follow the data by dereferencable URIs, in practice it is mostly necessary to pool into one data store linked datasets from different sources that shall be queried together due to performance reasons. This implies that the data to be kept by the data store will be decisively larger, than “just” the original dataset.

6.4 Summary

In this chapter, an expression of the whole of the CMD data domain into RDF was proposed, with special focus on the method to translate the string values in metadata fields to corresponding semantic entities. This task can be also seen as building a bridge between the world XML resources and semantic resources expressed in RDF. Additionally, some technical considerations were discussed regarding exposing this dataset as Linked Open Data and the implications for real semantic ontology-based data exploration.

Chapter 7

Results and Findings

In this chapter, the results of the work are presented. After a short update about the current state of affairs in the infrastructure as whole, the individual parts of the work are listed with pointers to their specifications in previous chapters and links to the running prototypes.

In the subsequent two sections, we explore a few specific aspects of the CMD data domain – regarding the usage of the data categories (7.3.1) and the integration of existing formats (7.3.2). While these topics are not directly results of this work, the presented analyses are. They were made possible by the technical solution of this work, yield a valuable test case for the usefulness of the work and are an indispensable prerequisite for the necessary coordination and curation work being carried out by the CMDI community.

7.1 Current status of the infrastructure

Before we get to the results of this work, we briefly summarize the current state of affairs within the CLARIN infrastructure at large to help contextualize the actual results.

7.1.1 CMDI - services

The main services of the infrastructure have been in stable production for the last two years. Relation Registry is operational as early prototype. Three instances of OpenSKOS are running, one of them being hosted by ACDH.

7.1.2 CMDI - data

More than 130 profiles are defined. (See table 3.1 for more details about profiles.) The official CLARIN harvester[1] collects data from 69 providers on daily basis. The collection amounts to over 550.000 records in more than 60 distinct profiles.

7.1.3 ACDH - the home of SMC

Within CLARIN-AT a new centre has been brought to life, the Austrian Centre for Digital Humanities with the mission to foster digital research paradigm in humanities. It is designed to provide depositing and publishing services to the DH community, as well as infrastructural services that are part of the CLARIN Metadata Infrastructure. SMC is one of these services provided by this centre. Figure B.3 sketches the broader context of ACDH and its different roles.

[1] http://catalog.clarin.eu/oai-harvester/

7.2 Technical solution

With this work we delivered a module embedded in a larger metadata infrastructure, aimed at supporting the semantic interoperability across the heterogeneous data in this infrastructure. The module consists of multiple interrelated components. The technical specification of the module can be found in chapter 5. A prototypical implementation has been developed for the three main parts of the system. The code of this implementation is maintained in the central CMDI code repository[2].

The module itself is hosted at the CLARIN-AT server, offering a main entry point page linking to the various parts of the module at:

http://clarin.aac.ac.at/smc (soon: http://acdh.ac.at/smc)

7.2.1 SMC - crosswalks service

the crosswalk service as a REST web service

exposes an interface that provides mappings between search indexes as defined in 5.3 This interface is available as part of the smc application:
http://clarin.aac.ac.at/smc/cx

7.2.2 SMC - as a module within Metadata Repository

The SMC is also integrated as module with the Metadata Repository enabling *semantic search* over the joint metadata domain.
http://clarin.aac.ac.at/mdrepo/smc

7.2.3 SMC Browser – advanced interactive user interface

SMC Browser is an advanced web-based visualization application to explore the complex dataset of the Component Metadata Infrastructure, by visualizing its structure as an interactive graph. In particular, it enables the metadata modeller to examine the reuse of components or DCs in different profiles. The graph is accompanied by numerical statistics about the dataset as whole and about individual items (profiles, components, data categories), a set of example results and user documentation. Details about design and implementation can be found in 5.5. The publicly available instance is maintained under:
http://clarin.aac.ac.at/smc/browser

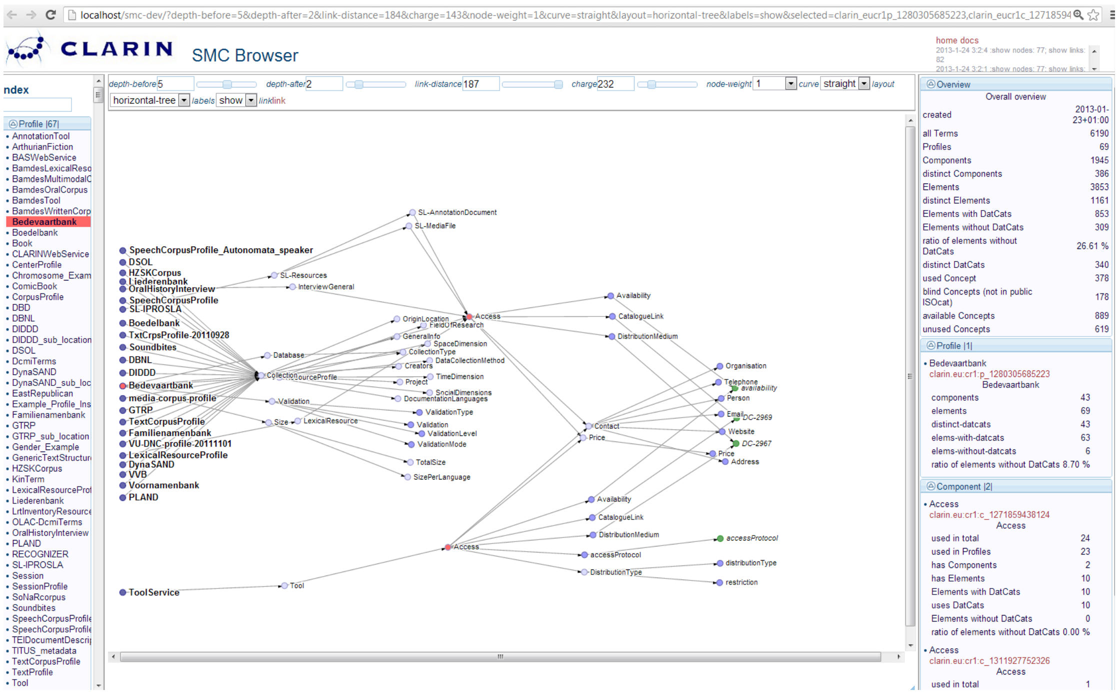
7.2.4 SMC LOD

In a separate track, a model has been proposed (cf. 6) to express CMD data in RDF, as first important step towards incorporating the dataset in the *Web of Data*.

7.3 Exploring the CMD data – SMC reports

SMC reports is a (growing) set of documents analyzing specific phenomena in the CMD data domain that were created making extensive use of the visual and numerical output from the SMC Browser. In this section, we deliver a few examples of these analyses. A complete up to date listing is maintained on the SMC website:
http://clarin.aac.ac.at/smc/reports

[2] http://svn.clarin.eu/SMC


localhost/smc-dev/?depth-before=5&depth-after=2&link-distance=184&charge=143&node-weight=1&curve=straight&layout=horizontal-tree&labels=show&selected=clarin_eucr1p_1280305685223,clarin_eucr1c_12718594

CLARIN SMC Browser

home docs
2013-1-24 3:2:4 :show nodes: 77; show links: 82
2013-1-24 3:2:1 :show nodes: 77; show links:

depth-before 5 · depth-after 2 · link-distance 187 · charge 232 · node-weight 1 · curve straight · layout horizontal-tree · labels show · linklink

ndex

Profile |67|
- AnnotationTool
- ArthurianFiction
- BASWebService
- BamdesLexicalReso
- BamdesMultimodalC
- BamdesOralCorpus
- BamdesTool
- BamdesWrittenCorp
- Bedevaartbank
- Boedelbank
- Book
- CLARINWebService
- CenterProfile
- Chromosome_Exam
- ComicBook
- CorpusProfile
- DBD
- DBNL
- DIDDD
- DIDDD_sub_location
- DSOL
- DcmiTerms
- DynaSAND
- DynaSAND_sub_loc
- EastRepublican
- Example_Profile_Ins
- Familienamenbank
- GTRP
- GTRP_sub_location
- Gender_Example
- GenericTextStructur
- HZSKCorpus
- KinTerm
- LexicalResourceProf
- Liederenbank
- LrtInventoryResource
- OLAC-DcmiTerms
- OralHistoryInterview
- PLAND
- RECOGNIZER
- SL-IPROSLA
- Session
- SessionProfile
- SoNaRcorpus
- Soundbites
- SpeechCorpusProfile
- SpeechCorpusProfile
- TEIDocumentDescri
- TITUS_metadata
- TextCorpusProfile
- TextProfile
- Tool

Overview

Overall overview

created	2013-01-23+01:00
all Terms	6190
Profiles	69
Components	1945
distinct Components	386
Elements	3853
distinct Elements	1161
Elements with DatCats	853
Elements without DatCats	309
ratio of elements without DatCats	26.61 %
distinct DatCats	340
used Concept	378
blind Concepts (not in public ISOcat)	178
available Concepts	889
unused Concepts	619

Profile |1|

- Bedevaartbank
  clarin.eu:cr1:p_1280305685223
  Bedevaartbank

components	43
elements	69
distinct-datcats	43
elems-with-datcats	63
elems-without-datcats	6
ratio of elements without DatCats	8.70 %

Component |2|

- Access
  clarin.eu:cr1:c_1271859438124
  Access

used in total	24
used in Profiles	23
has Components	2
has Elements	10
Elements with DatCats	10
uses DatCats	10
Elements without DatCats	0
ratio of elements without DatCats	0.00 %

- Access
  clarin.eu:cr1:c_1311927752326
  Access

used in total	1

Figure 7.1: Screenshot of the SMC browser

7.3.1 Usage of data categories

At the core of the whole SMC (and CMDI) are the data categories as basic semantic building blocks or anchors.

In the ISOcat DCR, currently 791 DCs are defined in the Metadata thematic profile, starting from 222 that were initially created by the so-called *Athens Core* group in 2010. As can be seen in table 3.1, around 500 distinct data categories are being used in CMD profiles. We want to take a closer look on the usage of the data categories in the CMD data domain, examplifying on the very common concepts – language, name.

Language

While there are 69 components and 97 elements containing a substring `'language'` defined in the CR still only 19 distinct DCs with a `'language'` substring are being used[3]. The most commonly used ones: languageID#DC-2482) and languageName#DC-2484) are referenced by more than 80 profiles. Additionally, these two DCs are linked to the Dublin Core term Language in the RR. Thus a search engine capable of interpreting RR information could offer the user a simple Dublin Core-based search interface, while – by expanding the query – still searching over all available data, and, moreover, on demand offer the user a more finegrained semantic interpretation for the matches based on the originally assigned DCs. Figure 7.2 depicts the relations between the language data categories and their usage in the profiles. We encounter all types of situations: profiles using only dc:Language or dcterms:Language, isocat:languageId or isocat:languageName, most profiles use both isocat:languageId and isocat:languageName and there are even profiles that refer to both isocat and dublincore data categories (data, HZSKCorpus, ToolService).

It requires further inspection and in the end a case by case decision, if the other less often used 'language' DCs can be treated as equivalent to the above mentioned ones. languageScript, implementationLanguage, as well as noLanguages or sizePerLanguage

[3]Here the term 'used' means referenced in CMD components and elements.

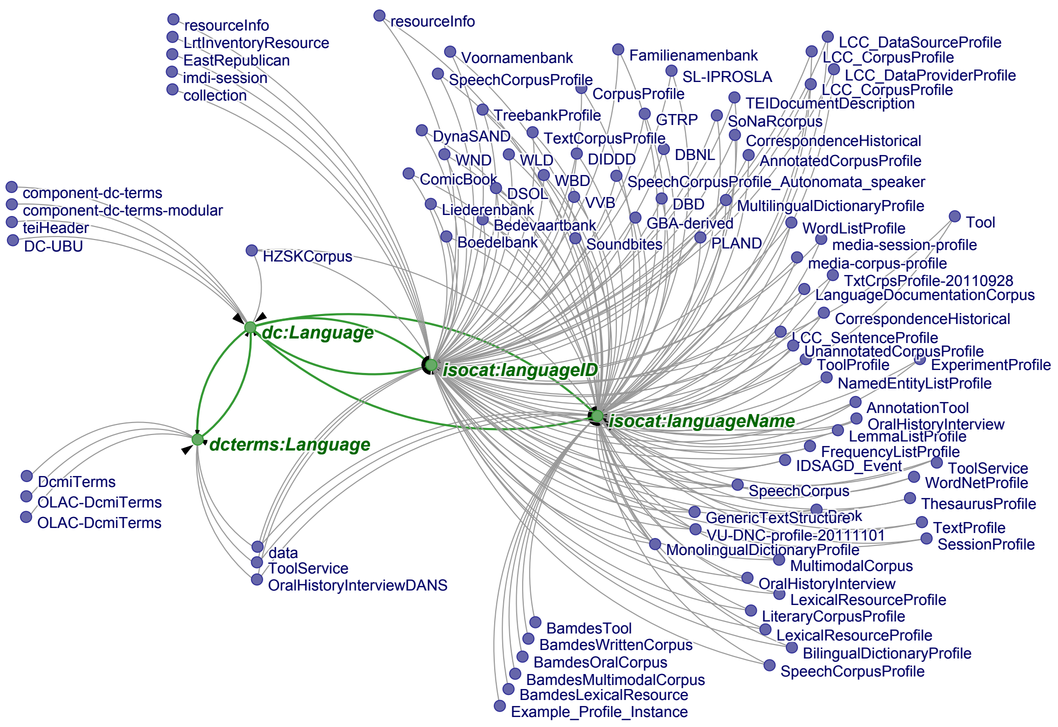


Figure 7.2: The four main Language data categories and in which profiles they are being used

clearly do not belong to the language cluster. But sourceLanguage, languageMother or participantDominantLanguage can at least be expected to share the same value domain (natural languages) and even if they do not describe the language of the resource, they could be considered when one aims at maximizing the recall (i.e., trying to find anything related to a given language). This is actually exactly the scenario the RR was conceived for – allow to define custom relation sets based on specific needs of a project or of a research question.

Name / Title

There are as many as 72 CMD elements with the label `Name`, referring to 12 different DCs. Again the main DC resourceName#DC-2544) being used in 74 profiles together with the semantically near resourceTitle#DC-2545) used in 69 profiles offer a good coverage over available data.

Some of the DCs referenced by `Name` elements are author#DC-4115), contact full name#DC-2454), dcterms:Contributor, project name#DC-2536), web service name#DC-4160) and language name#DC-2484). This implies, that a naïve search in a `Name` element would match semantically very heterogeneous fields and only applying the semantic information provided by the DCs and/or the context of the element (the enclosing components) allows to disambiguate the meaning of the values.

7.3.2 Integration of existing formats

CLARIN set out with the aspiration /yearning to overcome the babylon of metadata formats and its flexible CMD metamodel is specifically designed to integrate existing

Table 7.1: Profiles modelling dublincore terms

profile name	*created*	*creator*	*count*	*instances*
component-dc-terms-modular	2010-04	CMDI-team	15 / 15 / 15	
component-dc-terms	2010-04	CMDI-team	0 / 15 / 15	
DcmiTerms	2010-10	D.Van Uytvanck	0 / 55 / 55	46.156
OLAC-DcmiTerms	2010-10	D. Van Uytvanck	0 / 55 / 55	85.149
OLAC-DcmiTerms[4]	2013-02	M. Windhouwer	1 / 71 / 62	
DC-UBU	2013-05	Utrecht Uni Lib	0 / 15 / 15	
OLAC-DcmiTerms-ref	2013-06	Fankhauser, IDS	0 / 55 / 55	697
OLAC-DcmiTerms-ref-DWR	private	?	1 / 61 / 55	775

formats. In this section, we want to elaborate on/analyze the state of integration efforts for 4 major formats: dublincore/OLAC, teiHeader and META-SHARE resourceInfo.

dublincore / OLAC

Very widely used (because) simple format 3.2.2

Here the problem of proliferation seems especially virulent. Table 7.1 lists all the profiles modelling dcterms. As all these profiles are link to the corresponding dublincore data categories, this does not pose a major problem on the exploitation side, however the cluttering of the component registry with structurally identical or almost identical profiles needs to be questioned within the community.

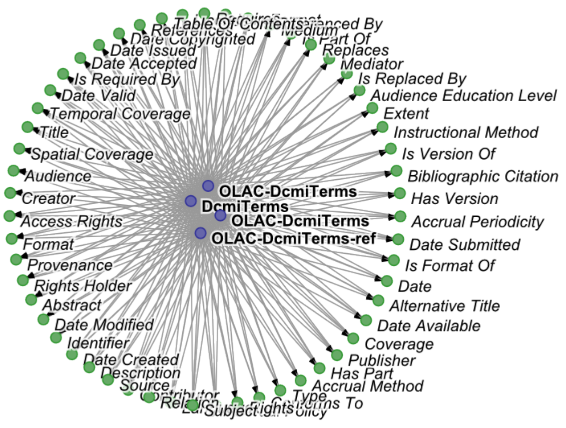


Figure 7.3: The meanwhile four DCMI profiles with identical conceptual linking

Additionally, there is a number of profiles with concept links to dublincore terms, Some use all of the dublincore elements or terms as one component within a larger profile, one example being the data profile created by the Czech initiative LINDAT models the minimal obligatory set of META-SHARE resourceInfo schema, cf. subsection about META-SHARE below) combined with a simple dublincore record. Other profiles refer only to some data categories. Most often used: dc:Title (used in 33 profiles) and dc:Creator (in 29 profiles). Profiles that make more frequent use of the dublincore terms:

EastRepublican	8
HZSKCorpus	17
teiHeader	8
ToolService	15
OralHistoryInterviewDANS	15

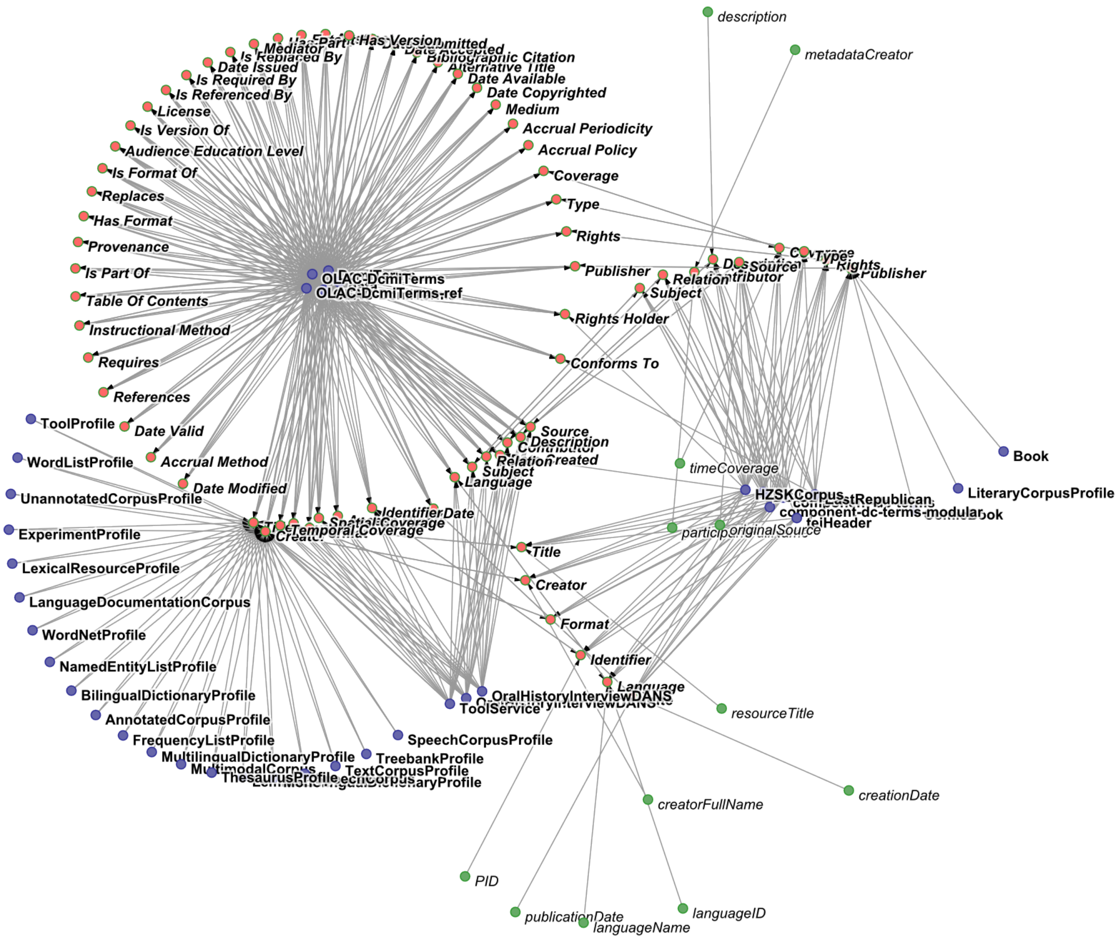


Figure 7.4: Profiles referring to at least some of the dublincore data categories/terms

teiHeader

TEI is a de-facto standard for encoding any kind of textual resources. It defines a set of elements to annotate individual aspects of the text being encoded. For the purposes of text description / metadata the complex element `teiHeader` is foreseen. TEI does not provide just one fixed schema, but allows for a certain flexibility wrt to elements used and inner structure, allowing to generate custom schemas adopted to projects' needs. 3.2.3. Thus there is also not just one fixed teiHeader.

The widespread use of TEI for encoding textual resources brings about a strong interest of multiple research teams of the CLARIN community to integrate TEI with CMDI. There was a first attempt already in 2010, modelling the recommended teiHeader[5], encoding fileDesc and profileDesc components, leaving out encodingDesc and revisionDesc. The leaf elements were bound to the most prominent data categories, making it a mixture of both dublincore and isocat.

The large research project Deutsches Textarchiv[6][61], digitizing a hoist of historical german texts from the period 1650 - 1900 also uses TEI to encode the material and consequently the teiHeader to hold the metadata information. Part of the project is also to integrate the data and metadata with the CLARIN infrastructure, meaning CMD records need to be generated for the resources. For this the team generated a completely new profile (as yet private) closely modelling the version of the teiHeader[7] used in the project. Regarding the question, why another teiHeader-based profile was generated not reusing the existing one, according to a personal note by a member of the project team and author of the profile, Axel Herold[88] the profile was custom made for this particular project and it seemed undesirable to create a generalised TEI header profile.

Nederlab is another large-scale project aiming processing historic Dutch newspaper articles into a platform for search and analysis, starting 2013 in Netherlands[8]. Within this project, the metadata is also encoded in a teiHeader and the data shall be integrated within CLARIN. Here, another set of CMD profiles was created, however reusing existing components. As seen in figure 7.5, components fileDesc and profileDesc were reused, while the components encodingDesc and revisionDesc, left out in the original profile, were added.

Another approach was applied within the context of other CLARIN-NL projects[89]. Based on an ODD-file, a data category for every element of the teiHeader (135 datcats) was generated. In a subsequent step, an enriched schema was generated, that remodells the original teiHeader-schema, but with the individual elements being annotated with the new data categories (`dcr:datcat`-attribute). This schema is now maintained in the SCHEMAcat (cf. 4). The next step would be to create again a new profile, but with all the components and elements in it bound to the corresponding data categories, moving the semantic linking into the relation registry, where appropriate relations could be defined between the data categories derived from TEI and the isocat and/or dublincore DCs. This yields a more complex, but also a more systematic and flexible setup, with a clean separation/boundary/interface of the semantic space of TEI and the possibility to map the TEI elements (via their data categories) to multiple and/or different data categories according to the specific needs of a project or research question.

[5]http://www.tei-c.org/release/doc/tei-p5-doc/en/html/HD.html#HD7
[6]http://deutschestextarchiv.de/
[7]http://www.deutschestextarchiv.de/doku/basisformat_header
[8]http://www.nederlab.nl

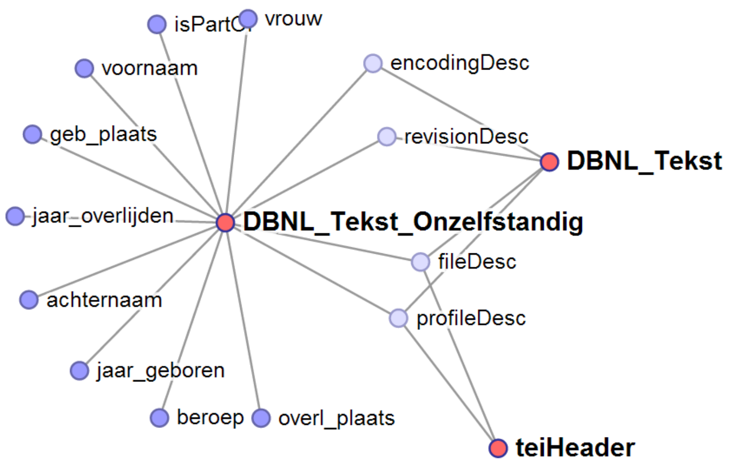


Figure 7.5: The reuse of components between the original teiHeader-profile (2010) and the profiels used in Nederlab project

Table 7.2: Overview of TEI-related CMD profiles

profile name	*created*	*creator*	*count*	*instances*
teiHeader	2010	Durco, ICLTT	16/35/13	467
teiHeader	2012	Deutsches Text Archiv	56/82/10	857
TEIDocument Description	2012	Eckart, Leipzig Corpora	16/35/13	?
DBNL_Tekst	2013	Zhang, Nederlab	20/38/15	>37 Mio.[9]
DBNL_Tekst_ Onzelfstandig			20/47/21	

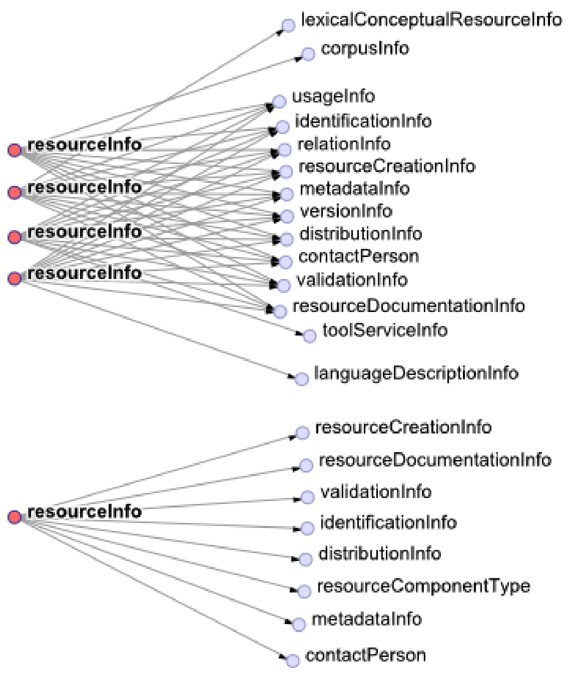


Figure 7.6: The five resourceInfo profiles with the first level of components

META-SHARE

META-SHARE created a new metadata model [55]. Although inspired by the Component Metadata, META-SHARE metadata imposes a single large schema for all resource types with a minimal core subset of obligatory metadata elements and with many optional components.

The original META-SHARE schema actually accomodates four models for different resource types. Consequently, the model has been expressed as 4 CMD profiles each for a distinct resource type however all four sharing most of the components, as can be seen in figure 7.6. The biggest single profile is currently the remodelled maximum schema from the META-SHARE project for describing corpora, with 117 distinct components and 337 elements. When expanded, this translates to 419 components and 1587 elements. However, many of the components and elements are optional (and conditional), thus a specific instance will never use all the possible elements.

In a parallel effort, LINDAT, the czech national infrastructure initiative engaged in both CLARIN and META-SHARE, created a CMD profile (data[10]) modelling the minimal obligatory set of META-SHARE resourceInfo), however combined with a simple dublincore record. This way, the information gets partly duplicated, but with the advantage, that a minimal information is conveyed in the widely understood format, retaining the expressivity of the feature-rich schema.

The expression of the META-SHARE schema in CMD allows a direct comparison of the two different approaches taken in the two projects: a metamodel allowing to generate custom profiles with shared semantics vs. the more traditional way of trying to generate one schema to fit in all the information. It shows nicely the trade-off: many custom schemas with the risk of proliferation and problems with semantic interoperability or one very large with the risk of overwhelming the user and still not being able to capture all specific informations.

[10] http://catalog.clarin.eu/ds/ComponentRegistry/?item=clarin.eu:cr1:p_1349361150622

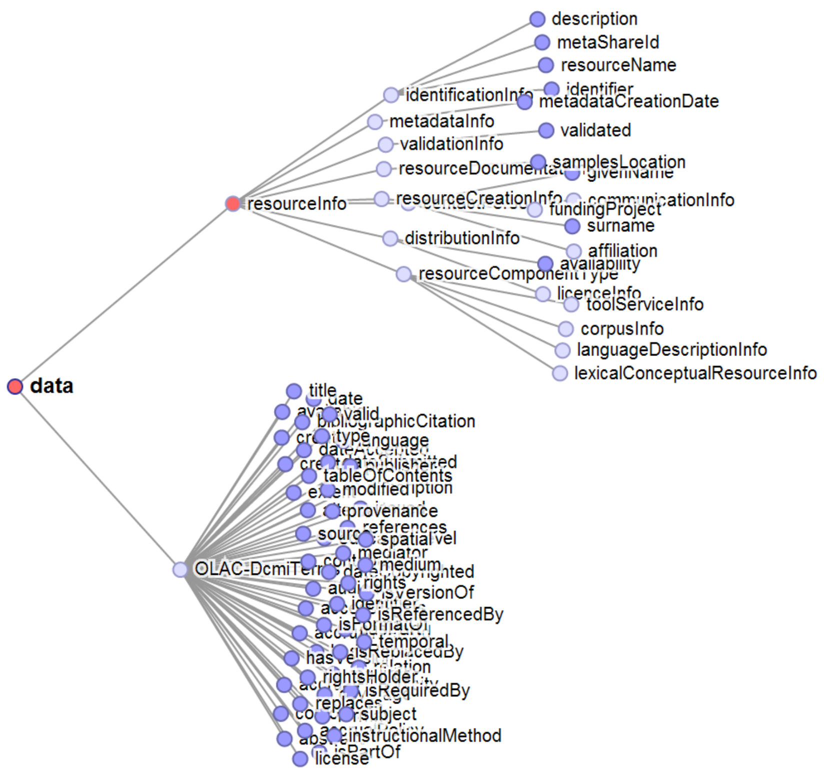


Figure 7.7: profile by LINDAT combining META-SHARE resourceInfo component with dublincore elements

Table 7.3: Profiles modelling resourceInfo

profile name	*created*	*creator*	*count*	*instances*
resourceInfo (minimal)	2013-02	LINDAT.CZ	34 / 41 / 21	67
resourceInfo (lexical)	2013-06	P. Labropoulou	86 / 226 / 57	
resourceInfo (tools)	2013-06	P. Labropoulou	61 / 176 / 52	
resourceInfo (language)	2013-06	P. Labropoulou	89 / 228 / 54	
resourceInfo (corpus)	2013-06	P. Labropoulou	117 / 337 / 72	

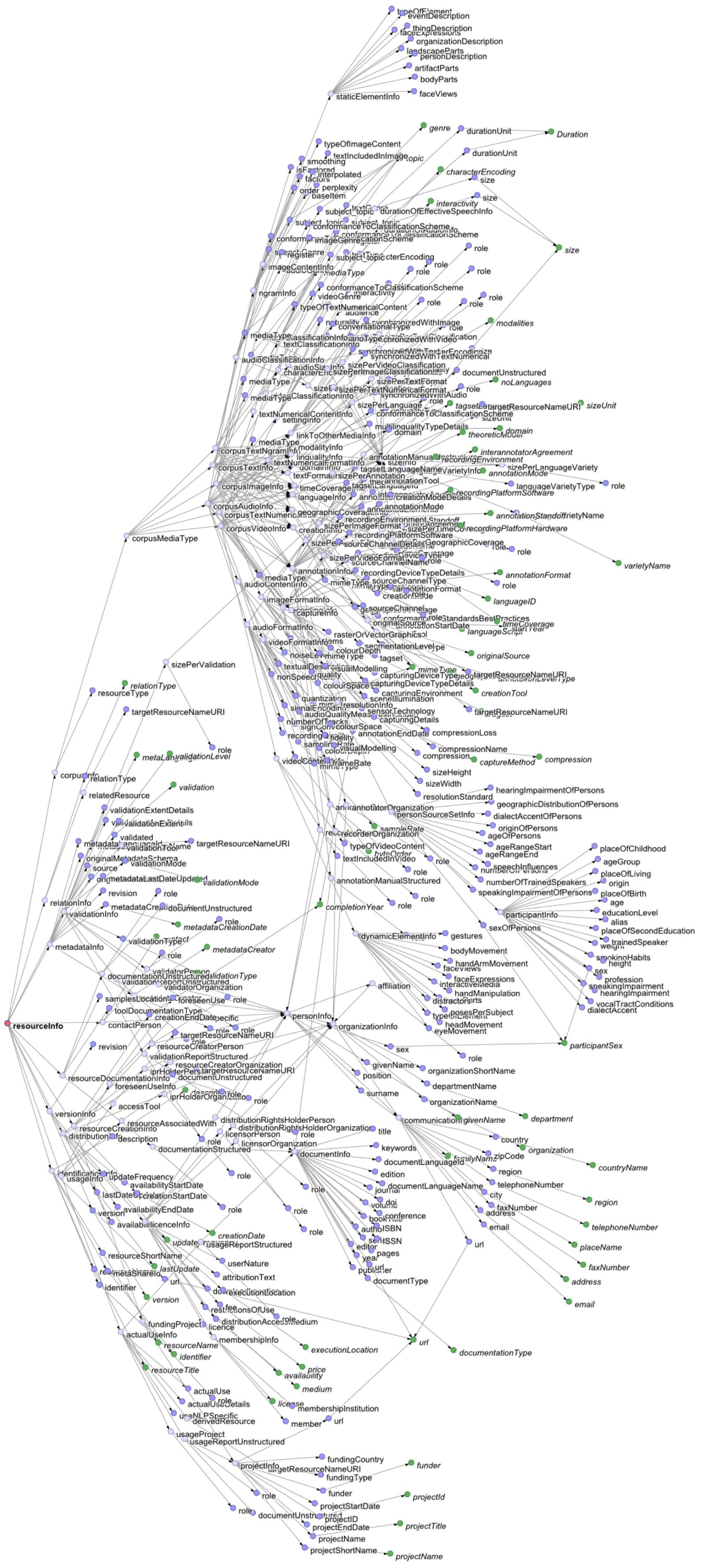


Figure 7.8: the META-SHARE based profile for describing corpora

7.3.3 SMC cloud

As a latest, still experimental, addition, SMC browser provides a special type of graph, that displays only profiles. The links between them reflect the reuse of components and data categories (i.e. how many components or data categories do the linked pairs of profiles share), indicating the degree of similarity or semantic proximity. Figure 7.9 depicts one possible output of the graph covering a large part of the defined profiles. It shows nicely the clusters of strongly related profiles in contrast to the greater distances between more loosely connected profiles.

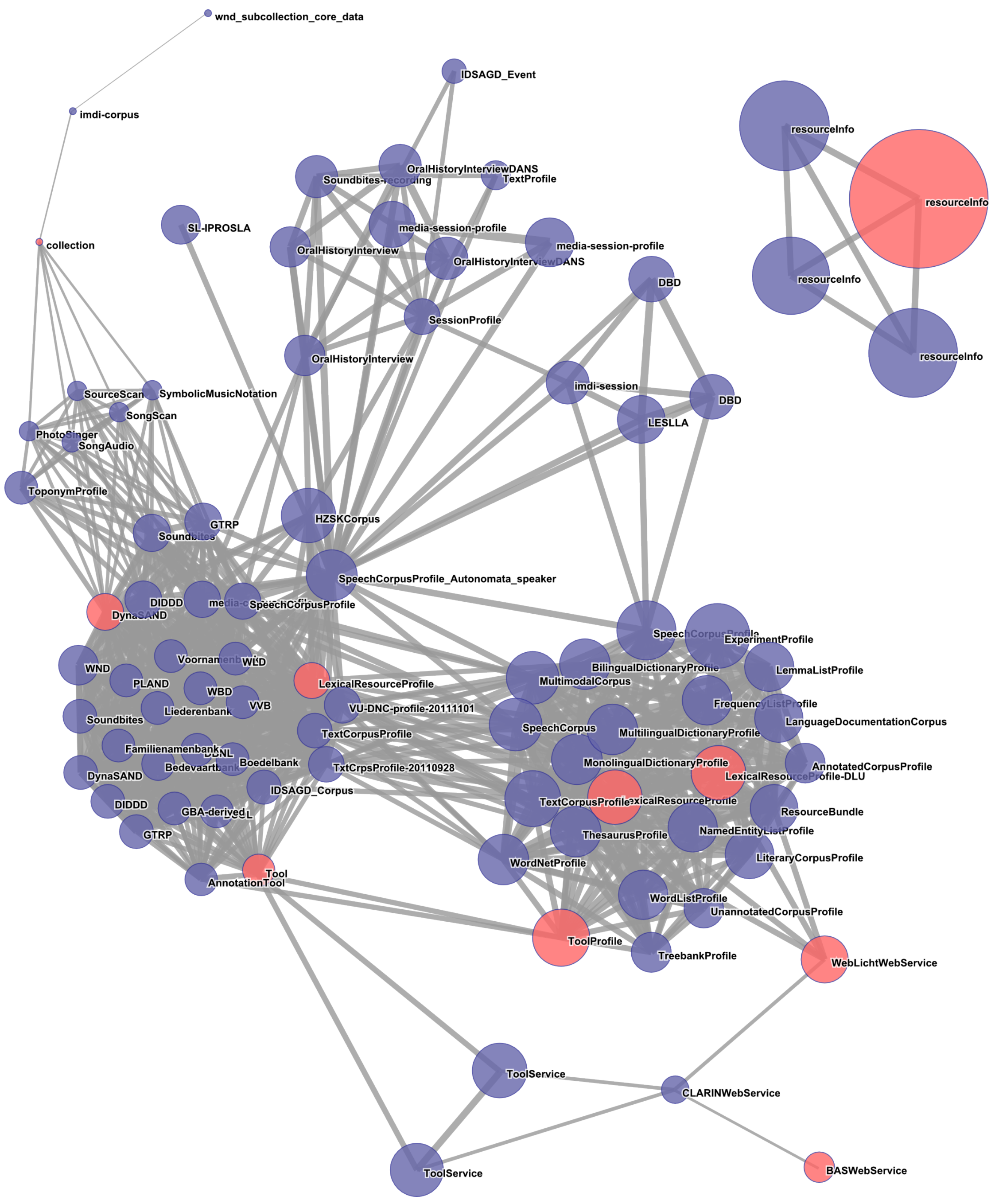


Figure 7.9: SMC cloud – graph visualizing the semantic proximity of profiles

7.4 Summary

In this final chapter, we presented the results, on the one hand the technical solution of the module Semantic Mapping Component, on the other hand we spent a good part of the chapter on commented analyses of the processed dataset, that were made possible by SMC Browser, a interactive visualization tool developed as part of this work for exploration of the schema level data of the discussed collection. As such, the analyses can be seen as an evaluation, a proof of concept and usefulness of the presented work.

Chapter 8

Conclusions and Future Work

With this work, a technical description together with a prototypical implementation for the *Semantic Mapping Component* was delivered – one module within an infrastructure for providing metadata, the *Component Metadata Infrastructure*.

SMC features a concept-based crosswalk service providing correspondences between fields in metadata formats and a module for query expansion building on top of it, allowing concept-based semantic search. Further work is needed on the crosswalk service providing more complex types of response (similarity ratio, relation types) with implications for the query expansion module. The integration of the semantic mapping features in the search user interface is only rudimentary at present, calling for a more elaborate solution.

A whole separate track is the effort to deliver the CMD data as *Linked Open Data*, for which only the groundwork has been done by specifying the modelling of the data in RDF. Further steps are: setup of a processing workflow to apply the specified model and transform all the data (profiles and instances) into RDF, a server solution to host the data and allow querying it and finally, on top of it offer a web interface for the users to explore the dataset.

And finally, a visualization tool for the schema level data of the discussed data collection was developed – the *SMC Browser*. Considering the feedback received until now from the colleagues in the community, it is already now a useful tool with high further potential. As detailed in 5.5.3, there is a number of features, that could enhance the functionality and usefulness of the tool: integrate with instance data to be able to directly see which profiles are effectively being used; allow set operations on subgraphs (like intersection and difference) to enable differential views; generalize the matching algorithm; enhance the tool to act as an independent visualization service, by accepting external graph data (from any domain).

Within the CLARIN community a number of (permanent) tasks has been identified and corresponding task forces have been established, one of them being metadata curation. The results of this work represent a directly applicable groundwork for this ongoing effort. One particularly pressing aspect of the curation is the consolidation of the actual values in the CMD records, a topic explicitly treated in this work.

Bibliography

[1] P. Wittenburg, D. Broeder, and B. Sloman, "Eagles/isle: A proposal for a meta description standard for language resources, white paper," in *LREC 2000 Workshop, Athens*, 2000.

[2] D. Broeder, O. Schonefeld, T. Trippel, D. Van Uytvanck, and A. Witt, "A pragmatic approach to xml interoperability - the component metadata infrastructure (cmdi)," in *Balisage: The Markup Conference 2011*, vol. 7, (Montréal, Canada), 2011. citeulike:9861691.

[3] S. Piperidis, "The meta-share language resources sharing infrastructure: Principles, challenges, solutions," in *Proceedings of LREC 2012*, pp. 36–42, 2012.

[4] J. G. Fenly and B. Wiggins, *The Linked Systems Project: a networking tool for libraries.* Columbus, OH, USA: OCLC (Online Computer Library Center), 1988. Fenly:1988:LSP:60988.

[5] OCLC, "Annual report," tech. rep., OCLC, 2012.

[6] J. Purday, "Think culture: Europeana. eu from concept to construction," *Electronic Library, The*, vol. 27, no. 6, pp. 919–937, 2009.

[7] S. Jänicke, C. Heine, and G. Scheuermann, "Geotemco: Comparative visualization of geospatial-temporal data with clutter removal based on dynamic delaunay triangulations," in *Computer Vision, Imaging and Computer Graphics. Theory and Application*, pp. 160–175, Springer, 2013.

[8] M. Dayh, "Mapping between metadata formats." online, 2002.

[9] P. Shvaiko and J. Euzenat, "Ontology matching: state of the art and future challenges," 2012.

[10] Y. Kalfoglou and M. Schorlemmer, "Ontology mapping: the state of the art," *The Knowledge Engineering Review*, vol. 18, pp. 1–31, Jan. 2003.

[11] P. Shvaiko and J. Euzenat, "Ten challenges for ontology matching," in *On the Move to Meaningful Internet Systems: OTM 2008* (R. Meersman and Z. Tari, eds.), vol. 5332 of *Lecture Notes in Computer Science*, pp. 1164–1182, Springer Berlin / Heidelberg, 2008. 10.1007/978-3-540-88873-4_18.

[12] N. Noy and H. Stuckenschmidt, "Ontology alignment: An annotated bibliography," in *Semantic Interoperability and Integration - Schloss Dagstuhl*, 2005.

[13] N. F. Noy, "Semantic integration: A survey of ontology-based approaches," *SIGMOD Record*, vol. 33, p. 2004, 2004.

[14] P. Shvaiko and J. Euzenat, "A survey of schema-based matching approaches," tech. rep., 2005.

[15] S. Amrouch and S. Mostefai, "Survey on the literature of ontology mapping, alignment and merging," in *Information Technology and e-Services (ICITeS), 2012 International Conference on*, pp. 1–5, IEEE, 2012.

[16] M. Ehrig and Y. Sure, "Ontology mapping – an integrated approach," in *The Semantic Web: Research and Applications* (C. Bussler, J. Davies, D. Fensel, and R. Studer, eds.), vol. 3053 of *Lecture Notes in Computer Science*, pp. 76–91, Springer Berlin / Heidelberg, 2004. 10.1007/978-3-540-25956-5_6.

[17] M. Ehrig, *Ontology Alignment: Bridging the Semantic Gap (Semantic Web and Beyond).* Springer, 2006. PhD thesis.

[18] J. Euzenat and P. Shvaiko, *Ontology Matching.* Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007. Euzenat:2007:OM:1203689.

[19] M. Ehrig and S. Staab, "Qom–quick ontology mapping," in *The Semantic Web–ISWC 2004*, pp. 683–697, Springer, 2004.

[20] S. Melnik, H. Garcia-Molina, and E. Rahm, "Similarity flooding: A versatile graph matching algorithm and its application to schema matching," in *Data Engineering, 2002. Proceedings. 18th International Conference on*, pp. 117–128, IEEE, 2002.

[21] A. Algergawy, R. Nayak, and G. Saake, "Element similarity measures in {XML} schema matching," *Information Sciences*, vol. 180, no. 24, pp. 4975 – 4998, 2010.

[22] E. Jiménez-Ruiz, B. C. Grau, Y. Zhou, and I. Horrocks, "Large-scale interactive ontology matching: Algorithms and implementation.," in *ECAI*, pp. 444–449, 2012.

[23] Y. Kalfoglou and M. Schorlemmer, "If-map: An ontology-mapping method based on information-flow theory," in *Journal on data semantics I*, pp. 98–127, Springer, 2003.

[24] M. Ehrig and Y. Sure, "Foam - framework for ontology alignment and mapping results of the ontology alignment evaluation initiative," *System*, vol. 156, pp. 72–76, 2005.

[25] F. Giunchiglia, M. Yatskevich, and E. Giunchiglia, "Efficient semantic matching," in *The Knowledge Engineering Review*, p. 2003, Springer, 2007.

[26] N. F. Noy and M. A. Musen, "The prompt suite: Interactive tools for ontology merging and mapping," *International Journal of Human-Computer Studies*, vol. 59, p. 2003, 2003.

[27] D. Aumueller, H.-H. Do, S. Massmann, and E. Rahm, "Schema and ontology matching with coma++," in *IN SIGMOD CONFERENCE*, pp. 906–908, ACM Press, 2005.

[28] T. Berners-Lee, "Linked data." online: http://www.w3.org/DesignIssues/LinkedData.html, 07 2006. Status: personal view only. Editing status: imperfect but published. Last visited: 2011-04-13.

[29] C. Bizer, T. Heath, and T. Berners-Lee, "Linked data-the story so far," *International Journal on Semantic Web and Information Systems (IJSWIS)*, vol. 5, no. 3, pp. 1–22, 2009.

[30] T. Heath and C. Bizer, "Linked data: Evolving the web into a global data space," *Synthesis Lectures on the Semantic Web: Theory and Technology*, vol. 1, pp. 1–136, Feb 2011.

[31] B. Haslhofer, E. M. Roochi, B. Schandl, and S. Zander, "Europeana rdf store report," technical report, University of Vienna, Vienna, March 2011.

[32] O. Erling and I. Mikhailov, "Rdf support in the virtuoso dbms," in *Networked Knowledge-Networked Media*, pp. 7–24, Springer, 2009.

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

[34] S. Noah, N. Alias, N. Osman, Z. Abdullah, N. Omar, Y. Yahya, and M. Yusof, "Ontology-driven semantic digital library," in *Information Retrieval Technology* (P.-J. Cheng, M.-Y. Kan, W. Lam, and P. Nakov, eds.), vol. 6458 of *Lecture Notes in Computer Science*, pp. 141–150, Springer Berlin / Heidelberg, 2010. 10.1007/978-3-642-17187-1_13.

[35] S. Pouyllau, "Isidore: une plateforme de recherche de documents et d'information pour les sciences humaines et sociales," tech. rep., Huma-Num, 2011.

[36] S. Farrar and T. Langendoen, "A linguistic ontology for the semantic web," *GLOT INTERNATIONAL*, vol. 7, 2003. GOLD.

[37] S. Farrar and D. T. Langendoen, "General ontology for linguistic description (gold)." online dataset (OWL), 2010.

[38] B. F. Grimes, J. E. Grimes, and S. I. of Linguistics, *Ethnologue.* SiL international Dallas, TX, USA, 2000.

[39] M. S. Dryer and M. Haspelmath, eds., *The World Atlas of Language Structures Online.* Munich: Max Planck Digital Library, 2011 ed., 2011.

[40] G. Simons, "Developing markup metaschemas to support interoperation among resources," *ALLC/ACH 2003*, p. 141, 2003.

[41] G. F. Simons, W. D. Lewis, S. O. Farrar, D. T. Langendoen, B. Fitzsimons, and H. Gonzalez, "The semantics of markup: Mapping legacy markup schemas to a common semantics," in *Proceeedings of the Workshop on NLP and XML (NLPXML-2004): RDF/RDFS and OWL in Language Technology*, pp. 25–32, Association for Computational Linguistics, 2004.

[42] G. Hirst, "Ontology and the lexicon," in *Handbook on Ontologies* (P. Bernus, J. Błażewics, G. Schmidt, M. Shaw, S. Staab, and R. Studer, eds.), International Handbooks on Information Systems, pp. 269–292, Springer Berlin Heidelberg, 2009. 10.1007/978-3-540-92673-3_12.

[43] P. Buitelaar, P. Cimiano, P. Haase, and M. Sintek, "Towards linguistically grounded ontologies," in *The Semantic Web: Research and Applications* (L. Aroyo, P. Traverso, F. Ciravegna, P. Cimiano, T. Heath, E. Hyvönen, R. Mizoguchi, E. Oren, M. Sabou, and E. Simperl, eds.), vol. 5554 of *Lecture Notes in Computer Science*, pp. 111–125, Springer Berlin / Heidelberg, 2009. 10.1007/978-3-642-02121-3_12.

[44] P. Buitelaar, "Ontology-based semantic lexicons: Mapping between terms and object descriptions," *Ontology and the Lexicon*, pp. 212–223, 2010.

[45] J. McCrae, G. A. de Cea, P. Buitelaar, P. Cimiano, T. Declerck, A. G. Pérez, J. Gracia, L. Hollink, E. M.-P. D. Spohr, and T. Wunner, *The lemon cookbook*. Monnet Project, 11 2010.

[46] P. Buitelaar, P. Cimiano, J. McCrae, E. Montiel-Ponsoda, and T. Declerck, "Ontology lexicalisation: the lemon perspective," 2011.

[47] J. McCrae, G. Aguado-de Cea, P. Buitelaar, P. Cimiano, T. Declerck, A. Gómez-Pérez, J. Gracia, L. Hollink, E. Montiel-Ponsoda, D. Spohr, *et al.*, "Interchanging lexical resources on the semantic web," *Language Resources and Evaluation*, vol. 46, no. 4, pp. 701–719, 2012.

[48] W3C, "Simple knowledge organization system (skos)," 2009.

[49] ISO, "Iso24613:2008 – language resource management – lexical markup framework (lmf)," 2008.

[50] ISO, "Iso12620:2009 – computer applications in terminology – data categories – specification of data categories and management of a data category registry for language resources," 2009.

[51] G. Francopoulo, M. George, N. Calzolari, M. Monachini, N. Bel, M. Pet, C. Soria, *et al.*, "Lexical markup framework (lmf)," in *International Conference on Language Resources and Evaluation-LREC 2006*, 2006.

[52] C. Chiarcos, S. Nordhoff, and S. Hellmann, eds., *Linked Data in Linguistics. Representing and Connecting Language Data and Language Metadata.* Heidelberg: Springer, 2012.

[53] D. Broeder, M. Kemps-Snijders, D. V. Uytvanck, M. Windhouwer, P. Withers, P. Wittenburg, and C. Zinn, "A data category registry- and component-based metadata framework," in *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)* (N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odjik, S. Piperidis, M. Rosner, and D. Tapias, eds.), (Valletta, Malta), European Language Resources Association (ELRA), May 2010.

[54] A. Powell, M. Nilsson, A. Naeve, and P. Johnston, "DCMI Abstract Model," tech. rep., Mar. 2005.

[55] M. Gavrilidou, P. Labropoulou, E. Desipri, S. Piperidis, H. Papageorgiou, M. Monachini, F. Frontini, T. Declerck, G. Francopoulo, V. Arranz, *et al.*, "The meta-share metadata schema for the description of language resources," in *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC 2012), Istanbul, Turkey, May. European Language Resources Association (ELRA).*, 2012.

[56] E. Hinrichs, P. Banski, K. Beck, G. Budin, T. Caselli, K. Eckart, K. Elenius, G. Faaß, M. Gavrilidou, V. Henrich, V. Quochi, L. Lemnitzer, W. Maier, M. Monachini, J. Odijk, M. Ogrodniczuk, P. Osenova, P. Pajas, M. Piasecki, A. Przepiórkowski, D. V. Uytvanck, T. Schmidt, I. Schuurman, K. Simov, C. Soria, I. Skadina, J. Stepanek, P. Stranak, P. Trilsbeek, T. Trippel, and I. Vogel, "Interoperability and standards," deliverable, CLARIN, March 2011.

[57] J. Riley and D. Becker, "Seeing standards: a visualization of the metadata universe." online, 2010.

[58] S. Bird and G. Simons, "The olac metadata set and controlled vocabularies," *CoRR*, vol. cs.CL/0105030, 2001.

[59] R. Heery and M. Patel, "Application profiles: mixing and matching metadata schemas," *Ariadne*, vol. 25, pp. 27–31, 2000.

[60] G. Simons and S. Bird, "The open language archives community: An infrastructure for distributed archiving of language resources," *Literary and Linguistic Computing*, vol. 18, no. 2, pp. 117–128, 2003.

[61] A. Geyken, S. Haaf, B. Jurish, M. Schulz, J. Steinmann, C. Thomas, and F. Wiegand, "Das deutsche textarchiv: Vom historischen korpus zum aktiven archiv," *Digitale Wissenschaft. Stand und Entwicklung digital vernetzter Forschung in Deutschland, 20./21. September 2010, Köln. Beiträge der Tagung, 2., ergänzte Fassung*, pp. 157–161, 2011.

[62] Digital Library Federation, *¡METS¿ METADATA ENCODING AND TRANSMISSION STANDARD: PRIMER AND REFERENCE MANUAL*, 2010.

[63] I. S. G. on the Functional Requirements for Bibliographic Records, *FRBR: Functional Requirements for Bibliographic Records. final report.*, vol. 19 of *IFLA Series on Bibliographic Control*. Munich: K.G. Saur Verlag, 1998.

[64] A. Isaac, R. Clayphan, and B. Haslhofer, "Europeana: Moving to linked open data," *Information Standards Quarterly*, vol. 24, no. 2/3, 2012.

[65] B. Haslhofer and A. Isaac, "data. europeana. eu: The europeana linked open data pilot," in *International Conference on Dublin Core and Metadata Applications*, pp. 94–104, 2011.

[66] M. Doerr, S. Gradmann, S. Hennicke, A. Isaac, C. Meghini, and H. van de Sompel, "The europeana data model (edm)," in *World Library and Information Congress: 76th IFLA general conference and assembly*, pp. 10–15, 2010.

[67] B. Jörg, H. Uszkoreit, and A. Burt, "Lt world: Ontology and reference information portal," in *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)* (N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odjik, S. Piperidis, M. Rosner, and D. Tapias, eds.), (Valletta, Malta), European Language Resources Association (ELRA), May 2010.

[68] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: a core of semantic knowledge," in *Proceedings of the 16th international conference on World Wide Web*, pp. 697–706, ACM, 2007.

[69] T. Váradi, S. Krauwer, P. Wittenburg, M. Wynne, and K. Koskenniemi, "Clarin: Common language resources and technology infrastructure," in *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)* (N. C. C. Chair), K. Choukri, B. Maegaard, J. Mariani, J. Odjik, S. Piperidis, M. Rosner, and D. Tapias, eds.), (Marrakech, Morocco), European Language Resources Association (ELRA), may 2008. http://www.lrec-conf.org/proceedings/lrec2008/; VRADI08.317.

[70] S. Krauwer, "Clarin website mission." online, 2013.

[71] M. Durco and M. Windhouwer, "Semantic mapping in clarin component metadata," in *7 th Metadata and Semantics Research Conference (MTSR 2013)*, (Thessaloniki), 2013.

[72] M. Kemps-Snijders, M. Windhouwer, and S. E. Wright, "Putting data categories in their semantic context," in *Proceedings of the IEEE e-Humanities Workshop (e-Humanities)*, (Indianapolis, Indiana, USA), December 2008.

[73] M. Windhouwer, "Relcat and friends," in *Presentation at CLARIN-NL ISOcat workshop*, (Nijmegen), MPI for Psycholinguistics, 05 2011.

[74] I. Schuurman and M. Windhouwer., "Explicit semantics for enriched documents. what do isocat, relcat and schemacat have to offer?," in *2nd Supporting Digital Humanities conference (SDH 2011), 17-18 November 2011, Copenhagen, Denmark*, (Copenhagen, Denmark), 2011.

[75] I. Schuurman and M. Windhouwer, "Explicit semantics for enriched documents. what do isocat, relcat and schemacat have to offer," in *2nd Supporting Digital Humanities conference (SDH 2011), Copenhagen*, 2011.

[76] P. Withers, "Metadata management with arbil," in *Describing LRs with Metadata: Towards Flexibility and Interoperability in the Documentation of LR Workshop Programme*, p. 72, 2012.

[77] E. Dima, C. Hoppermann, E. W. Hinrichs, T. Trippel, and C. Zinn, "A metadata editor to support the description of linguistic resources.," in *LREC*, pp. 1061–1066, 2012.

[78] D. V. Uytvanck, C. Zinn, D. Broeder, P. Wittenburg, and M. Gardellini, "Virtual language observatory: The portal to the language resources and technology universe," in *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)* (N. Calzolari, K. Choukri, B. Maegaard, J. Mariani, J. Odjik, S. Piperidis, M. Rosner, and D. Tapias, eds.), (Valletta, Malta), European Language Resources Association (ELRA), May 2010.

[79] M. Ďurčo and L.-J. Olsson, "CMDRSB - CLARIN Metadata Repository/Service/Browser," in *Presentation at CMDI Workshop, Nijmegen*, (Nijmegen), MPI for Psycholinguistics, 01 2011.

[80] ISO, "Language codes - iso 639," 1998 - 2009.

[81] M. Windhouwer, "Mail communication on use of relation registry and vocabulary repository," 03 2013.

[82] P. Wittenburg, D. V. Uytvanck, T. Zastrow, P. Straňák, D. Broeder, F. Schiel, V. Boehlke, U. Reichel, and L. Offersgard, "Checklist for clarin b centres," tech. rep., CLARIN ERIC, 2013.

[83] H. Stehouwer, M. Durco, E. Auer, and D. Broeder, "Federated search: Towards a common search infrastructure.," in *Proceedings of LREC 2012*, (Istanbul, Turkey), pp. 3255–3259, 2012.

[84] R. Denenberg, "searchretrieve version 1.0," 04 2012.

[85] C. A. Lynch, "The z39.50 information retrieval protocol: an overview and status report," *SIGCOMM Comput. Commun. Rev.*, vol. 21, no. 1, 1991.

[86] E. Morgan, "An introduction to the Search/Retrieve URL Service (SRU)," *Ariadne*, July 2004.

[87] M. Windhouwer and S. E. Wright, "Linking to linguistic data categories in isocat," in *Linked Data in Linguistics*, pp. 99–107, Springer, 2012.

[88] A. Herold, "Mail communication on tei profile in cmd," 07 2013.

[89] M. Windhouwer, "Personal talk on expressing tei in isocat and cmd," 05 2013.

[90] P. Wittenburg, "Persistent and unique identifiers," tech. rep., CLARIN, 2009.

[91] OCLC, "Purl - persistent uniform resource locator." online, 1995.

[92] W3C, "Resource description framework (rdf)," 2004.

Appendix A

Definitions

A.1 Abbreviations

Table A.1: Acronyms used throughout this document

ACDH	Austrian Centre for Digital Humanities, cf. ??
CLARIN	Common Language Resources and Technology Infrastructure – a research infrastructure initiative, cf. 4.1
CLAVAS	Vocabulary Alignement Service for CLARIN, cf. 4.3.2
CMD	Component Metadata Framework – the data model underlying the CMD Infrastructure, cf. 3.1
CMDI	Component Metadata Infrastructure, cf. 4.2
ERIC	European Research Infrastructure Consortium – a legal entity for long-term research infrastructure initiatives
DARIAH	Digital Research Infrastructure for Arts and Humanities[1] – another research infrastructure initiative, sister project to CLARIN
DC	data category, cf. 4.2.1
DCR	data category registry, cf. 4.2.1 [1]
DH	Digital Humanities, also eHumanities
LINDAT	Czech national infrastructure for LRT[2]
MPI	Max Planck Institute, especially MPI for Psycholinguistics in Nijmegen, task leader of CMDI
OLAC	Open Language Archive Community[3] 3.2.2
PID	persistent identifier [90]
PURL	persistent uniform resource locator [91]
RDF	Resource Description Framework [92]
RR	Relation Registry, cf. 4.2.1
TEI	Text Encoding Initiative, cf. ??

A.2 Namespaces

A.3 Formatting conventions

Inline formatting for highlighting:

Table A.2: Namespaces referenced in this document

Prefix name	*Prefix IRI*
rdf:	http://www.w3.org/1999/02/22-rdf-syntax-ns#
rdfs:	http://www.w3.org/2000/01/rdf-schema#
xsd:	http://www.w3.org/2001/XMLSchema#
owl:	http://www.w3.org/2002/07/owl#
skos:	http://www.w3.org/2004/02/skos/core#
isocat:	http://www.isocat.org/datcat/
dcr:	http://isocat.org/ns/dcr.rdf#
cmd:	http://clarin.eu/cmd/1.0#
cmds:	https://infra.clarin.eu/cmd/general-component-schema.xsd
dce:	http://purl.org/dc/elements/1.1/
dcterms:	http://purl.org/dc/terms
oa:	http://www.w3.org/ns/oa#
olac:	http://www.language-archives.org/OLAC/1.1/
ore:	http://www.openarchives.org/ore/terms/
cr:	http://catalog.clarin.eu/ds/ComponentRegistry/rest/registry/

Named Entity	an application or project name (institution names are written in plain text)
`code`	names of xml elements and attributes; also a concrete (sample) value
`concept`	lexical label denoting a concept
variable	definitions and variables

Definition A.1: A definition in a block with caption

$$some\ formal\ expression\ equation\ or\ grammar$$

Example blocks, simple:

```
  Short piece of sample data
```

or with tabs (especially for RDF triples):

```
my:work                 my:example              my:block
```

Appendix B

Data model reference

In the following complete data models, schemas are listed for reference: The diagram of the data model for data category specification in figure B.1, Terms.xsd – the XML schema used by the SMC module internally in listing B.1 (cf. 5.2.2) and the general-component-schema.xsd[1] – the schema representing the CMD meta model for defining CMD profiles and components in listing B.2. Figure B.2 depicts an abstract reference architecture, that provides a conceptual frame for this work and in figure B.3 an overview of the roles and services of the ACDH – Austrian Centre for Digital Humanities – the home of SMC – explicates the concrete current situation regarding the architectural context of SMC.

Listing B.1: Terms.xsd – schema of the internal data model 5.2.2

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified" xmlns:ns2="http://www.w3.
  <xs:import namespace="http://www.w3.org/1999/xlink" schemaLocation="ns2.xsd"/>
  <xs:import namespace="http://www.w3.org/XML/1998/namespace" schemaLocation="xml.xsd"/>
  <xs:element name="Termsets">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" ref="Termset"/>
      </xs:sequence>
      <xs:attribute name="count" type="xs:integer"/>
      <xs:attribute name="type" type="xs:NCName"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="Termset">
    <xs:complexType>
      <xs:sequence>
        <xs:element minOccurs="0" ref="info"/>
        <xs:choice>
          <xs:element minOccurs="0" maxOccurs="unbounded" ref="Concept"/>
          <xs:element minOccurs="0" maxOccurs="unbounded" ref="Term"/>
        </xs:choice>
      </xs:sequence>
      <xs:attribute name="id" type="xs:anyURI"/>
      <xs:attribute name="name"/>
      <xs:attribute name="set" type="xs:NCName"/>
      <xs:attribute name="type" type="xs:NCName"/>
      <xs:attribute name="url" type="xs:anyURI"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="Concept">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" ref="Term"/>
```

[1] https://infra.clarin.eu/cmd/general-component-schema.xsd

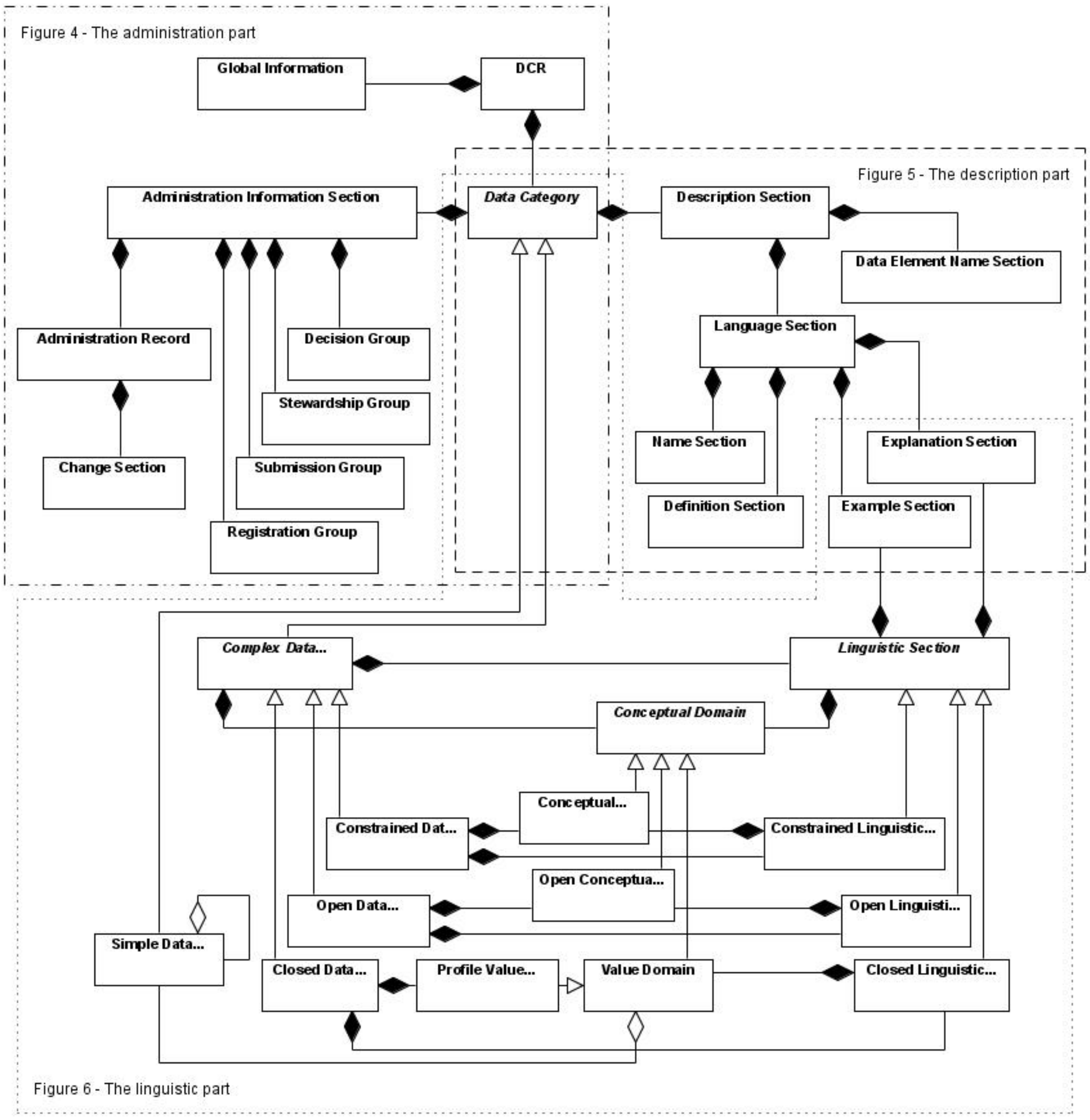


Figure B.1: DCIF – the data model for the Data Category Registry as defined by the ISO Standard ISO12620:2009 [1]

```
        <xs:element minOccurs="0" maxOccurs="unbounded" ref="info"/>
      </xs:sequence>
      <xs:attribute name="datcat-type" type="xs:NCName"/>
      <xs:attribute name="id" use="required" type="xs:anyURI"/>
      <xs:attribute name="type" use="required" type="xs:NCName"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="info">
    <xs:complexType mixed="true">
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element ref="commentsCount"/>
        <xs:element ref="creatorName"/>
        <xs:element ref="description"/>
        <xs:element ref="domainName"/>
        <xs:element ref="groupName"/>
        <xs:element ref="id"/>
        <xs:element ref="name"/>
        <xs:element ref="registrationDate"/>
        <xs:element ref="showInEditor"/>
        <xs:element ref="userId"/>
        <xs:element ref="ns2:href"/>
      </xs:choice>
      <xs:attribute ref="xml:lang"/>
    </xs:complexType>
  </xs:element>
  <xs:element name="commentsCount" type="xs:integer"/>
  <xs:element name="creatorName" type="xs:string"/>
  <xs:element name="description" type="xs:string"/>
  <xs:element name="domainName" type="xs:string"/>
  <xs:element name="groupName" type="xs:string"/>
  <xs:element name="id" type="xs:anyURI"/>
  <xs:element name="name" type="xs:NCName"/>
  <xs:element name="registrationDate" type="xs:dateTime"/>
  <xs:element name="showInEditor" type="xs:boolean"/>
  <xs:element name="userId" type="xs:integer"/>
  <xs:element name="Term">
    <xs:complexType mixed="true">
      <xs:attribute name="datcat" type="xs:anyURI"/>
      <xs:attribute name="elem" type="xs:NCName"/>
      <xs:attribute name="id" type="xs:anyURI"/>
      <xs:attribute name="name" type="xs:NCName"/>
      <xs:attribute name="parent"/>
      <xs:attribute name="path" type="xs:NCName"/>
      <xs:attribute name="schema" type="xs:NMTOKEN"/>
      <xs:attribute name="set" type="xs:NCName"/>
      <xs:attribute name="type" use="required" type="xs:NCName"/>
      <xs:attribute ref="xml:lang"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Listing B.2: general-component-schema.xsd – schema of the CMD meta model for defining CMD profiles and components

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
    $Revision: 2517 $
    $Date: 2013-01-30 16:29:31 +0100 (Wed, 30 Jan 2013) $
-->

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:import namespace="http://www.w3.org/XML/1998/namespace"
        schemaLocation="http://www.w3.org/2005/08/xml.xsd"/>

    <!-- root element -->
    <xs:element name="CMD_ComponentSpec">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Header">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="ID" type="xs:anyURI" minOccurs="0"/>
                            <xs:element name="Name" type="xs:string" minOccurs="0"/>
                            <xs:element name="Description" type="xs:string" minOccurs="0"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
                <xs:element name="CMD_Component" type="CMD_Component_type" maxOccurs="unbounded">
                    <xs:annotation>
                    <xs:documentation>At the root level there should always be a
                    Component.</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
        <xs:attribute name="isProfile" type="xs:boolean" use="required"/>
    </xs:complexType>
</xs:element>

<!-- recursive construction: A component can contain elements and/or other components
-->
<xs:group name="group">
    <xs:sequence>
        <!-- from small (attribute) to big (component) -->
        <xs:element name="AttributeList" type="AttributeList_type" minOccurs="0" maxOccurs="1"/>
        <xs:element name="CMD_Element" type="CMD_Element_type" minOccurs="0"
            maxOccurs="unbounded"> </xs:element>
        <xs:element name="CMD_Component" type="CMD_Component_type" minOccurs="0"
            maxOccurs="unbounded"/>
    </xs:sequence>
</xs:group>

<!-- type definitions -->
<xs:complexType name="CMD_Element_type">
    <xs:sequence>
        <xs:element name="AttributeList" type="AttributeList_type" minOccurs="0" maxOccurs="1">
            <xs:annotation>
                <xs:documentation>The AttributeList child of an element contains a set of XML
                    attributes for that element.</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element minOccurs="0" maxOccurs="1" name="ValueScheme" type="ValueScheme_type">
            <xs:annotation>
                <xs:documentation>When an element is linked to a regular expression or a
                    controlled vocabulary, the ValueScheme sub-element contains more information
                    about this.</xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:sequence>
    <xs:attributeGroup ref="clarin_element_attributes"/>
</xs:complexType>


<xs:complexType name="ValueScheme_type">
    <xs:choice>
        <xs:element name="pattern" type="xs:string" maxOccurs="1">
            <xs:annotation>
                <xs:documentation>Specification of a regular expression the element should
                    comply with.</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="enumeration" type="enumeration_type">
            <xs:annotation>
                <xs:documentation>A list of the allowed values of a controlled
                vocabulary.</xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:choice>
</xs:complexType>


<xs:complexType name="AttributeList_type">
    <xs:sequence>
```

```
        <xs:element name="Attribute" minOccurs="1" maxOccurs="unbounded">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="Name" type="xs:string">
                        <xs:annotation>
                            <xs:documentation>The name of the attribute.</xs:documentation>
                        </xs:annotation>
                    </xs:element>
                    <xs:element name="ConceptLink" type="xs:anyURI" minOccurs="0">
                        <xs:annotation>
                            <xs:documentation>A link to the ISOcat data category registry (or any other concept
                        </xs:annotation>
                    </xs:element>
                    <xs:choice>
                        <xs:element name="Type" type="allowed_attributetypes_type">
                            <xs:annotation>
                                <xs:documentation>For the use of simple XML types as the type of
                                    the attribute.</xs:documentation>
                            </xs:annotation>
                        </xs:element>
                        <xs:element name="ValueScheme" type="ValueScheme_type">
                            <xs:annotation>
                                <xs:documentation>For the use of a regular expression or a
                                    controlled vocabulary as the type of the
                                attribute.</xs:documentation>
                            </xs:annotation>
                        </xs:element>
                    </xs:choice>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
</xs:complexType>

<xs:complexType name="CMD_Component_type">
    <xs:group ref="group" minOccurs="0"/>
    <xs:attributeGroup ref="clarin_component_attributes"/>
</xs:complexType>


<!-- list of all attributes that can be bound to a cl_el -->
<xs:attributeGroup name="clarin_element_attributes">
    <xs:attribute name="name" type="xs:Name" use="required">
        <xs:annotation>
            <xs:documentation>The name of the element.</xs:documentation>
        </xs:annotation>
    </xs:attribute>
    <xs:attribute name="ConceptLink" type="xs:anyURI">
        <xs:annotation>
            <xs:documentation>A link to the ISOcat data category registry (or any other concept
                registry).</xs:documentation>
        </xs:annotation>
    </xs:attribute>
    <xs:attribute name="ValueScheme" type="allowed_attributetypes_type">
        <xs:annotation>
            <xs:documentation>Used to specify that an element has a simple XML type (string,
                integer, etc)</xs:documentation>
        </xs:annotation>
    </xs:attribute>
    <xs:attribute name="CardinalityMin" type="cardinality_type">
        <xs:annotation>
            <xs:documentation>Minimal number of occurrences.</xs:documentation>
        </xs:annotation>
    </xs:attribute>
    <xs:attribute name="CardinalityMax" type="cardinality_type">
        <xs:annotation>
            <xs:documentation>Maximal number of occurrences.</xs:documentation>
        </xs:annotation>
    </xs:attribute>
    <xs:attribute name="Documentation" type="xs:string">
        <xs:annotation>
            <xs:documentation>Some information an application (eg Arbil) can display to give
                guidance to the user when entering metadata.</xs:documentation>
        </xs:annotation>
    </xs:attribute>
    <xs:attribute name="DisplayPriority" type="xs:integer">
        <xs:annotation>
            <xs:documentation>The element with the highest priority will be displayed as the
                label for a metadata file (eg in Arbil)</xs:documentation>
        </xs:annotation>
    </xs:attribute>
    <xs:attribute name="Multilingual" type="xs:boolean">
        <xs:annotation>
            <xs:documentation>Indicates that this element can have values in multiple languages
                (and thus is repeatable). This will result in the possibility of using the
                xml:lang attribute in the metadata instances that are
            created.</xs:documentation>
        </xs:annotation>
    </xs:attribute>
</xs:attributeGroup>

<!-- list of all attributes that can be bound to a cl_comp -->
<xs:attributeGroup name="clarin_component_attributes">
    <xs:attribute name="name" type="xs:Name"/>
    <xs:attribute name="ComponentId" type="xs:anyURI">
        <xs:annotation>
            <xs:documentation>Indicates that a component (using its unique ComponentId issued by
                the ComponentRegistry) should be included.</xs:documentation>
        </xs:annotation>
    </xs:attribute>
    <xs:attribute name="ConceptLink" type="xs:anyURI">
        <xs:annotation>
            <xs:documentation>A link to the ISOcat data category registry (or any other concept
                registry). Currently not used.</xs:documentation>
        </xs:annotation>
    </xs:attribute>
    <xs:attribute name="filename" type="xs:anyURI">
        <xs:annotation>
            <xs:documentation>Outdated way of including an external component. Here for backward
                compatibility with the XML-cmdi-toolkit.</xs:documentation>
        </xs:annotation>
    </xs:attribute>

    <!-- (components cannot have a ValueScheme attribute) -->

    <xs:attribute name="CardinalityMin" type="cardinality_type"/>
    <xs:attribute name="CardinalityMax" type="cardinality_type"/>
    <xs:attribute ref="xml:base"/>
</xs:attributeGroup>


<xs:simpleType name="cardinality_type">
    <xs:annotation>
        <xs:documentation>cardinality for elements and components</xs:documentation>
    </xs:annotation>
    <xs:union>
```

```xml
            <xs:simpleType>
                <xs:list itemType="xs:nonNegativeInteger"/>
            </xs:simpleType>
            <xs:simpleType>
                <xs:restriction base="xs:string">
                    <xs:enumeration value="unbounded"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:union>
    </xs:simpleType>

    <xs:simpleType name="allowed_attributetypes_type">
        <xs:annotation>
            <xs:documentation>Subset of XSD types that are allowed as CMD type</xs:documentation>
        </xs:annotation>

        <xs:restriction base="xs:token">
            <xs:enumeration value="boolean"/>
            <xs:enumeration value="decimal"/>
            <xs:enumeration value="float"/>
            <xs:enumeration value="int"/>
            <xs:enumeration value="string"/>
            <xs:enumeration value="anyURI"/>
            <xs:enumeration value="date"/>
            <xs:enumeration value="gDay"/>
            <xs:enumeration value="gMonth"/>
            <xs:enumeration value="gYear"/>
            <xs:enumeration value="time"/>
            <xs:enumeration value="dateTime"/>
        </xs:restriction>
    </xs:simpleType>

    <xs:complexType name="enumeration_type">
        <xs:annotation>
            <xs:documentation>controlled vocabularies</xs:documentation>
        </xs:annotation>
        <xs:choice minOccurs="0" maxOccurs="unbounded">
            <xs:element name="item" type="item_type">
                <xs:annotation>
                    <xs:documentation>An item from a controlled vocabulary.</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:element name="appinfo" type="xs:string">
                <xs:annotation>
                    <xs:documentation>End-user guidance about the value of the controlled vocabulary
                        as a whole. Currently not used.</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:choice>
    </xs:complexType>

    <xs:complexType name="item_type">
        <xs:simpleContent>
            <xs:extension base="xs:string">
                <xs:attribute type="xs:anyURI" name="ConceptLink">
                    <xs:annotation>
                        <xs:documentation>A link to the ISOcat data category registry (or any other
                            concept registry) related to this controllec vocabulary
                        item.</xs:documentation>
                    </xs:annotation>
                </xs:attribute>
                <xs:attribute type="xs:string" name="AppInfo">
                    <xs:annotation>
```

```
                        <xs:documentation>End-user guidance about the value of this controlled
                            vocabulary item.</xs:documentation>
                    </xs:annotation>
                </xs:attribute>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>

</xs:schema>
```

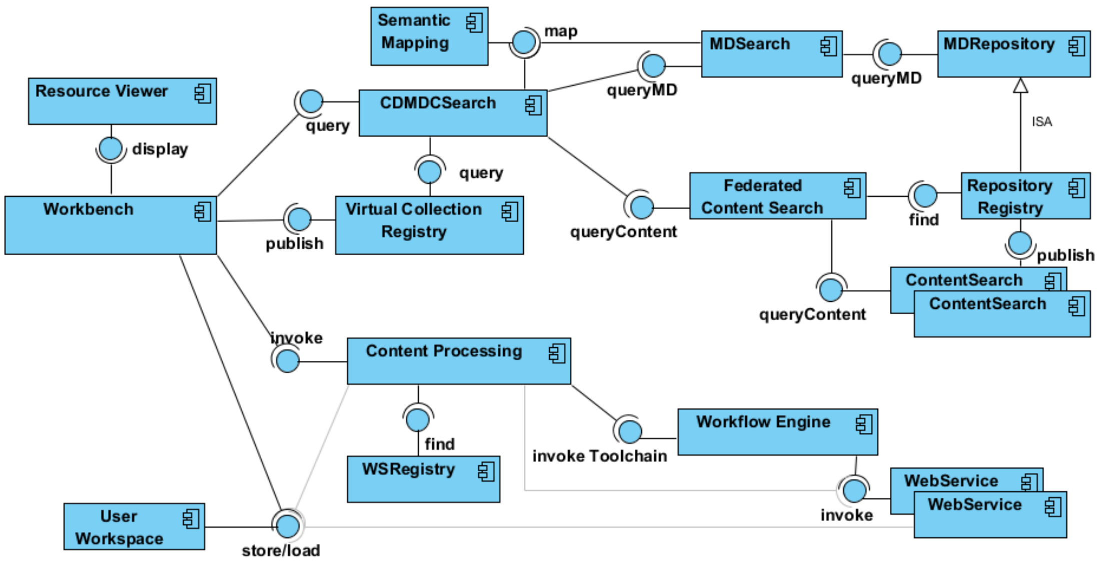
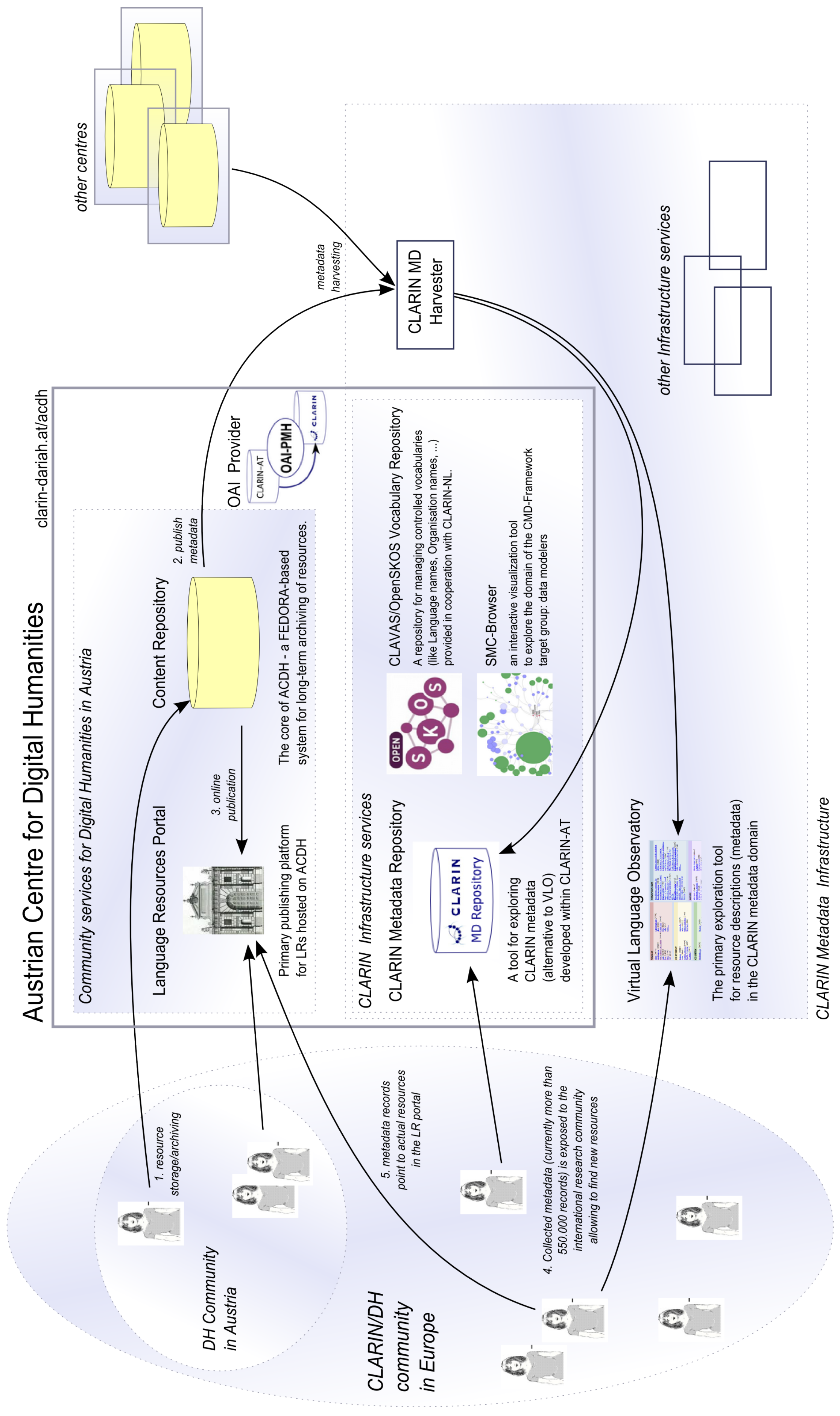


Figure B.2: Reference Architecture



Austrian Centre for Digital Humanities
clarin-dariah.at/acdh
Community services for Digital Humanities in Austria
Language Resources Portal
Primary publishing platform for LRs hosted on ACDH
3. online publication
Content Repository
The core of ACDH - a FEDORA-based system for long-term archiving of resources.
2. publish metadata
OAI Provider
CLARIN-AT
OAI-PMH
CLARIN
CLARIN Infrastructure services
CLARIN Metadata Repository
CLARIN MD Repository
A tool for exploring CLARIN metadata (alternative to VLO) developed within CLARIN-AT
CLAVAS/OpenSKOS Vocabulary Repository
A repository for managing controlled vocabularies (like Language names, Organisation names, ...) provided in cooperation with CLARIN-NL.
SMC-Browser
an interactive visualization tool to explore the domain of the CMD-Framework target group: data modelers
other centres
metadata harvesting
CLARIN MD Harvester
other Infrastructure services
Virtual Language Observatory
The primary exploration tool for resource descriptions (metadata) in the CLARIN metadata domain
CLARIN Metadata Infrastructure
1. resource storage/archiving
DH Community in Austria
CLARIN/DH community in Europe
5. metadata records point to actual resources in the LR portal
4. Collected metadata (currently more than 550.000 records) is exposed to the international research community allowing to find new resources

Figure B.3: Austrian Centre for Digital Humanities - the home of SMC - in context

Appendix C

CMD – sample data

C.1 Definition of a CMD profile

Following listing presents a sample CMD specification for the collection#clarin.eu:cr1:p_1345561703620 profile.

Listing C.1: Specification of the collection profile

```
<?xml version="1.0" encoding="UTF-8"?>
<CMD_ComponentSpec xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" isProfile="true"
                xsi:schemaLocation="http://www.clarin.eu/cmd https://infra.clarin.eu/cmd/general-component-sch
    <Header>
        <ID>clarin.eu:cr1:p_1345561703620</ID>
        <Name>collection</Name>
        <Description>standard profile for the description of collections</Description>
    </Header>
    <CMD_Component CardinalityMax="1" CardinalityMin="1" name="collection">
        <CMD_Component CardinalityMax="1" CardinalityMin="1"
                    ComponentId="clarin.eu:cr1:c_1345561703619"
                    name="CollectionInfo">
            <CMD_Element Multilingual="true" DisplayPriority="2" Documentation="Collection name"
                    CardinalityMax="unbounded"
                    CardinalityMin="1"
                    ValueScheme="string"
                    ConceptLink="http://www.isocat.org/datcat/DC-2544"
                    name="Name"/>
            <CMD_Element Multilingual="true" DisplayPriority="1"
                    Documentation="Allows a more elaborate description than Name"
                    CardinalityMax="unbounded"
                    CardinalityMin="0"
                    ValueScheme="string"
                    ConceptLink="http://www.isocat.org/datcat/DC-2545"
                    name="Title"/>
            <CMD_Element Multilingual="true"
                    Documentation="Person or organisation that owns the collection"
                    CardinalityMax="unbounded"
                    CardinalityMin="0"
                    ValueScheme="string"
                    ConceptLink="http://www.isocat.org/datcat/DC-2956"
                    name="Owner"/>
            <CMD_Component CardinalityMax="unbounded" CardinalityMin="0"
                    ComponentId="clarin.eu:cr1:c_1271859438110"
                    name="ISO639">
                <CMD_Element DisplayPriority="1" CardinalityMax="1" CardinalityMin="1"
                    ConceptLink="http://www.isocat.org/datcat/DC-2482"
                    name="iso-639-3-code">
                    <ValueScheme>
                        <enumeration>
                <item AppInfo="Ghotuo (aaa)" ConceptLink="http://cdb.iso.org/lg/CDB-00132443-0
                <item AppInfo="Alumu-Tesu (aab)" ConceptLink="http://cdb.iso.org/lg/CDB-0013377
                ... [7.674 items]
                <item AppInfo="Zyphe (zyp)" ConceptLink="http://cdb.iso.org/lg/CDB-00136139-001
                <item AppInfo="Zaza; Dimili; Dimli; Kirdki; Kirmanjki; Zazaki (zza)"
               ConceptLink="http://cdb.iso.org/lg/CDB-00131000-001">zza</item>
                <item AppInfo="Zuojiang Zhuang (zzj)"
               ConceptLink="http://cdb.iso.org/lg/CDB-00136140-001">zzj</item>
            </enumeration>
        </ValueScheme>
    </CMD_Element>
</CMD_Component>
<CMD_Component CardinalityMax="1" CardinalityMin="0"
            ComponentId="clarin.eu:cr1:c_1271859438127"
            name="Modality">
    <CMD_Element CardinalityMax="unbounded" CardinalityMin="1"
             ConceptLink="http://www.isocat.org/datcat/DC-2490"
             name="Modality">
        <ValueScheme>
            <enumeration>
                <item ConceptLink="http://www.isocat.org/datcat/DC-2591">Unknown</item>
                <item ConceptLink="http://www.isocat.org/datcat/DC-2592">Unspecified</item>
                <item>Spoken</item>
                <item>Written</item>
                <item>Music notation</item>
                <item ConceptLink="http://www.isocat.org/datcat/DC-2994">Gestures</item>
                <item>Pointing-gestures</item>
                <item>Signs</item>
                <item ConceptLink="http://www.isocat.org/datcat/DC-2977">Eye-gaze</item>
                <item>Facial-expressions</item>
                <item>Emotional-state</item>
                <item>Haptic</item>
                <item>Song</item>
                <item ConceptLink="http://www.isocat.org/datcat/DC-2611">Instrumental music</it
                <item ConceptLink="http://www.isocat.org/datcat/DC-2599">Other</item>
            </enumeration>
        </ValueScheme>
    </CMD_Element>
    <CMD_Component CardinalityMax="1" CardinalityMin="0"
               ConceptLink="http://www.isocat.org/datcat/DC-2520"
               ComponentId="clarin.eu:cr1:c_1271859438118"
               name="Description">
        <CMD_Element Multilingual="true" CardinalityMax="1" CardinalityMin="1" ValueScheme="st
                ConceptLink="http://www.isocat.org/datcat/DC-2520"
                name="Description">
            <AttributeList>
                <Attribute>
                    <Name>LanguageID</Name>
                    <Type>string</Type>
                </Attribute>
            </AttributeList>
        </CMD_Element>
    </CMD_Component>
</CMD_Component>
<CMD_Component CardinalityMax="1" CardinalityMin="0"
            ConceptLink="http://www.isocat.org/datcat/DC-2502"
            name="TimeCoverage">
    <CMD_Element DisplayPriority="1"
             Documentation="The start of the timespan that the resource is about (yyyy)"
             CardinalityMax="1"
             CardinalityMin="0"
             ValueScheme="gYear"
             name="StartYear"/>
        <CMD_Element Documentation="The end of the timespan that the resource is about (yyyy)"
                CardinalityMax="1"
                CardinalityMin="0"
                ValueScheme="gYear"
                name="EndYear"/>
      </CMD_Component>
      <CMD_Component CardinalityMax="1" CardinalityMin="0"
                ConceptLink="http://www.isocat.org/datcat/DC-2520"
                ComponentId="clarin.eu:cr1:c_1271859438118"
                name="Description">
        <CMD_Element Multilingual="true" CardinalityMax="1" CardinalityMin="1" ValueScheme="string"
                ConceptLink="http://www.isocat.org/datcat/DC-2520"
                name="Description">
          <AttributeList>
            <Attribute>
              <Name>LanguageID</Name>
              <Type>string</Type>
            </Attribute>
          </AttributeList>
        </CMD_Element>
      </CMD_Component>
    </CMD_Component>
    <CMD_Component CardinalityMax="1" CardinalityMin="1"
              ComponentId="clarin.eu:cr1:c_1345561703649"
              name="License">
      <CMD_Element DisplayPriority="4" CardinalityMax="1" CardinalityMin="1"
              ConceptLink="http://www.isocat.org/datcat/DC-5439"
              name="DistributionType">
        <ValueScheme>
          <enumeration>
            <item AppInfo="openly available" ConceptLink="http://www.isocat.org/datcat/DC-2621">public
            <item AppInfo="available for academic use"
            ConceptLink="http://www.isocat.org/datcat/DC-5438">academic</item>
            <item AppInfo="only available for license owners"
            ConceptLink="http://www.isocat.org/datcat/DC-5306">restricted</item>
            <item AppInfo="no availability information specified"
            ConceptLink="http://www.isocat.org/datcat/DC-2592">unspecified</item>
          </enumeration>
        </ValueScheme>
      </CMD_Element>
      <CMD_Element Multilingual="false" DisplayPriority="3"
              Documentation="Name of the license. Eg: GPL, CC-BY-SA, BSD, ELRA_END_USER"
              CardinalityMax="1"
              CardinalityMin="1"
              ValueScheme="string"
              ConceptLink="http://www.isocat.org/datcat/DC-2457"
              name="LicenseName"/>
      <CMD_Element Documentation="URL where the license can be retrieved, eg http://creativecommons.org/lic
              CardinalityMax="1"
              CardinalityMin="1"
              ValueScheme="anyURI"
              name="LicenseURL"/>
      <CMD_Element Documentation="Use of this resource is not allowed for commercial purposes."
              CardinalityMax="1"
              CardinalityMin="0"
              ValueScheme="boolean"
              name="NonCommercialUsageOnly"/>
      <CMD_Element Documentation="The user needs to report usage to the resource producer."
              CardinalityMax="1"
              CardinalityMin="0"
              ValueScheme="boolean"
              name="UsageReportRequired"/>
      <CMD_Element Documentation="If the resource is changed, it should be made available again in the alte
```

```
                      CardinalityMax="1"
                      CardinalityMin="0"
                      ValueScheme="boolean"
                      name="ModificationsRequireRedeposition"/>
        </CMD_Component>
        <CMD_Component CardinalityMax="1" CardinalityMin="0"
                     ComponentId="clarin.eu:cr1:c_1271859438113"
                     name="Contact">
            <CMD_Element CardinalityMax="unbounded" CardinalityMin="0" ValueScheme="string"
                      ConceptLink="http://www.isocat.org/datcat/DC-2978"
                      name="Person"/>
            <CMD_Element CardinalityMax="unbounded" CardinalityMin="0" ValueScheme="string"
                      ConceptLink="http://www.isocat.org/datcat/DC-2505"
                      name="Address"/>
            <CMD_Element CardinalityMax="unbounded" CardinalityMin="0" ValueScheme="string"
                      ConceptLink="http://www.isocat.org/datcat/DC-2521"
                      name="Email"/>
            <CMD_Element Multilingual="true" CardinalityMax="unbounded" CardinalityMin="0"
                      ValueScheme="string"
                      ConceptLink="http://www.isocat.org/datcat/DC-2979"
                      name="Organisation"/>
            <CMD_Element CardinalityMax="unbounded" CardinalityMin="0" ValueScheme="string"
                      ConceptLink="http://www.isocat.org/datcat/DC-2461"
                      name="Telephone"/>
            <CMD_Element CardinalityMax="unbounded" CardinalityMin="0" ValueScheme="anyURI"
                      ConceptLink="http://www.isocat.org/datcat/DC-63"
                      name="Website"/>
        </CMD_Component>
        <CMD_Component CardinalityMax="unbounded" CardinalityMin="0"
                     ComponentId="clarin.eu:cr1:c_1316422391221"
                     name="WebReference">
            <CMD_Element DisplayPriority="2" Documentation="A URL (e.g. http://www.clarin.eu)"
                      CardinalityMax="1"
                      CardinalityMin="1"
                      ValueScheme="anyURI"
                      ConceptLink="http://www.isocat.org/datcat/DC-2546"
                      name="Website"/>
            <CMD_Element Multilingual="true" DisplayPriority="1"
                      Documentation="A description: what is this website about (e.g. Documentation about a
                      CardinalityMax="1"
                      CardinalityMin="1"
                      ValueScheme="string"
                      ConceptLink="http://www.isocat.org/datcat/DC-2520"
                      name="Description"/>
        </CMD_Component>
    </CMD_Component>
</CMD_ComponentSpec>
```

C.2 CMD record

Following listing represents a sample CMD record - an instance of the collection profile listed above.

add a sample collection record

[1]

[1]http://clarin.arz.oeaw.ac.at/exist/apps/cr-xq/mdrepo/fcs?operation=searchRetrieve&query=cmd.profile%3D%22clarin.eu:cr1:p_1345561703620%22&x-context=&x-format=html

Appendix D

SMC – documentation

D.1 Documentation of smc-xsl

generate and reference XSLT-documentation

D.2 SMC Browser user documentation

Explore the *Component Metadata Framework*

In *CMD*, metadata schemas are defined by profiles, that are constructed out of reusable components - collections of metadata fields. The components can contain other components, and they can be reused in multiple profiles. Furthermore, every CMD element (metadata field) refers via a PID to a data category to indicate unambiguously how the content of the field in a metadata description should be interpreted (Broeder et al., 2010).

Thus, every profile can be expressed as a tree, with the profile component as the root node, the used components as intermediate nodes and elements or data categories as leaf nodes, parent-child relationship being defined by the inclusion (`componentA -includes-> componentB`) or referencing (`elementA -refersTo-> datcat1`).The reuse of components in multiple profiles and especially also the referencing of the same data categories in multiple CMD elements leads to a blending of the individual profile trees into a graph (acyclic directed, but not necessarily connected).

SMC Browser visualizes this graph structure in an interactive fashion. You can have a look at the examples for inspiration.

It is implemented on top of wonderful js-library d3, the code checked in clarin-svn (and needs refactoring). More technical documentation follows soon.

D.2.1 Data

The graph is constructed from all profiles defined in the Component Registry. To resolve name and description of data categories referenced in the CMD elements definitions of all (public) data categories from DublinCore and ISOcat (from the Metadata Profile [RDF] - retrieving takes some time!) are fetched. However only data categories used in CMD will get part of the graph. Here is a quantitative summary of the dataset.

When inspecting the numbers, it is important to be aware of the occurrence expansion resulting from the reusability of the components. So in an example, a component C has 2 subcomponents and is reused within one profile by two other components A and B, the resulting profile will consist of (at least) 8 components (`[A, B, A/C, B/C, A/C/C1, A/C/C2, B/C/C1, B/C/C2]`), although only 5 distinct components are used. The same goes for elements in reused components. In most cases it is indicated in the label, if the number reflect distinct items, or all (expanded) occurrences.

(Some of the) numbers in the statistics lead to a list of corresponding terms. E.g. in the summary for a profile, clicking on the components-number lists all the components of given profile alphabetically. Currently there are such lists for:

- `profile -> components`
- `profile -> elements`
- `profile -> data categories`
- `data category -> profiles`

D.2.2 User Interface

The User interface is divided into 4 main parts:

Index Lists all available Profiles, Components, Elements and used Data Categories The lists can be filtered (enter search pattern in the input box at the top of the index-pane) By clicking on individual items, they are added to the *selected nodes* and get rendered in the graph pane

Main (Graph) Pane for rendering the graph.

Navigation This is the control panel governing the rendering of the graph. See below for available Options.

Detail In this pane, overall summary of the data is displayed by default, but mainly the detail information about the selected nodes is listed here.

D.2.3 Interaction

Following data sets are distinguished wrt user interaction:

all data the full graph with all profiles, components, elements and data categories and links between them.

Currently this amounts to roughly 2.000 nodes and 3.000 links

selected nodes nodes explicitely selected by the user (see below how to select nodes)

data to show the subset of data that shall be displayed.

Starting from the selected nodes, connected nodes (and connecting edges) are determined based on the options (`depth-before`, `depth-after`).

The nodes are colour-coded by type:
There are multiple ways to select/unselect nodes:

select from index by clicking individual items in the index list, the item will be **added** to the selected nodes

clicking on an already selected item unselects it

select in graph by clicking on a visible node in the graph, the node will be **added** to the selected nodes

clicking on an already selected node unselects it

select area in graph by dragging (hold mouse button down and pull) a rectangle in the graph pane, all nodes within that rectangle get selected all other nodes will be unselected

unselect in detail pane clicking on an item in the detail pane unselects it

select in statistics as mentioned in Data (some) numbers in the statistics reveal a list of corresponding terms. Clicking on these terms in the statistics page leads to the browser, with given term as selected node (and default settings)

select in statistics in the detail pane the numbers from statistics page are shown also in the detail pane for selected nodes. Here, clicking on a term from these lists adds it to the graph, as a selected node.

mouseover on mouse over a node, all connected nodes to given node (and connecting links) within the visible sub-graph are highlighted and all other nodes and links are faded

drag a node click and hold on a node, one can move the node around, however usually the layout is stronger and puts the node back to its original position. Not so with the freeze-layout, that freezes all the nodes and lets you move them around freely

D.2.4 Options

The navigation pane provides following option to control the rendering of the graph:

depth-before how many levels of connected ancestor nodes shall be displayed

depth-after how many levels of connected descendant nodes shall be displayed

link-distance approximate distance between individual nodes (not exact, because it is just one of multiple factor for the layouting of the graph)

charge the higher the charge, the more the nodes tend to drift apart

friction factor for "cooling down" the layout, lower numbers (50-70) stabilize the graph more quickly, but it may be too early, with higher numbers (95-100) the layout has more time/freedom to arrange, but may get jittery

node-size N = all nodes have given diameter N;

usage = node is scaled based on how often the node appears in the complete dataset i.e. often reused elements (like description or language) will be bigger

labels show/hide all labels hiding the labels accelerates the rendering significantly, which may be an issue if more nodes are displayed. irrespective of this option, on mouseover labels for all and only the highlighted nodes are displayed

curve straight or arc (better visibility)

layout There are a few layouting algorithms provided. They are all not optimal in any way, but most of the time, they deliver quite good results. For different data displayed other algorithm may be more appropriate:

force undirected layout, trying to spread the nodes in the pane optimally, equally in all directions This is the underlying layouting algorithm. All the other layouts build on top of it, by just adding further constraints.

vertical-tree top-down layout respect the direction of the edges, children are always below the parents

horizontal-tree left-right layout respect the direction of the edges, children are always right to the parents (at least they should be, currently, in certain configurations, the layout does not get the orientation for some links right)

weak-tree a layout that "tends" towards left to right arrangement, but not strictly so (experimental)

dot strict left to right reusing the x-positioning as determined by dot Arranges the nodes in strict ranks (typical for dot layout) This is done in a separate preprocessing step for the whole graph, so the positioning may be suboptimal for a given subgraph. The y-coordinate is approximated on the fly by the base algorithm.

freeze this is actually a "no-layout" - the nodes just stay fixed in their last position, However, individual nodes still can be dragged around, so this can be used to adjust a few nodes for better legibility (or aesthetics), but only when you start moving around inividual nodes, you will learn to appreciate the great (and tedious) work of the layouting algorithms, so generally you want to try to play around with the other settings to achieve a satisfying result.

D.2.5 Linking, Export

The navigation pane exposes a **link**, that captures the exact current state of the interface (just the options and the selection, not the positioning of the elements), so that it can be bookmarked, emailed etc.

Furthermore, there is the **download**, that allows to export the current graph as SVG. This is accomplished without a round trip to the server, with a javascript trick serializing the svg as base64-data into the url (so you don't want to save (or see) the exported url). But you can both, right click the link and [Save link as...], or click on the link, which opens the SVG in a new tab where you can view, resize, print and save it. Employing this simple method also means, that there is no possibility to export the graph in PNG, PDF or any other format, because this would require server-side processing. (However this is a planned future enhancement.)

D.2.6 Issues

Performance Chrome is by far the fastest, followed by IE(9). A serious performance degradation was observed for graphs above 200 nodes on Firefox. Showing labels also significantly affects performance.

Bounds When the graph gets to big, it does not fit in the viewing pane. This will be tackled soon (either scrollbars or applying boundaries). Meanwhile, you can reduce the link-distance and charge parameters or change the layout.

D.2.7 Plans and ToDos

Substantial issues:

- Add information from **RelationRegistry** (relations between DatCats)
- Blend in instance data from **MDRepository** (allow search on MDRepository)
- graph operations (intersect, difference of subrgraphs)

Smaller enhancements of the user interface:

- select nodes by querying the names (e.g. show me all nodes with “Access” in their name)
- option to show only selected types of nodes (e.g. only profiles and datcats)
- detail-info on hover
- full HTML-rendering of a node (Profile, Component)
- backlinking from detail (e.g. view all the profiles a data category is used in by clicking on the number (’used in profiles’)
- store/export SVG/PDF/PNG-renderings of the graphs
- add edge-weight: scale based on usage, i.e. how often appears the relation in the complete dataset i.e. often reused combinations of components/elements will be nearer
- allow to blend in further (private) CMD-profiles dynamically

D.3 Sample SMC graphs

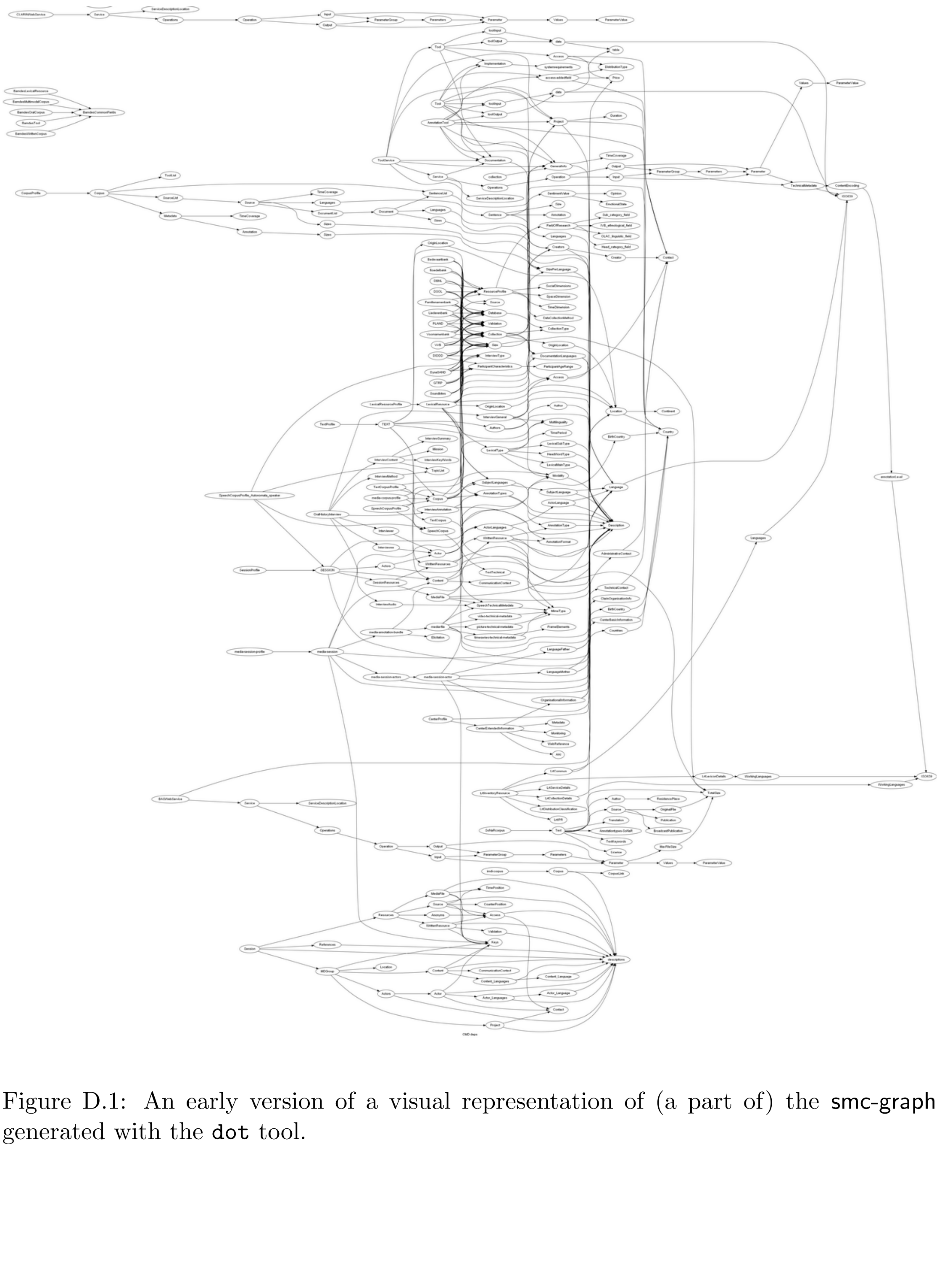


Figure D.1: An early version of a visual representation of (a part of) the smc-graph generated with the dot tool.