The CMDI MI Search Engine: Access to Language Resources and Tools Using Heterogeneous Metadata Schemas

Junte Zhang, Marc Kemps-Snijders, and Hans Bennis

Meertens Institute, Royal Netherlands Academy of Arts and Sciences

Abstract. The CLARIN Metadata Infrastructure (CMDI) provides a solution for access to different types of language resources and tools across Europe. Researchers have different research data and tools, which are large-scale and described differently with domain-specific metadata. In the context of the Search & Develop (S&D) project at the Meertens Institute within CLARIN, we present a system description of an advanced search engine that semantically converges differently structured metadata records based on CMDI for search and retrieval. It allows different groups of users – such as language researchers – to search across yet unexplored research data and locate relevant data for new insights, and find existing tools that could provide novel use cases.

1 Introduction

The Common Language Resources and Technology Infrastructure (CLARIN) initiative seeks to establish an integrated and interoperable research infrastructure of language resources and its technology.[1] Descriptive metadata is used to characterize large number of (legacy) data resources (collections) and tools (e.g. Web services) to facilitate their management and discovery. The Search & Develop (S&D) project within CLARIN in the Netherlands uses the Component MetaData Infrastructure (CMDI; [2]) to open up the sharing of resources and Web services for people and machines first within the collections of a single institution, then across institutions in the Netherlands and eventually across Europe as whole. This infrastructure enables new research methods in language research and stimulates the Digital Humanities, where new insights can be gained by combining and reusing resources from different institutions and domains, and existing tools can be more effectively found and reused based on new insights.

But how to search for data and services, which can be understood by both people from varying disciplines and machines? The challenge is that the data is heterogenous both in content and structure, and can be massive in amount. This flexibility encourages the exchange of resources and services, but at the same time is a challenge for access. Our aim is to develop a unified solution that can make language resources and research data and tools across collections and even institutions accessible and usable for advanced users like Humanities researchers based on metadata. We present our solution to this challenge with the CMDI Meertens Institute (MI) search engine.

[1] See `http://www.clarin.eu/external/index.php?page=about-clarin`

P. Zaphiris et al. (Eds.): TPDL 2012, LNCS 7489, pp. 492–495, 2012.

2 System Description of the CMDI Search Engine

In this section we present the system description of the CMDI MI search engine, and we focus on our robust indexing method and user interface. This search engine is driven by CMDI metadata, which is used as the first step to gain access to resources.

2.1 CMDI Files

A CMDI file in XML has a **<Header>**, **<Resources>**, and **<Components>**. The former 2 are fixed, while the content and structure within **<Components>** is flexible and can encapsulate any data in any structured form. An XML schema can be used to make CMDI files coherent in structure for a (sub)collection and it contains references to ISOcat data categories (DC) stored in the Registry (DCR; [4]). The DCR was established by the *ISO Technical Committee 37, Terminology and other language and content resources* based on the ISO 12620:2009 standard. Because multiple elements may refer to the same DC, a certain degree of semantic interoperability can be achieved across different datasets. A specification using the DCR and projected for example in an XML schema is called a *profile* and can be (re)used for describing datasets.

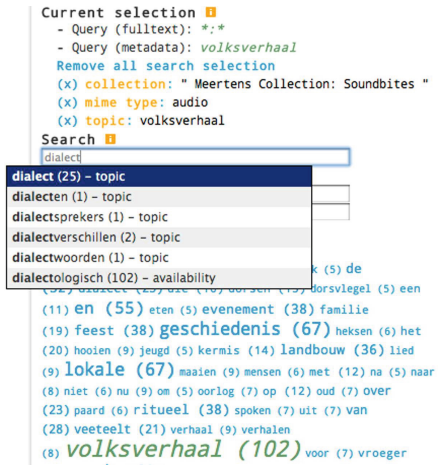
We initially have indexed different CMDI datasets available across Europe and at the MI such as the Dutch Song Database, which have been converted from metadata of legacy research datasets. At the MI, these are primarily Dutch language resources that describe variation in language and cultural data resulting from ethnographic research.

2.2 Indexing Method

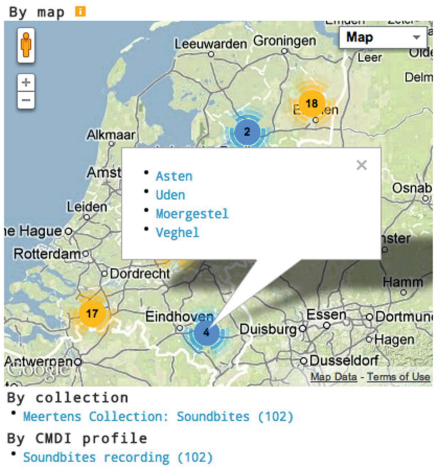
The CMDI MI search engine is driven by the Apache Solr enterprise search server that uses the Lucene Java search library. This backend allows us to index the data efficiently and implement state-of-the-art Information Retrieval (IR) features. We use the standard relevance ranking based on the Vector Space Model. We have not yet enabled stop-word removal or conducted stemming, but this can be supported and fine-tuned in the future.

The following procedure is used to index the CMDI files, and can be used robustly to index even more diversely structured CMDI files from more institutes in a loop. We use an XML schema parser to index on the XML element level by using XPath expressions for focused access. XML elements are disambiguated with their parent.

- Export all CMDI files from the MySQL database, FTP, etc with a script.
- Create a list of the CMDI files that needs to be indexed.
- Use the XML Schema parser to link CMDI elements to their ISOcat DC and
  - run a Perl script based on all XML schemas and create a single Lucene indexing schema for all the CMDI files.
  - create an XSLT stylesheet β with a script and use β to convert CMDI elements to the Lucene indexing format with templates for each XPath. The same elements with different XPaths are defined by setting the 'priority' attribute based on the XPath length. Elements without ISOcat DC are indexed as fulltext.
- Extract the ISOcat IDs as unique index field names and extract with the XML Schema parser the corresponding 'human-readable' (semantic) labels for metadata 'relabeling' in the user interface with a parallel array.

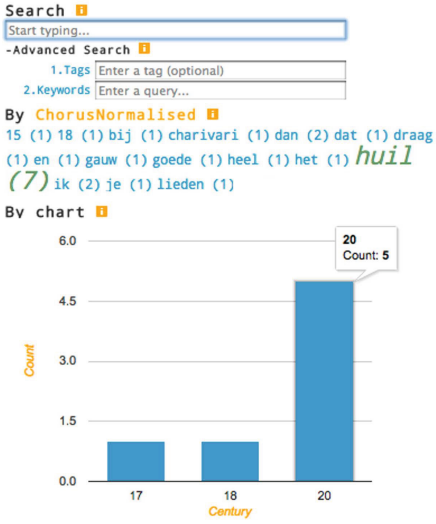


(a) Query autocompletion based on the result counts and the tags, and preserving the overview of the search trail and allow users to change it (in any order). The last used query is 'volksverhaal' (*Folktale*).



(b) The system projects location-based results on a map, zooms and aggregrates by clustering the markers. Users can directly click to go to the resources. The lists in 'collection' and 'CMDI profile' are kept in sync.

(c) Display of results with snippets and keywords in context within the last searched metadata label and the presentation of all used keywords in context given the fulltext. For each retrieved result in the list, there is a recommendation (when available) of related results based on the content similarity of the last used metadata label, or else the fulltext.



(d) Advanced Search allows user with (some) prior knowledge of the metadata schemas to select a tag, and then enter a query aided by the autocompletion feature based on that tag. Besides a tag cloud, the system uses bar and line charts to display aggregrated time-based results (in tags 'Century', 'Year'). The results are narrowed down by clicking in the chart.

Fig. 1. The user interface of the CMDI MI search engine and its advanced features

We have automated this indexing method with a shell script in Linux, so that we can set a cron job, for instance, as the CMDI files get revised or a collection gets extended with a batch of new CMDI files. Our results show that this method is robust, as we can index 246,728 CMDI files from 18 different XML (metadata) schemas in a single stream. We have indexed 150 different types of elements (based on the ISOcat DCs).

2.3 User Interface

For the user interface in Fig 1, we employ, modify, and extend the JavaScript library AJAX Solr[2], which allows for faceted search that matches with the information seeking behavior of Humanities users [1, 3]. It is setup with widgets, where the search box and features of Fig. 1(a), 1(b) and 1(d) are located on the left and the result list and display of Fig. 1(c) are presented on the right side. Users can express queries within the tags and retrieve snippets within them for focused search, or else search in the fulltext.

The system supports serendipity with query autocompletion that directly includes the hit counts, a tag cloud with keyword highlighting to give an overview on an aggregrated level of the results to support query expansion, recommendation of similar results based on content similarity, grouping of results by clicking on subject headings, display on an aggregrated level of the results based on temporal (charts) and geographical (maps) information, and different displays of the result types based on CMDI profile. It keeps track of the cumulative search trail and allows users to revise it in any order.

3 Conclusion

We have presented the CMDI MI search engine, which includes the concise description of CMDI metadata, the indexing method, and the user interface. The novelty is that it is a common (semantic) gateway to diversely structured descriptions of language resources with different metadata schemas for users such as Humanities researchers with very specific and complex information (research) needs. It is a tool that provides focused and interactive access to heterogeneous metadata of (legacy) language research datasets and tools, supports serendipity, and provides new insights for further research and development. It can be found and used at www.meertens.knaw.nl/cmdi/search.

References

[1] Bates, M.J.: The design of browsing and berrypicking techniques for the online search interface. Online Review 13(5), 407–424 (1989)

[2] Broeder, D., Kemps-Snijders, M., Uytvanck, D.V., Windhouwer, M., Withers, P., Wittenburg, P., Zinn, C.: A data category registry- and component-based metadata framework. In: LREC. European Language Resources Association, ELRA (2010)

[3] Hearst, M.A., Karadi, C.: Cat-a-cone: an interactive interface for specifying searches and viewing retrieval results using a large category hierarchy. In: SIGIR, pp. 246–255. ACM, New York (1997)

[4] Kemps-Snijders, M., Windhouwer, M., Wittenburg, P., Wright, S.E.: ISOcat: remodelling metadata for language resources. IJMSO 4(4), 261–276 (2009)

[2] See `https://github.com/evolvingweb/ajax-solr`