

The CMDI Explorer

Denis Arnold

Leibniz-Institut für Deutsche Sprache
Mannheim, Germany
arnold@ids-mannheim.de

Ben Campbell

Eberhard Karls Universität
Tübingen, Germany
ben.campbell@uni-tuebingen.de

Thomas Eckart

Universität Leipzig
Leipzig, Germany
teckart@informatik.uni-leipzig.de

Bernhard Fisseni

Leibniz-Institut für Deutsche Sprache
Mannheim, Germany
fisseni@ids-mannheim.de

Thorsten Trippel

Eberhard Karls Universität
Tübingen, Germany
thorsten.trippel@uni-tuebingen.de

Claus Zinn

Eberhard Karls Universität
Tübingen, Germany
claus.zinn@uni-tuebingen.de

Abstract

We present the CMDI Explorer, a tool that empowers users to easily explore the contents of complex CMDI records and to process selected parts of them with little effort. The tool allows users, for instance, to analyse virtual collections represented by CMDI records, and to send collection items to other CLARIN services such as the Switchboard for subsequent processing. The CMDI Explorer hence adds functionality that many users felt was lacking from the CLARIN tool space.

1 Motivation

A scientific resource often comprises many different parts. A proper description of such a project with metadata according to CLARIN standards will yield a rich metadata record that lists each of the significant parts with detailed information. Consider the following example. A doctoral project that investigates the acquisition of language in small children might involve a number of experiments where babies are exposed to various visual and auditive stimuli, where eye tracking and other sensor data is used to observe their reactions, and where various Python and R scripts are employed to manipulate and analyse such data automatically. To describe such study, the doctoral candidate will attach, for instance, the media type to each stimulus, describe the nature of the sensor data, or refer to each of the processing scripts and the order they need to be executed. Rich metadata makes it easier for others to follow-up on research, say, when trying to reproduce research results, or to build a follow-up project on existing work, say by conducting a meta-study where the work of our doctoral student is taken to be one of many similar studies. A proper description of the meta-study, in turn, will yield a yet more complex metadata record, now describing the meta-study and how it has used the individual studies in the amalgamation.

Reading and processing complex metadata is no trivial matter. In this paper, we propose a tool that supports researchers to work with highly structured metadata and its associated research data.

2 Background

With CMDI being the de-facto standard for metadata in the CLARIN world, our community has built a good range of tools that process, in some way or another, CMDI metadata (Broeder et al., 2012).

The CLARIN *Virtual Language Observatory* (VLO; <https://vlo.clarin.eu>) gives researchers access to hundreds of thousands of language-related resources via their metadata descriptions (van Uytvanck et al., 2012). At regular intervals, its back-end engine harvests CMDI-based metadata from many different metadata providers. It needs to analyse these CMDI records, which adhere to many

This work is licenced under a Creative Commons Attribution 4.0 International License. License details: <http://creativecommons.org/licenses/by/4.0/>

different metadata profiles, for content to correctly fill the various facets (*e.g.*, *language*, *resource type*, *modality*, *format*) that users will use to conduct faceted search in the VLO front-end. Once users have found a resource of interest, its individual metadata records are presented in a tabbed user interface, which includes a listing of its constituent parts via a simple hierarchical representation (*cf.* Fig. 1). Many descriptions in the VLO, however, are highly structured CMDI records. Navigating such metadata in such a tabbed environment, where an tabular entry points to a complex structure, involves following persistent identifiers attached to substructures manually. It can hence take some time to identify a sub-tree's leaves where, say, the auditive stimuli of a study of interest are being found.



Figure 1: A simple structured CMDI record in the VLO.

The CLARIN *Virtual Collection Registry* (VCR; <https://collections.clarin.eu>) enables scientists to assemble resources of interest into a virtual collection via persistent identifiers (PIDs) that refer to their individual metadata (Elbers, 2017). With virtual collections themselves being referred to by PIDs, scientists can easily create collections that have references to simple elements (such as single publications), and to complex elements (such as other virtual collections). It is hence easily possible to construct highly-structured virtual collections. The VCR is the primary entry point for scholars to create new virtual collections, share them with others, and browse through shared collection records. The portal offers some basic search functionality; it also provides a lean presentation of associated resources (*cf.* Fig. 2), which however, fails short at mirroring the potentially hierarchical structure of a collection.

The CLARIN *Language Resource Switchboard* (<https://switchboard.clarin.eu>) makes it easy for users to identify and invoke software tools that can process a language-related resource in one way or another (Zinn, 2018). The Switchboard's tool space, however, is geared towards, so to speak, the leaves of CMDI record trees, the actual scientific resources such as their text or audio files. The Switchboard cannot, for instance, handle a CMDI file that, say, describes a *plain* set of text files, which users will want to batch-process one by one with the same chosen tool.

Both the VLO and the VCR stop analysing CMDI files when it comes to resolving hierarchical structures marked by `ResourceProxy` lists of type `Metadata`. Unaware of the deep hierarchical structures behind a CMDI file, both VCR and VLO fail to offer users the crucial capability to navigate through them. Hence, users cannot easily explore those structures, select parts of them, say, to download them for off-line processing, or to send them to the Switchboard for a further analysis.

Other tools face similar deficits. The CLARIN community offers a number of converters from CMDI to bibliographic metadata standards such as Dublin Core or MARC 21 (Zinn et al., 2016). These converters

Resources		
Type	Reference	Actions
Resource	Peer Gynt (in English) An untraced edition, apparently from 1875.	...
Resource	HDL 2027/njp.32101068574639	...
Resource	HDL 2027/wu.89035754027	...
Resource	HDL 2027/uc1.32106002253281	...
Resource	HDL 2027/mdp.39015005058386	...
Resource	HDL 11245/1.146820 Dutch PhD thesis (with English abstract) by R.G.C. van der Zalm	...
Resource	HDL 1874/250346 Dutch BA thesis by L.G. de Jong	...

Figure 2: A simple structured CMDI record in the VCR.

either show similar shortcomings when it comes to processing highly structured CMDI files, or are tailored to specific CMDI profiles where hence structural complexities are known in advance.

There are a couple of tools that go the extra mile of processing highly structured CMDI files: *SMC Browser* (Đurčo, 2013), see <https://clarin.oeaw.ac.at/smc-browser/index.html> and *Curation Module* (King et al., 2016; Ostojic et al., 2017), see <https://curate.acdh.oeaw.ac.at/>) Both applications focus on a computer-assisted quality assessment of CMDI files. The Curation Module aims at providing statistical information relevant to the evaluation of the usability of a metadata instance. This includes link resolution checks, and an evaluation of a record's adequacy for faceted search in the VLO. The focus on quality assurance makes these tools mostly suited for metadata creators and publishers, not for the general user.

In sum, the VLO, VCR, and the Switchboard would profit from software that crosses navigational boundaries. The CLARIN *CMDI Explorer* aims at complementing (and supporting) the CLARIN tool space with the much-needed functionality of handling complex CMDI metadata. It provides a simple way of accessing all data files associated with a resource described by a CMDI metadata file. For this, it accesses the `ResourceProxy` list of a (possibly recursive) CMDI file and provides a navigable tree overview of all files associated with a collection. Each individual file can then either be downloaded directly, or be send to the Switchboard for further processing. The CMDI Explorer also allows users to download all data files (or a selection thereof), pending on license restriction, for off-line usage.

3 The CMDI Explorer

In line with the other pillars of the CLARIN infrastructure, the CMDI Explorer is implemented as a web-based application. Similar to the Switchboard, users have three options to enter their CMDI metadata: they can enter a PID that resolves to the metadata, upload a metadata file from their local computer, or copy and paste metadata they have found elsewhere. Similar to the Switchboard, the Explorer has an open communication channel with the VLO and the VCR. That is, VLO and VCR users will get the option to send a CMDI file to the Explorer for further analysis.

The CMDI Explorer is designed to analyse highly recursive CMDI files and to provide a tree-based representation and visualisation of its entire structure, both in the browser and as a downloadable HTML and JSON file. The CMDI Explorer can also work with resources composed of constituents that are described using different CMDI profiles: As it only relies on the resource proxy list, it will just assemble all files referenced there irrespectively of their role in the CMDI file. As it is designed to work with collections, the CMDI Explorer can operate across repositories, provided the metadata is freely available,

i.e., not behind an Authentication and Authorization Infrastructure (AAI) wall.

The implementation of CMDI Explorer uses Java for the back-end and `react-js` for the front-end.

Back-end. The basic algorithm for retrieving the data associated with a collection is as follows: first, the CMDI XML data associated with the PID of the collection is retrieved. The resource proxy list is then extracted and each resource is analyzed. Resources with the resource type `Resource` are downloaded and saved as files. For resources with the resource type `Metadata`, the CMDI XML data is retrieved and the process is repeated recursively until all information for all files associated with the collection has been retrieved. The file information is then stored in a tree structure corresponding to the structure of the collection with the node names based on the names of the various collections and files, with the corresponding PID being added as a prefix to each file name in order to avoid any name collisions.

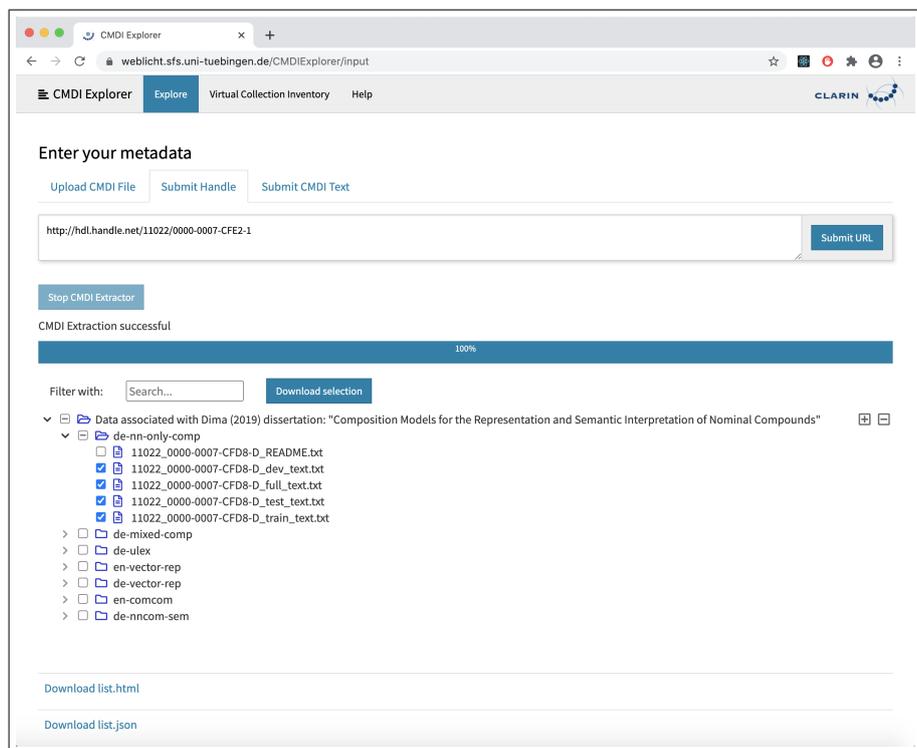


Figure 3: A CMDI record displayed in the CMDI Explorer

There were a number of challenges involved with the extraction of the data. Firstly, it is not always straightforward to extract the CMDI XML data associated with a PID. For each PID URL, there are a number of redirects to go through before arriving at the final URL. The path that the redirects take is also affected by the value of the `Accept` header. It was found that this needed to be set to `application/x-cmdi+xml`, `application/xml+cmdi`, `application/xml` for both following the redirects as well as extracting the data from the URL to cover the different possible Media Types the CMDI XML data could have as its `Content-Type`. There are also a number of CMDI metadata resources in certain collections that are mislabelled as `Resource` as their resource type, when the resource type should be `Metadata`. In order to solve this, all resources labelled as `Resource` with an XML, CMDI, or no Media Type are analyzed and checked if they can be analyzed as a CMDI file, *i.e.*, checking whether the file contains a resource proxy list which is non-empty. If so, then the resource is analyzed as CMDI metadata. Another problem is that some resources are inaccessible for various reasons, such as the need for authentication. Inaccessibility of resources is indicated in the data tree and the corresponding HTML or JSON files. Moreover, some collections are ‘circular’, *i.e.*, a resource in the CMDI metadata of one resource will eventually lead back to the original resource. The Explorer keeps

records of resources encountered so far, and any resource that has already been analyzed is simply not included in the tree that is being constructed.

Front-end. Fig. 3 shows the main interface, presenting users the three options to enter their input.

Once the metadata has been submitted, the Explorer's back-end attempts to extract its underlying tree. During extraction, users see a progress bar, and live updates to the emerging tree visualisation. Fig. 3 shows a top node together with its immediate children. Each branch of the tree can be unfolded individually, and there are also two controls (+, -) to unfold or fold the entire tree. Leaf nodes have actions attached to them: (i) copy the handle to the clipboard, (ii) send the handle to the Switchboard, (iii) click on the handle so that it resolves in the browser, and (iv) download the resource referred to by the handle. Also, the entire tree can be downloaded in a structure-preserving HTML or JSON format. Moreover, users can select nodes individually. Selected nodes will be added to the 'download basket'; the corresponding data resources of their selection are made available as a ZIP archive file.

4 Current State and Future Work

We have built a prototype of the CMDI Explorer that implements its core functionality, and which is now open for beta testing at <https://weblicht.sfs.uni-tuebingen.de/CMDIExplorer/>. We invite the reader to explore the tool and encourage their feedback. Given the Explorer's lineage from the Switchboard's codebase, we expect users to easily grasp its user interface and the functionality it offers.

The assemblage of resources from archives becomes a complex task when resources are protected by usage rights or licenses, and hence by AAI protocols. We are aware of the issue but are unsure which path to take as we do not want to move the Explorer behind a Shibboleth wall.

The CMDI Explorer has already been included in the test version of the Switchboard as a metadata processing tool. We expect the Explorer to be connected to the VLO and the VCR as well so that users can easily invoke it wherever they find complex CMDI metadata they need to explore further.

Acknowledgements

Our work was funded by the German Federal Ministry of Education and Research (BMBF), the Ministry of Science, Research and Art of the Federal State of Baden-Württemberg (MWK), and CLARIN-D. Emanuel Dima, Willem Elbers, Dirk Goldhahn, Marie Hinrichs and Dieter van Uytvanck participated in the initial discussion and contributed to the conceptualisation of the explorer.

References

- Daan Broeder, Menzo Windhouwer, Dieter Van Uytvanck, Twan Goosen, and Thorsten Trippel. 2012. CMDI: a component metadata infrastructure. In *Describing LRs with metadata: towards flexibility and interoperability in the documentation of LR workshop programme*, volume 1.
- Willem Elbers. 2017. Virtual collection registry v2. Technical report, CLARIN ERIC.
- Margaret King, Davor Ostojic, Matej Ďurčo, and Go Sugimoto. 2016. Variability of the facet values in the vlo – a case for metadata curation. In Koenraad De Smedt, editor, *Selected Papers from the CLARIN Annual Conference 2015*, pages 25–44, Linköping. Linköping University Electronic Press.
- Davor Ostojic, Go Sugimoto, and Matej Ďurčo. 2017. Curation module in action – preliminary findings on vlo metadata quality. In *Proceedings – CLARIN Annual Conference 2016*.
- Dieter van Uytvanck, Herman Stehouwer, and Lari Lampen. 2012. Semantic metadata mapping in practice: the virtual language observatory. In Nicoletta Calzolari et al., editor, *Proceedings of LREC'12*, Istanbul, Turkey. ELRA.
- Claus Zinn, Thorsten Trippel, Steve Kaminski, and Emanuel Dima. 2016. Crosswalking from cmdi to dublin core and marc 21. In Nicoletta Calzolari et al., editor, *Proceedings of LREC'16*, Portorož/Paris. ELRA.
- Claus Zinn. 2018. The language resource switchboard. *Comput. Linguist.*, 44(4):631–639.
- Matej Ďurčo. 2013. *SMC4LRT – semantic mapping component for language resources and technology*. Ph.D. thesis, Technische Universität Wien.