Discovering software resources in CLARIN

Jan Odijk
UiL-OTS
Utrecht University, the Netherlands
j.odijk@uu.nl

Abstract

I present a CMDI profile for the description of software that enables discovery of the software and formal documentation of aspects of the software, and a proposal for faceted search in metadata for software. The profile has been tested by making metadata for over 80 pieces of software. The profile forms an excellent basis for formally describing properties of the software, and for a faceted search dedicated to software which enables better discoverability of software in the CLARIN infrastructure. A faceted search application for this purpose has been implemented. A curation procedure is proposed to ensure that descriptions of software made on the basis of other profiles contain the relevant information in the right form and use the right vocabularies, and we created an experimental faceted search that includes software descriptions based on the WebLichtWebService profile.

1 Introduction

Enabling the easy discovery of resources is an important goal of CLARIN. The Virtual Language Observatory (VLO) serves this purpose, but it is currently mostly suited for the discovery of *data*. Discovering *software* is not so easy in the current VLO. The (pretty complex) query Software Query[1] approximates finding all software descriptions in the VLO. It finds 1219 descriptions of software (on 2019-01-11). However, there are no facets dedicated to software to refine one's search. In order to address this issue I present a CMDI profile for the description of software (ClarinSoftwareDescription, CSD) that enables discovery of the software and formal documentation of aspects of the software (section 2). The profile has been tested by making metadata for over 80 pieces of software (section 3). I also describe how the quality of these metadata descriptions was ensured (section 4). I present a proposal for faceted search in metadata for software (section 5). An experimental version of the proposed faceted search has been implemented. I propose to add this faceted search to the VLO. It should then also cover descriptions of software created on the basis of other profiles. I show how metadata curation software, combined with provided metadata curation files, can curate existing metadata descriptions for software using other profiles to make them suited for this faceted search (section 6). An experiment with the WebLichtWebService profile was carried out, resulting in a faceted search covering not only CSD but also WebLichtWebService based descriptions of software. In section 7 I summarise the main findings, point out some problems encountered, indicate required future work, and make some recommendations.

2 Metadata Profile ClarinSoftwareDescription

The ClarinSoftwareDescription (CSD) profile[2] enables one to describe information about software in accordance with the CMDI metadata framework used in CLARIN (Broeder et al., 2010; Broeder et al.,

[1] https://vlo.clarin.eu/search?0&fq=resourceClass:R-code&fq=resourceClass:software&fq=resourceClass:Tools&fq=resourceClass:web+application&fq=resourceClass:webservice&fq=resourceClass:software,+webservice&fq=resourceClass:web+service&fq=resourceClass:tool+service&fq=resourceClass:software&fqType=resourceClass:or

[2] clarin.eu:cr1:p_1342181139640.

Jan Odijk 2019. Discovering software resources in CLARIN. *Selected papers from the CLARIN Annual Conference 2018*. Linköping Electronic Conference Proceedings 159: 121–132.

2012). The profile has been set up in such a way that it enables (1) the description of properties that support discovery of the resource, and (2) the description of properties for documenting the resource, in as formal a manner as possible.

I briefly describe the major components and elements of the profile. The elements crucial for finding the resource are dealt with in more detail in section 5.

The profile consists of the CMDI components *Generalinfo*, *SoftwareFunction*, *SoftwareImplementation*, *Access*, *ResourceDocumentation*, *SoftwareDevelopment*, *TechnicalInfo*, *Service* and *LRS*.

The component *Generalinfo* enables one to describe general information about the resource. It is an extension of the component *cmdi-generalinfo*.[3] It includes elements for the *name* and *title* of the resource, its *version*, *publication year*, *owner* and contact *email address*, a *URL* and/or a *PID* to the resource, its *time coverage*, *release status*, *CLARIN Centre* hosting the resource and the *national project(s)* in which it has been made part of the CLARIN infrastructure, any *alternative names* for the resource and a *description* of the software.

The *SoftwareFunction* component enables one to describe the function of the software in terms of the closed vocabulary elements *tool category, tool tasks*, *research phase(s)* (for which it is most relevant), *research domains* and, for the linguistics domain, relevant *linguistic subdisciplines* for which it was originally developed.[4] It also describes the *language variants* that the software applies to and offers facilities for documentation about its *performance*.

The *SoftwareImplementation* component enables one to describe information for users on the implementation and installation of the software. It describes how the software is *distributed*, what the *installation requirements* are, the nature of the *interface* with the user or other software, properties of the *package* that the software is delivered in (if any) and what the *input* and the *output* of the software is.

The input and output specification enable a quite detailed description of properties of the input required and output generated by a piece of software. The following is an example of the output specification for the Alpino parser:

outputType text
characterEncoding utf8
Schema LASSY DTD
MimeType text/xml
AnnotationType
    AnnotationType Morphosyntax/Inflection
    AnnotationType Morphosyntax/Lemma
    AnnotationType Morphosyntax/POS
    AnnotationType Morphosyntax/Word form
    AnnotationType Orthography/Token
    TagSet POSTags/DCOI Tagset
AnnotationType
    AnnotationType Syntax/Chunks
    AnnotationType Syntax/Dependency Relations
    AnnotationType Syntax/Grammatical Relations
    AnnotationType Syntax/Phrases
    AnnotationType Syntax/Syntactic Categories
    AnnotationType Syntax/Multiword Expressions
    TagSet Syntax/Alpino Tagset

It specifies that Alpino yields *text* as output, with character encoding *utf8*, in accordance with a schema called *LASSY DTD*, and with mimetype *text/xml*. Alpino generates multiple annotations, here grouped in two groups because the tag set used differs per group. The first group of annotations concerns *inflection*, *lemma*, *part of speech (POS)*, *word form* and *token*, encoded with the *DCOI Tagset*. The second group involves *chunks*, *dependency relations*, *grammatical relations*, *phrases*, *syntactic categories*, and

[3] clarin.eu:cr1:c_1342181139620.

[4] which, of course, does not preclude its use in other research domains that were not foreseen during development.

multiword expressions, encoded with the Alpino Tagset. This information can be used, together with the metadata of the input data, to automatically generate metadata for the output data generated by Alpino, provided of course that metadata for (textual or other) data use the same annotation labels. The ability to generate such rich metadata for output of tools is very important for data provenance in general, and for applications such as the CLARIN SwitchBoard (see (Zinn, 2016a) and below), which forms a great aid for users for finding applications and services that they can apply to a particular data set.

Note that all values of the metadata elements *AnnotationType* and *TagSet* come from a closed vocabulary. Since the number of possible values is already large (currently 61 different possible values for *AnnotationType*, 10 different possible values for *TagSet*), and since one can certainly expect these numbers to grow, I grouped the values in classes, indicated here by the label before the slash. In this way, closely related values can be inspected together, and one can concentrate on those finegrained distinctions that one is interested in without being bothered by finegrained distinctions that one is not interested in. The importance of such grouping was already pointed out by (Odijk, 2009, 12-13). No support for such a feature is currently available in CLARIN. This is why I opted for the poor man's option of specifying a superordinate category in each value before the actual value separated from it by a slash, but this is clearly to be seen as an ad hoc and temporary solution. For a more principled solution, see section 2.1.

The *Access* component enables one to describe information about the availability and accessibility of the resource. It is an extension of the *cmdi-access* component.[5] It contains a reference to a *catalogue link*, information about the *license* for the resource, information about *copyright* and *copyright holder(s)* and a *contact* organisation and/or person.

The *ResourceDocumentation* component enables one to describe the documentation of the resource. It offers facilities to describe the *documentation* and *publications* on the resource, a *description* of the resource and *pictures* (including *logos*) related to the resource. The *SoftwareDevelopment* component is intended for information on the history and development of the software. It offers facilities to describe the *source*(s) that the software is based on or from which it has been derived, the *project* in which the software was created, the *creator(s)* of the software, and any planned *software updates*. The *TechnicalInfo* component enables one to describe technical information on a resource and is mainly aimed at developers. It provides facilities to describe the *run time environment*, any *access protocols*, as well as the *programming language(s)* that have been used to implement the software.

The *Service* component (CLARIN-NL Web Service description) is intended for describing properties of web services. It is compatible with the CLARIN CMDI core model for Web Service description version 1.0.2.[6]

The *LRS* component is intended for the description of the properties of a particular task for the CLARIN Language Resource SwitchBoard (CLRS, (Zinn, 2016b; Zinn, 2016a; Zinn, 2017)). Multiple LRS components can be present. It is our viewpoint that specifications for an application for inclusion in the CLRS registry[7] should be derivable from the metadata for this application. This was not the case for the CSD profile when the CLRS came into existence, so I added a component to offer facilities for supplying the missing information. I devised a script to turn a CSD-compatible metadata record that contains an LRS component into the format required for the CLRS and tested it successfully with the *Frog* web service and application (van den Bosch et al., 2007). See `https://languagemachines.github.io/frog/` for Frog's source code and `http://portal.clarin.nl/node/8516` for its entry in the faceted search described in Section 5.

2.1 Semantics

Many of the profile's components, elements and their possible values have a semantic definition by a link to an entry in the CLARIN Concept Registry (CCR, (Schuurman et al., 2016)).[8] For the ones that were lacking I created definitions and provided other relevant information required for inclusion into the CCR.

[5] clarin.eu:cr1:c_1311927752326.

[6] This component was created by Menzo Windhouwer, and adapted to the requirements of CMDI version 1.2.

[7] `https://github.com/clarin-eric/LRSwitchboard-rest/blob/master/Registry.js`

[8] `https://concepts.clarin.eu/ccr/browser/`.

I submitted this file (2017-09-08), in the format required, to the maintainers of the CCR.[9] However, the CCR coordinators'[10] mill runs slowly, and, though there has been some activity on the proposed concepts, so far none of them have been incorporated in the CCR. This constitutes a real bottleneck, which should be addressed in CLARIN. After our submission to the CCR, I made some new modifications to the profile, so there are new elements and values for which the semantics does not exist yet.

I also specified relations between concepts in the input, but I was immediately told that that was not supported yet by CCR. It could be implemented in CLARIN by specifying *isa* relations in the CCR. By making use of small taxonomies of concepts derived from this information, the CMDI Component Registry Editor, dedicated CMDI metadata editors, and faceted search facilities can make the work of people who edit profiles and components, create or adapt metadata, or use faceted search considerably more pleasant and more effective. Unfortunately, no such options are currently offered in the CCR.

2.2 Comparison with other profiles for software

There are about 20 profiles for the description of software in the CLARIN Component Registry (as determined on 2017-09-29), but most are not in use or in use for a single description only. The only profiles that are used for multiple software resources (measured on 2019-01-15) are *ToolProfile*[11] (69 resources), *WebLichtWebService*[12] (419 resources), *resourceInfo*[13] (189 software resources), *OLAC-DcmiTerms*[14] (175 software resources), and *LINDAT_CLARIN*[15] (83 software resources). The instances describing software can be identified on the basis of the VLO facet *resource type*, using the query given in Section 1 and repeated here for convenience: Software Query. It finds (on 2019-01-11) 535 descriptions with *resource type= software,web service*[16], 419 descriptions with *resource type = web service*[17], 162 with *resource type = webservice*[18], 181 with *resource type = software*[19], 72 with *resource type = tool service*[20], and a small number of descriptions with other values for *resource type*.

I summarise the most important differences between the CSD profile and the most used profiles: (1) the CSD profile is fully dedicated to the description of software (v. the *OLAC-DcmiTerms* and *LINDAT_CLARIN* profiles); (2) the CSD profile can be used to describe any type of software (v. *WebLicht-Webservice*, which is intended for web services only); (3) CSD offers more elements, and more formalised elements than the other profiles, not only elements useful for discovery but also for (formalised) documentation; (4) CSD offers more and/or more extensive closed vocabularies for many metadata properties, e.g. for *toolTask*, *applicationType*, *ResearchDomain*, *LinguisticsSubject*, etc. Corresponding metadata elements in other profiles usually allow any string as value.[21]

[9] It can be found here: `https://surfdrive.surf.nl/files/index.php/s/oWUgl1664VraCMo`.

[10] https://www.clarin.eu/content/concept-registry-coordinators

[11] clarin.eu:cr1:p_1290431694581.

[12] clarin.eu:cr1:p_1320657629644.

[13] clarin.eu:cr1:p_1360931019836.

[14] clarin.eu:cr1:p_1288172614026.

[15] clarin.eu:cr1:p_1403526079380

[16] `https://vlo.clarin.eu/search;jsessionid=07859474E8155F6D595632FD827F38F4?0&fq=resourceClass:software,+webservice&fqType=resourceClass:or`.

[17] `https://vlo.clarin.eu/search;jsessionid=07859474E8155F6D595632FD827F38F4?0&fq=resourceClass:web+service&fqType=resourceClass:or`

[18] `https://vlo.clarin.eu/search;jsessionid=07859474E8155F6D595632FD827F38F4?0&fq=resourceClass:webservice&fqType=resourceClass:or`

[19] `https://vlo.clarin.eu/search;jsessionid=07859474E8155F6D595632FD827F38F4?0&fq=resourceClass:software&fqType=resourceClass:or`.

[20] `https://vlo.clarin.eu/search;jsessionid=07859474E8155F6D595632FD827F38F4?0&fq=resourceClass:tool+service&fqType=resourceClass:or`.

[21] There are also other desiderata: in particular one would like to avoid what (from the perspective of successful communication) I would like to call the 'horrors of natural language'. One can achieve this by not using existing natural language words as values in a closed vocabulary. Successful communication is seriously hampered by natural language, even in as simple a domain as words or terms: natural language words have associations, have a (common sense) meaning, are often ambiguous, are specific to one language, and have variations (abbreviations, acronyms etc.). These properties make successful communication difficult if not impossible. Good closed vocabulary definitions use arbitrary labels that at best resemble existing words for mnenomical reasons but that are no natural language words. Many standards adhere to this guideline, e.g., the ISO-codes for languages and countries. Unfortunately, they have not been adhered to for all metadata elements in the CSD profile, and here surely improvement is possible.

The *WebLichtWebService* profile is intended for web services only, and it has many relevant properties not represented in a formal way (e.g. there is no formal specification of the language(s) that a web service can apply to, not for Mimetype.

The *ToolProfile* profile is a very good profile, which offers many facilities for specifying properties of software. However, the metadata elements *ToolType* and *ClassificationType* allow any string as value and are not restricted to a closed vocabulary. The same holds for *FieldOfResearch*.

The *resourceInfo* profile for tools, which did not yet exist when I started creating the CSD profile is actually a specific instantiation of a profile for all kinds of resources. It is a profile that has been used mainly to convert META-SHARE descriptions into CMDI descriptions. It offers special elements for the description of software in the component toolServiceInfo[22], with closed vocabulary elements *resourceType* and *toolServiceType*, an open vocabulary element *toolServiceSubtype*, and components *inputInfo*, *outputInfo*, *toolServiceOperationInfo*, *toolServiceEvaluationInfo*, and *toolServiceCreationInfo*, which are all highly relevant to the description of software.

The *LINDAT_CLARIN* profile is intended for all kinds of resources, including software, but it has only very general metadata elements and nothing dedicated to the description of software.

3 Metadata Descriptions using the CSD profile

I have described more than 80 software resources with the CSD profile, and describing these software resources resulted in various improvements of earlier versions of the profile.[23] These software resources mainly concern resources from the Netherlands. Most descriptions started from the information contained in the CLARIN-NL Portal, Services part.[24] The information there was semi-automatically converted to CMDI metadata in accordance with the CSD profile. The resulting descriptions were further extended and then submitted to the original developers and CLARIN Centres that host the resources for corrections and/or additions. The CMDI descriptions can be found here: `https://surfdrive.surf.nl/files/index.php/s/VEJOEkfbFtWR6Y6`. our team is in the process of making them available to the Centres that host the software so that they can be harvested by the VLO.

4 Metadata Quality

All metadata descriptions have been validated against the profile definition. I created several schematron[25] files for issuing errors or warnings for phenomena that are syntactically correct but incorrect or potentially incorrect in other ways. These schematron files check for the presence or absence of important (but optional) elements, for dependencies between elements or their values, e.g. an element to specify the language that the software can apply to must be present unless the value for the element *languageIndependent* is *yes*. I also made a schematron file to check for the presence of elements that are crucial for the faceted search described in section 5. A script has been provided for validation and for applying the schematron files. Additionally, a script was made to identify all URLs in the metadata descriptions and check for their resolution.[26]

The quality checks offered by the CLARIN Curation Module[27] (Ostojic et al., 2017) have also been used but are less useful because they can be applied only to a single metadata description at a time, check for the presence of metadata relevant for faceted search for data in the VLO, many of which are not so relevant for software, and because the profiles are cached so that modifications of the profiles are not immediately taken into account.

[22] clarin.eu:cr1:c_1360931019834

[23] A first version of the profile was presented by (Westerhout and Odijk, 2013), and at that time the profile was tested only on 5 software resources.

[24] `http://portal.clarin.nl/clarin-resource-list-fs`.

[25] `http://schematron.com/`

[26] On 2018-10-01, 969 URLs were correctly found, 11 URLs were found but no access was granted, 35 URLs were not found, and 4 exceptions were raised.

[27] `https://clarin.oeaw.ac.at/curate/`

5 Faceted Search

A major purpose of metadata is to facilitate the discovery of resources. An important instrument for this purpose in CLARIN is the Virtual Language Observatory (VLO, (Van Uytvanck, 2014)). The VLO offers faceted search for resources through their metadata, but its faceted search is fully tuned to the discovery of *data*. For this reason, our team defined a new faceted search, specifically tuned to discovery of *software*. This faceted search offers *search* facets and *display* facets:

Search Facets LifeCycleStatus, ResearchPhase, toolTask, ResearchDomain, LinguisticsSubject, inputLanguage, applicationType, NationalProject, CLARINCentre,

Display Facets name, title, version, inputMimetype, outputMimetype, outputLanguage, Country, Description, ResourceProxy, AccessContact, ProjectContact, CreatorContact, Documentation, Publications, sourcecodeURI, Licence, CMDI File Link, Project, logo or picture, OriginalLocation, and all search facets.

I will discuss search facets in section 5.1, display facets in section 5.2, and end in section 5.3 with a description of the implementation of the faceted search.

5.1 Search Facets

I submit that many of the facets under search facets are very useful for a researcher who is trying to find a piece of software that might be relevant to his/her research. The *ResearchDomain* facet enables the researcher to select the tools that (according to the developers of the software) are relevant to a particular research domain (linguistics, philosophy, literary studies, etc.). For the research domain *Linguistics* further subdivisions can be made using the facet *LinguisticsSubject* (e.g. syntax, phonology, morphology, etc.). The *ResearchPhase* facet enables the researcher to restrict the tools to those tools that are suited for the actual research phase: is the researcher looking for data, does the researcher want to enrich existing data, does the researcher want to search in data, etc. etc. An extensive description of the meaning of this facet and its values (i.e., which research phases are distinguished) is provided here: `http://dev.clarin.nl/node/4723`.

The *toolTask* facet specifies the function(s) of a piece of software, i.e. what does it do? For example, is it a tool for searching in data, for enriching words in text with part of speech tags, for enriching words in text with lemma's, etc. The *applicationType* facet indicates whether the software is a web application, a desktop tool, or a web service, etc. The *inputLanguage* facet is also important, because often a researcher is only interested in a specific language or a small number of languages. The type of input that the tool can work on is also very important, but there currently is no search facet for it. There is a facet for *inputMimeType* but I believe that the large amount of possible values and the fine-grained distinctions made by it make it less suited as a search facet. In the future, I plan to add a facet that can be derived from the *inputMimeType* but has only a limited number of values, basically corresponding to the major modalities and a small number of subtypes (text, audio, audio/speech, video). I also hope to add a search facet for licence class in the future, but for that I first will have to define a limited number of values for licence classes (which I want to be a bit richer than the *Availability* facet in the VLO).

All the facets have values from (what I would like to call) half-open vocabularies. These are basically closed vocabularies, with one special value *other*. These closed vocabularies can be extended, yielding an *updated* closed vocabulary, but they can only be extended with new values with a semantics that does not overlap with the existing values (except for *other*). In the updated vocabulary, all previously existing values retain their original semantics, except for the value *other*, the scope of which is reduced. Such updates will be required regularly, especially in early phases, because no one has a full overview of all the different types of tools, and no one can foresee what new types of tools will come into existence in the future. For this reason, many people use open vocabularies, which of course provides the necessary flexibility, but results in a complete mess and impedes effective search seriously. This has been observed by many (e.g. (Odijk, 2014)) and a curation task force has been set up in CLARIN to reduce the mess resulting from this freedom as much as possible.[28] I try to avoid changes of such vocabularies in which

[28] So far such efforts have only been partially successful, e.g. the situation for the VLO facet *resource type* has been improved significantly recently, but, restricting attention to values for software, the values *software* and *software,webservice*, *Tools* and

the semantics of existing values change, though this may occasionally be necessary (in such a case we speak of an *upgrade* of the vocabulary).

It is crucial for effective search (i.e. easy queries with optimal recall and precision) to have closed vocabularies as much as possible. Values occurring in actual metadata descriptions may have different forms, but it is crucial to map these to values from the closed vocabulary. Regular monitoring of newly occurring values and adapting the curation tables is therefore required, and each national CLARIN project should reserve some effort and money to contribute to this task.

5.2 Display Facets

The display facets form a subset of the full metadata, and contain some additional elements.

The meaning of most display facets is obvious from their names (name, title, version, inputMimetype, outputMimetype, outputLanguage, Country, Description, AccessContact, ProjectContact, CreatorContact, Documentation, Publications, Licence, and Project).

The facet *ResourceProxy* contains one or more links to the actual application(s). The facet *sourcecodeURI* provides a link to the source code of the resource. The facet *CMDI File Link* contains a URL to the full metadata. If the metadata contain a logo or a picture, it is displayed in the faceted search.

The facet *OriginalLocation* contains the URL of the description in the CLARIN-NL Portal, Services part.[29] that the metadata record is based on. It is mainly maintained for future redirection purposes.

5.3 Implementation

Of course, for a faceted search application to work on the metadata offered by the VLO, first of all a distinction must be made between the metadata that describe data and the metadata that describe software. Currently, no such distinction is made, but it can be largely added automatically on the basis of the CMDI profiles used and some existing facets (in particular *resource type*), e.g. by using the query described in section 1.

Furthermore, all metadata profiles for the description of software must be able to provide the values for the facets. That is the case to a large extent, though some metadata curation is needed (in some cases, quite a lot) and existing values must be mapped to the closed vocabulary for use in the faceted search. This is the topic of the next section.

An initial, experimental, implementation of this faceted search has been made available.[30]. It enables one to test the faceted search with many users and to identify errors and omissions in the metadata descriptions. It can thus be tested extensively before it or an improved version of it is included in the VLO. Initial results of this test already led to the suggestion for a new facet, i.e. a facet that indicates what skills a researcher must have in order to be able to use the software, e.g. must the researcher be able to program (and in which language), must the researcher know a particular query language, is extensive knowledge of the structure of a dataset required, etc. etc.

6 Curation of existing metadata for software

I followed the metadata curation strategy sketched by (Odijk, 2015). The basic idea is as follows: a new standardised metadata record is automatically created for all software descriptions, in principle each time a record is harvested. This metadata record contains the components and elements that are required for the faceted search as defined above. The record is constructed from the original CMDI record for the resource, combined with the data for this resource contained in a curation file, by a script. The curation file contains a sequence of conditions on each relevant element, and a specification of which values for which elements should be included in the new record if all the conditions are met. In general, the conditions simply test for identity with a value. The curation file basically consists of two *XSV* files, one specifying the conditions, and the other to specify the changes that must be made (mostly: set an element to a particular value). An *XSV* (eXtended Separated Value) file is a CSV file in which each value can itself

tool service, *Web service*, *web service* and *webservice* exist next to one another. Values for the *resource type* facet for data are a much greater mess.

[29] `http://portal.clarin.nl/clarin-resource-list-fs`.

[30] `http://portal.clarin.nl/clariah-tools-fs`

consist of multiple values separated by a separator. Working with XSV files is very easy, but imposes some limitations, which probably can be overcome by using XSLT. The curation file can be used to add information that was lacking or only present in an unformalised way, and it can be used to map existing values to other values from a specific closed vocabulary. I report on experiments with such a curation file for the *WebLichtWebService* profile, since curation was most needed and most complex for this profile.

The *WebLichtWebService* profile lacks many elements that are necessary for faceted search, e.g. *toolTask*, *researchDomain*, *linguisticsSubject*, *inputLanguage*, *outputLanguage*, *Country*, *CLARINCentre*, *Documentation*, *Publications* and *license*. I made a curation file for many of these properties, which can be used to add the relevant information in a new metadata record for a WeblichtWebService description: this is the case for the facets *toolTask*, *researchDomain*, *linguisticsSubject*, *inputLanguage*, *outputLanguage*, and *Country*.

It may be surprising that the WebLichtWebService lacks formal representations for input and output language, because many of these web services function properly in the WebLicht environment (Hinrichs et al., 2010). The descriptions indeed contain information about the input and output languages, but it is hidden in parameters which have a parameter name (e.g. *lang*), without an explicit meaning, and a parameter value (e.g. *de*), also without an explicit meaning. Therefore, this information is insufficiently formally encoded, and only an intelligent human being can perhaps interpret this. The same holds for input and output Mimetype specifications. Two web services may still interact correctly if the same parameter name and values are used for language and Mimetype in all WebLicht web services. This appears to be the case for Mimetype (parameter name *type*), but not for language (mostly *lang* is used as the parameter name, but occasionally *language* also occurs. For values, both *de* and *Deutsch* occur as values to specify, I assume, the German language, and both *en* and *English* occur as values to specify, I assume, the English language.

An initial, experimental, and still incomplete version of faceted search that includes 286 (partially) curated software descriptions that are based on the WebLichtWebService profile has been made available.[31].

I still have to make curation files for the *ToolProfile*, *resourceInfo* and the *OLACDcmiTerms* profiles. I already inventoried the problems for the first two profiles, and curation files for these will be much simpler than the one for the *WebLichtWebService* profile.

The *ToolProfile* profile has elements for most facets. All query facets can be derived from existing fields except for *ResearchPhase*. Some elements use open vocabularies and require a mapping to standardized values (e.g. *FieldOfResearch* from which *researchDomain* and *linguisticsSubject* can be derived). Elements for the display facets *NationalProject*, *Publication*, *SourceCodeURI*, *CLARINCentre*, and *picture* are lacking.

The *resourceInfo* profile also has elements for most facets. It lacks elements for the query facets *LifeCycleStatus*, *ResearchPhase*, *researchDomain*, *linguisticsSubject*, *NationalProject*, and *CLARINCentre*. It lacks elements to derive the display facets *sourcecodeURI* and *picture*.

I still have to investigate the *LINDAT_CLARIN* profile.

7 Concluding Remarks

7.1 Summary

I presented a CMDI profile for the description of software that enables discovery of the software and formal documentation of aspects of the software, and a proposal for faceted search in metadata for software. The profile has been tested by making metadata for over 80 pieces of software. The profile forms an excellent basis for formally describing properties of the software, and for a faceted search dedicated to software which enables better discoverability of software in the CLARIN infrastructure. A faceted search application for this purpose has been implemented. A curation procedure has been proposed to ensure that descriptions of software made on the basis of other profiles contain the relevant information in the right form and use the right vocabularies, and our team created an experimental faceted search that includes software descriptions based on the WebLichtWebService profile.

[31] `http://portal.clarin.nl/clariah-tools-fs-global`

I encountered some problems or less desirable features of the CMDI infrastructure, of which I will briefly mention some:

1. Closed vocabularies are defined with an element, not as a separate and reusable enumerated type. I believe that this is a very unfortunate design decision, which has many negative effects, in particular, it is not possible to reuse this closed vocabulary. This problem has been partially solved by CLAVAS, but the maintainers of CLAVAS only want to accept widely accepted and used and rather stable vocabularies. An additional (minor) problem is that copying the vocabulary is only possible in the Component Registry by editing the element.
2. CMDI offers no possibility to reuse metadata *elements*: one can only reuse components, not elements, which (especially in combination with the previous point) creates many problems (e.g. I cannot reuse the closed vocabulary for *license* from the *resourceInfo* profile, which is the most extensive list of license types in the whole CMDI infrastructure). If one wants to stimulate reuse of an element in one's profile, one has to put an (otherwise unnecessary) component on top of the element.
3. There is a lot of variety in the contents of the CMDI envelope element *MdSelfLink*, resulting in several unresolved or syntactically incorrect results. One special case is that the *MDSelfLink* refers to an OAI-PMH description containing the metadata rather than to the metadata itself.
4. Lack of good CMDI metadata editors. Though there are some CMDI editors (e.g. Arbil, CMDI Maker, COMEDI), all have severe limitations, e.g. none supports CMDI 1.2. Arbil is a desktop application (which is OK) but requires a steep learning curve and is not really supported any more. *CMDI Maker*[32], despite its name, only supports the IMDI profile. ProForma[33] has been discontinued. COMEDI[34] (Lyse et al., 2015) is a web-based editor, and it suffers from most of the problems that most web interfaces have (Odijk, 2018), which makes it not easy to use for metadata entry. It remains to be seen whether the editor based on the CLARIAH CMDI Forms based approach proposed by (Zeeman and Windhouwer, 2018) will be any better in this respect, but I am not optimistic.

7.2 Future work

The work on the profile and the faceted search has not finished yet. In particular,

- The CSD profile must still be published in the CMDI registry. I did this in an earlier phase, but because of a bug in the CMDI registry for published profiles that are still under development, it was impossible for a team member to edit components originally created by another member of the development team. Therefore, the publishing was partially undone.
- The semantics of the metadata elements has to be finished (cf. section 2.1).
- The documentation of the profile has to be finalised.
- In the metadata descriptions I did not systematically distinguish between input and input parameters. This distinction should be drawn more sharply, and it will probably require an improvement of the facilities for describing parameters. In addition, the profile should enable descriptions of triples of parameters, input and output. This will reduce the need for the (somewhat ad-hocly added) LRS component.
- Some details must still be harmonised. This involve mainly adapting the systematic naming conventions that were adopted but could not be maintained because I reuse components developed by others who follow different naming conventions.
- Due to the long development time of the profile by multiple persons some redundancies have occurred in the profile, which should be removed.
- The faceted search should be extended for other profiles that describe software.
- I would like to derive metadata information that is created or generated in other initiatives as much as possible in an automatic manner, with options for regular (automated) updates. Specifically, parts of

[32] http://cmdi-maker.uni-koeln.de/
[33] http://www.sfs.uni-tuebingen.de/nalida/proforma/
[34] http://clarino.uib.no/comedi/page

the metadata description should be derived automatically from CLAM[35] and WADL[36] descriptions for web services, and from descriptions originating from the codemeta[37] initiative.

I hope to work on these issues in the CLARIAH-PLUS project.

7.3 Recommendations

I end with some recommendations. Some of these follow directly from issues raised earlier, others were not mentioned before but result from our experiences in working with metadata:

- (to CLARIN ERIC) Set up a faceted search in the VLO dedicated to the discovery of software. The proposal sketched here can form a basis to start from.
- (to national coordinators) Coordinate metadata creation nationally. If every individual researcher or data centre manager creates metadata in isolation, the resulting metadata will be very diverse, use mutually incompatible vocabularies, vary enormously in quality and fine-grainedness, and will often lack important metadata information.
- (to national coordinators) Every national consortium must reserve effort (hence money) for active participation in the metadata curation task force. This is necessary because real work will only be done if people have been assigned an explicit task and are paid for the work they do.
- (to CLARIN ERIC) CLARIN should define a minimum set of metadata elements (defined semantically):
  - separately for data and for software
  - separately for faceted search and for a minimal proper description of the data or software

  Procedures and supporting software should be set up for testing compliance to these requirements, and deviations should only be allowed in exceptional cases. This is an extension of the work already started by the Austrian national consortium ((Ostojic et al., 2017). The metadata curation task force should coordinate this.
- (to profile and component developers) Use closed ('half-open') vocabularies whenever possible, but be prepared to update them regularly and to upgrade them occasionally
- (to the developers of CMDI) Enable the definition of closed vocabularies outside of a CMDI metadata element. Ensure that such vocabularies can be reused by others in multiple elements. Ensure that viewing and copying the values should be possible in the Component Registry without having to edit.
- (to CLARIN ERIC) There is a real need for a good CMDI editor, which is preferably not web-based, and enables editing of multiple files at once (both 'horizontally', i.e. all properties of one entry at a time, and 'vertically', i.e. to fill a property for a range of entries).
- (to the developers of the VLO and the CCR) It should be possible to use the 'isa' relation in the CCR to define small taxonomies of concepts, which can then be used in the faceted search to present the possible values of a facet in a hierarchical way, so that users see only a small list to select from and are only confronted with fine-grained distinctions when they are relevant to them. The CLARIN-NL Portal, CLARIN Services part[38] illustrates such hierarchical facet values. Such taxonomies will also be beneficial for profile and component editors, and for dedicated CMDI metadata editors.

Acknowledgements

The work on metadata for tools described here started already in 2012 but has been interrupted several times. Many people have worked with me on the profile and the metadata descriptions, in particular Eline Westerhout and Rogier Kraf. Eric Renckens wrote many of the descriptions on the CLARIN-NL Portal pages that formed the basis for these metadata descriptions. Daan Broeder created the faceted search in the *CLARIN in the Netherlands* Portal. I am indebted to Menzo Windhouwer and Twan Goosen for their excellent support. The developers of the software and the CLARIN Centre managers hosting the software and their metadata provided and/or corrected the information contained in the metadata descriptions.

[35] https://proycon.github.io/clam/
[36] https://javaee.github.io/wadl/
[37] https://codemeta.github.io/
[38] http://portal.clarin.nl/clarin-resource-list-fs.

References

[Broeder et al.2010] D. Broeder, M. Kemps-Snijders, D. Van Uytvanck, M. Windhouwer, P. Withers, P. Wittenburg, and C. Zinn. 2010. A data category registry- and component-based metadata framework. In N. Calzolari, B. Maegaard, J. Mariani, J. Odijk, K. Choukri, S. Piperidis, M. Rosner, and D. Tapias, editors, *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC 2010)*, pages 43–47, Valetta, Malta. European Language Resources Association (ELRA).

[Broeder et al.2012] Daan Broeder, Menzo Windhouwer, Dieter Van Uytvanck, Twan Goosen, and Thorsten Trippel. 2012. CMDI: A component metadata infrastructure. In *Proceedings of the LREC workshop 'Describing LRs with Metadata: Towards Flexibility and Interoperability in the Documentation of LR'.*, pages 1–4, Istanbul, Tyrkey. European Language Resources Association (ELRA).

[Hinrichs et al.2010] Erhard W. Hinrichs, Marie Hinrichs, and Thomas Zastrow. 2010. WebLicht: Web-Based LRT Services for German. In *Proceedings of the ACL 2010 System Demonstrations*, pages 25–29.

[Lyse et al.2015] Gunn Inger Lyse, Paul Meurer, and Koenraad De Smedt. 2015. COMEDI: A component metadata editor. In Jan Odijk, editor, *Selected Papers from the CLARIN 2014 Conference*, volume 28 of *NEALT Proceedings Series*, pages 82–98, Linköping, Sweden. Linköping Electronic Conference Proceedings. http://www.ep.liu.se/ecp/116/008/ecp15116008.pdf.

[Odijk2009] Jan Odijk. 2009. Data categories and ISOCAT: some remarks from a simple linguist. Presentation given at FLaReNet/CLARIN Standards Workshop, Helsinki, 30 September.

[Odijk2014] Jan Odijk. 2014. Discovering resources in CLARIN: Problems and suggestions for solutions. unpublished article, Utrecht University. http://dspace.library.uu.nl/handle/1874/303788, August.

[Odijk2015] Jan Odijk. 2015. Metadata curation strategy. manuscript, Utrecht, http://www.clarin.nl/sites/default/files/Metadata%20curation%20strategy%202015-06-29.pdf. Appendixes: http://www.clarin.nl/sites/default/files/Resource%20Type%20Curation%202015-6-29.xlsx and http://www.clarin.nl/sites/default/files/modality%20cleanup.xlsx, June 29.

[Odijk2018] Jan Odijk. 2018. Why I do not like web interfaces for data entry. Working paper, Utrecht University, October 11.

[Ostojic et al.2017] Davor Ostojic, Go Sugimoto, and Matej Ďurčo. 2017. The curation module and statistical analysis on VLO metadata quality. In *Selected papers from the CLARIN Annual Conference 2016, Aix-en-Provence, 26–28 October 2016*, number 136 in Linköping Electronic Conference Proceedings, pages 90–101. Linköping University Electronic Press, Linköpings Universitet.

[Schuurman et al.2016] Ineke Schuurman, Menzo Windhouwer, Oddrun Ohren, and Daniel Zeman. 2016. CLARIN Concept Registry: The New Semantic Registry. In Koenraad De Smedt, editor, *Selected Papers from the CLARIN Annual Conference 2015, October 14–16, 2015, Wroclaw, Poland*, number 123 in Linköping Electronic Conference Proceedings, pages 62–70, Linköping, Sweden. CLARIN, Linköping University Electronic Press. http://www.ep.liu.se/ecp/article.asp?issue=123&article=004.

[van den Bosch et al.2007] A. van den Bosch, G.J. Busser, W. Daelemans, and S. Canisius. 2007. An efficient memory-based morphosyntactic tagger and parser for Dutch. In F. Van Eynde, P. Dirix, I. Schuurman, and V. Vandeghinste, editors, *Selected Papers of the 17th Computational Linguistics in the Netherlands Meeting*, pages 99–114. Leuven, Belgium.

[Van Uytvanck2014] Dieter Van Uytvanck. 2014. How can I find resources using CLARIN? Presentation held at the *Using CLARIN for Digital Research* tutorial workshop at the *2014 Digital Humanities Conference*, Lausanne, Switzerland. https://www.clarin.eu/sites/default/files/CLARIN-dvu-dh2014_VLO.pdf, July.

[Westerhout and Odijk2013] Eline Westerhout and Jan Odijk. 2013. Metadata for tools: creating a CMDI profile for tools. Presentation held at CLIN 2013, Enschede, the Netherlands. http://www.clarin.nl/sites/default/files/13CLIN.pdf, 18January.

[Zeeman and Windhouwer2018] Rob Zeeman and Menzo Windhouwer. 2018. Tweak your CMDI forms to the max. Presentation at the CLARIN Annual Conference, Pisa, Italy. https://www.clarin.eu/sites/default/files/CLARIN2018_Session-4-5_Paper-22_Zeeman-Windhouwer.pdf, October10.

[Zinn2016a] Claus Zinn. 2016a. The CLARIN language resource switchboard. https://www.clarin.eu/sites/default/files/08%20-%20ZINN-Lg-Sw-Board.pdf. Presentation at the CLARIN 2016 Annual Conference.

[Zinn2016b] Claus Zinn. 2016b. The CLARIN language resource switchboard. https://www.clarin.eu/sites/default/files/zinn-CLARIN2016_paper_26.pdf. Abstract for the CLARIN 2016 Annual Conference.

[Zinn2017] Claus Zinn. 2017. A bridge from EUDAT's B2DROP cloud service to CLARIN's language resource switchboard. https://www.clarin.eu/sites/default/files/Zinn-CLARIN2017_paper_17.pdf. Abstract for the CLARIN 2017 Annual Conference.