Component Metadata Infrastructure Best Practices for CLARIN

Thomas Eckart
Leipzig University
teckart@informatik.uni-leipzig.de

Twan Goosen
CLARIN ERIC
twan@clarin.eu

Susanne Haaf
BBAW
haaf@bbaw.de

Hanna Hedeland
Hamburg University
hanna.hedeland@uni-hamburg.de

Oddrun Ohren
National Library of Norway
oddrun.ohren@nb.no

Dieter Van Uytvanck
CLARIN ERIC
dieter@clarin.eu

Menzo Windhouwer
CLARIN ERIC/Meertens Institute
menzo.windhouwer@meertens.knaw.nl

1 Introduction

Last year, 2016, saw the release of both version 1.2 of the Component Metadata (CMD) Infrastructure (CMDI) [CLARIN ERIC 2017] and a first complete technical specification [CMDI Task Force 2016]. This new version provides new possibilities, which are gradually opened up by the ecosystem of tools and registries in CMDI. One of the key properties of CMDI is its flexibility, which makes it possible to create metadata records closely tailored to the requirements of resources and tools/services. However, design and implementation choices made at various levels in the CMD lifecycle might influence how well or easily a CMD record is processed and its associated resources made available in the CLARIN infrastructure. Knowledge on this has traditionally been scattered around in various documents, web pages and even completely hidden from sight in experts' minds. To make this knowledge explicit the CMDI and Metadata Curation Task Forces have teamed up to create a Best Practices guide. This guide, together with the technical CMDI 1.2 specification, will be a valuable knowledge base and will help any (technical) CMDI user to bring her CMD records to their full potential use within CLARIN. In May 2017, the writing of this guide is still an ongoing effort and this paper gives a first overview and insight in its contents. Notice, this paper is not a replacement for the guide itself, which should be finished in 2017, but aims to draw attention to it and foster discussion and feedback. The next sections give a description of the scope, an outline and a glimpse into the content of the guide.

2 Scope

The target audience for the Best Practices guide include 1) CMD modellers, who select or create components and profiles to describe certain language resources, 2) developers who create metadata convertors, forms or editors to create records that comply with these profiles, and a, hopefully small, group of 3) CMD creators who create records by hand. In contrast researchers, while being the ultimate end-users of the CLARIN infrastructure, do not belong to the primary target audience of this guide. Researchers are, in general, not directly exposed to CMDI, but if so should be fully guided by an application (with its own user guide) and shielded from the technical details. Furthermore, using this Best Practices guide will require a basic understanding of CMDI - it is not intended as a tutorial.

The CLARIN infrastructure is able to deal with any valid CMD record that is based on a profile from its Component Registry. The technical specification [CMDI Task Force 2016] acts as a baseline here to specify valid CMD profiles, components and records. The scope of the best practices goes beyond those purely schematic constraints and ranges from modelling guidelines, e.g. prefer elements over attributes, to rather low-level technical guidelines, e.g. use the UTF-8 encoding for your CMD records. As a consequence, the CLARIN infrastructure will be able to extract information more efficiently from CMD records that meet these best practices and, hence, provide the end users improved access to those records, the context from which they originate and the resources they provide.

3 Outline

The first two sections of the CMDI Best Practices guide follow the lifecycle of Component Metadata: modelling CMD and authoring CMD records. Best practices are given for various CMD constructs encountered at the different levels. Both these sections also provide guidelines regarding the workflow, e.g. (order of) actions to take and tools to use, and indicators of potential problems (also known as ‘smells’). Some approaches and problems that can be considered “crosscutting concerns”, having aspects that are not limited to either modelling or authoring, are covered in their own section. The next sections provide insights in the current content.

4 CMDI Best Practices for CLARIN

4.1 Modelling Component Metadata

CMD modelling takes place on two similar, but distinct levels: the CMD profile and the CMD component level. The former is the higher of the two levels and represents the content scope of a full CMD record. Profiles are composed out of CMD components, elements and attributes. CMD components, which are composed out of (other) CMD components, elements and attributes, represent aspects of metadata descriptions and serve as building blocks. This distinguishes components from profiles, which only serve as blueprints for metadata records. This gives rise to two sets of modelling best practices, in addition to a general modelling principle that we will cover first. This modelling principle relates to components, profiles and concepts, advising to make these “as generic as possible and as specific as needed”. Rather than a concrete guideline, this best practice can be considered a ‘commandment’ from which many best practices follow. It reflects the principle of reuse, around which CMD has been designed, and the requirement for metadata modellers to always consider the potential of reuse even when modelling for a specific project or domain. This applies to, for example, naming of elements, scope of components, and the cardinality of attributes.

4.1.1 Component modelling best practices

Detailed documentation of the right type should be provided in the right place. Here, descriptive documentation should be distinguished from semantic identification (which is best done by means of a concept link), and operational instructions. Best practices related to naming serve to promote clarity and consistency. According to these, components, element and attribute names should be in English, verbose (avoiding abbreviations and acronyms) and not unnecessarily specific. Modellers should aim for a uniform naming ‘pattern’ (e.g. usage of upper and lower case) across their components, although consistency may be compromised by reusing existing components. Content validity and quality strongly depends on defining the right constraints and value schemes. The modeller should discourage usage of empty string values but rather demand values of a specific type where possible, selected from a vocabulary if feasible, even if a field's value scheme is of binary nature. Elements are generally more suitable than attributes, and the latter should only be used to model value annotation. In practice, reusability is often hampered by over-specific modelling. The Best Practice guide provides suggestions for modelling components with broad reusability in mind. It urges modellers to reuse existing components wherever possible, or as a fallback solution attempt to ‘recycle’, i.e. base new components on existing ones.

4.1.2 Profile modelling best practices

Since modelling at profile level methodologically overlaps with modelling at component level, all component best practices also apply to profile modelling as far as documentation, naming, constraints, value schemes and the use of vocabularies goes. Following the general reuse principle, profile creators are advised to compose their profiles out of existing components as much as possible. Each of a profile's components should deal with a distinct aspect of a resource, avoiding semantic overlaps between different parts of a profile. The modeller must ensure that metadata can be created for broad user groups, which implies modelling for verbose, self-contained resource descriptions. Providing proper documentation is at least as important for profiles as it is for components as metadata creators will primarily be exposed to CMD definitions at the profile level.

4.1.3 Problem Indicators (‘Smells’)

So-called ‘smells’ are indicators of bad or inefficient design of components or profiles and are a concept originating from design analysis in computer programming. Discovering such an indicator does not necessarily mean that a design is broken, but, instead, that there are warning signs that should be checked. An analysis of the public CMD components and profiles in the CLARIN Component Registry by the authors revealed several problem indicators.

As an example a “Standalone Profile” re-uses hardly any components, which may be a hint for insufficient evaluation of the existing component stock in the modelling phase. “Speculative Generality” is seen as a design flaw when a component or profile allows describing aspects of a resource that most likely will never occur in any real-world CMD record. In those cases a reduction of size and complexity may lead to simpler designs and improved intelligibility for users. Another kind of “Speculative Generality” holds for profiles that are modelled to describe different types of resources, leading to disjunct instantiation patterns in CMD records. In those cases it may be useful to extract resource type independent aspects into a single component and create a profile for every resource type while reusing the common parts.

4.2 Authoring Component Metadata Records

A CMD record consists of two main sections: 1) an envelope, which contains (technical) metadata on the record itself and on the resources it describes, and 2) the component section, which should be a valid instance of a CMD profile.

4.2.1 Authoring the envelope

An important best practice for the envelope is to include reference to at least one resource. Otherwise, it remains unclear which language resources are described by the descriptive metadata in the record. The exact type of these resources should also be made explicit, so that generic tools, for example the Language Resource Switchboard [Zinn 2016], can provide access to these resources. Some resources consist of distinct parts, e.g. items in a collection or annotations and annotated text in separate files. Such decomposition should be reflected in the CMD file by listing each part as individual resource proxies. Moreover, if the listed resources are related to each other in significant ways, those relationships should be represented as instances of ResourceRelation, each of which referencing the two related resources as well as a concept describing its meaning. The technical metadata of a CMD record should also contain the identifier of the CMD profile used.

4.2.2 Authoring the component section

The component section must be a valid instance of this profile, i.e. it should contain at least instances of all mandatory components, elements, attributes and values. In general, it should be aimed at providing ‘complete’ metadata information with regard to the selected profile, i.e. the information asked for by the profile’s components, elements etc. should be provided as exhaustively as possible. This applies especially to those elements which relate to VLO facets, as those have great impact on the visibility of the record and its resources within the CLARIN infrastructure. In addition, components dealing with the sources of a resource (e.g. for a digitized text a physical book in a library or a printed edition) and responsibilities should be attended to. The information provided should be specific rather than generic (e.g. “modality: gesture, speech” is preferred to “modality: multimodal”). Values of metadata elements that relate to VLO facets should be kept short (one-word expressions), should use standardized wording and spelling (e.g. in case of named entities such as organizations, collections, etc.) and should connect to the vocabulary already used in the VLO facets wherever possible.

4.2.3 Problem Indicators ('Smells')

'Smells' can also be identified for metadata records. This contains indicators on a variety of levels. As an example an extensive record size (several megabytes or more) may indicate an inappropriate granularity, where, instead of a single resource, a complete resource collection is described. In these cases it may be helpful to restructure the outsized file to a collection record referring to several individual records, each describing a single resource. Another aspect indicating 'fitness' of a record is the instantiation of highly relevant elements or the used level of detail: a record can only be accurately represented in central applications - like the VLO - if elements containing the precise name or a helpful description of the resource are instantiated. The complete guideline contains many more of those hints derived from identified shortcomings in real-life CMD records.

4.3 Common Approaches and Problems

The final version of the guidelines will contain sections on the following topics:

1. multilingual metadata;
2. hierarchies and granularity;
3. licensing;
4. dealing with uncertainty;
5. modelling for metadata conversion.

Next to these topics also some common use cases will be discussed, e.g. how to create a CMD profile and/or records for a common resource type, e.g. WordNets.

5 Conclusions and Future Work

The CMDI Best Practices guide is a, currently ongoing, joined effort by the CMDI and Metadata Curation Task Forces, but the best practices shortly outlined in this paper show already how this guide will help both CMD modellers and authors to make design decisions that will enable the CLARIN infrastructure to give end users optimal access to the CMD records and thus the associated linguistic resources provided by the CLARIN centres.

References

[CLARIN ERIC 2017] CLARIN ERIC. 2017. *CMDI 1.2*, https://www.clarin.eu/cmdi1.2 Accessed on April 18, 2017.

[CMDI Task Force 2016] CMDI Task Force. 2016. *CMDI 1.2 specification*. CE-2016-0880, CLARIN ERIC, Utrecht, The Netherlands.

[Zinn 2016] Claus Zinn. 2016. *LR Switchboard (software)*. CE-2016-0881, CLARIN ERIC, Utrecht, The Netherlands.