From CLARIN Component Metadata to Linked Open Data

Matej Ďurčo, Menzo Windhouwer

Institute for Corpus Linguistics and Text Technology (ICLTT), The Language Archive - DANS
Vienna, Austria, The Hague, The Netherlands
matej.durco@oeaw.ac.at, menzo.windhouwer@dans.knaw.nl

Abstract

In the European CLARIN infrastructure a growing number of resources are described with Component Metadata. In this paper we describe a transformation to make this metadata available as linked data. After this first step it becomes possible to connect the CLARIN Component Metadata with other valuable knowledge sources in the Linked Data Cloud.

Keywords: Linked Open Data, RDF, component metadata

1. Motivation

Although semantic interoperability has been one of the main motivations for CLARIN's Component Metadata Infrastructure (CMDI) (Broeder et al., 2010),[1] until now there has been no work on the obvious – bringing CMDI to the Semantic Web. We believe that providing the CLARIN CMD records as Linked Open Data (LOD) interlinked with external semantic resources, will open up new dimensions of processing and exploring of the CMD data by employing the power of semantic technologies. In this paper, we lay out how individual parts of the CMD data domain can be expressed in RDF and made ready to be interlinked with existing external semantic resources (ontologies, taxonomies, knowledge bases, vocabularies).

2. The Component Metadata Infrastructure

The basic building blocks of CMDI are components. Components are used to group elements and attributes, which can take values, and also other components (see Figure 1). Components are stored in the Component Registry (CR), where they can be reused by other modellers. Thus a metadata modeller selects or creates components and combines them into a profile targeted at a specific resource type, a collection of resources or a project, tool or service. A profile serves as blueprint for a schema for metadata records. CLARIN centres offer these CMD records describing their resources to the joint metadata domain. There are a number of generic tools which operate on all the CMD records in this domain, e.g., the Virtual Language Observatory.[2] These tools have to deal with the variety of CMD profiles. They can do so by operating on a semantic level, as components, elements and values can all be annotated with links to concepts in various registries. Currently used concept registries are the Dublin Core metadata terms and the ISOcat Data Category Registry. These concept links allow profiles, while being diverse in structure, to share semantics. Generic tools can use this semantic linkage to overcome differences in terminology and also in structure.

2.1. Current status of the joint CMD Domain

To provide a frame of reference for the proportions of the undertaking, this section gives a few numbers about the data in the CMD domain.

2.1.1. CMD Profiles

Currently[3] 153 public profiles and 859 components are defined in the CR. Next to the 'native' ones a number of profiles have been created that implement existing metadata formats, like OLAC/DCMI-terms, TEI Header or the META-SHARE schema. The individual profiles also differ very much in their structure – next to simple flat profiles there are complex ones with up to 10 levels and a few hundred elements.

2.1.2. Instance Data

The main CLARIN OAI-PMH harvester[4] regularly collects records from the – currently 56 – providers, all in all over 600.000 records. Some 20 of the providers offer CMDI records, the rest provides around 44.000 OLAC/DC records, that are converted into the corresponding CMD profile. Some of the providers of 'native' CMD records expose multiple profiles (e.g. Meertens Institute uses 12 different ones), so that overall instance data for more than 60 profiles is present.

3. CMD to RDF

In the following a RDF encoding is proposed for all levels of the CMD data domain:

- CMD meta model (see Figure 1),
- profile and component definitions,
- administrative and structural information of CMD records and
- individual values in the fields of the CMD records.

[1] http://www.clarin.eu/cmdi/
[2] http://www.clarin.eu/vlo/
[3] All numbers are as of 2014-03.
[4] http://catalog.clarin.eu/oai-harvester/

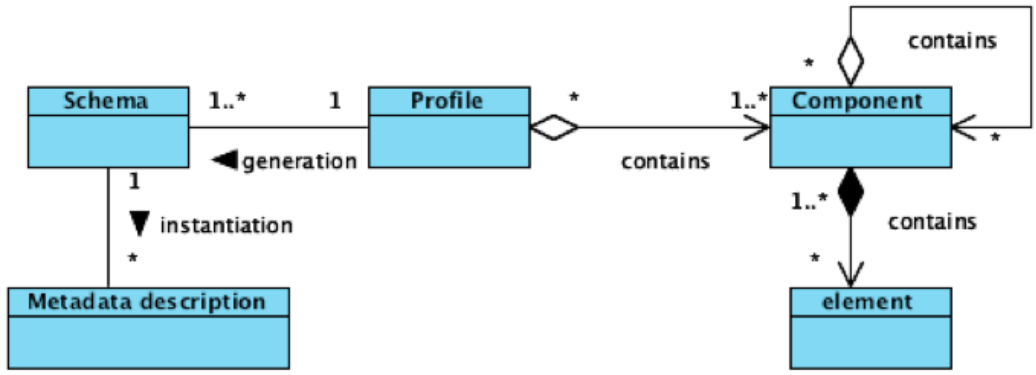

Figure 1: Component Metadata Model (ISO/DIS 24622-1, 2013)

3.1. CMD specification

The main entity of the meta model is the CMD component modelled as a `rdfs:Class` (see Figure 2). A CMD profile is basically a CMD component with some extra features, implying a specialization relation. It may seem natural to translate a CMD element to a RDF property (as it holds the literal value), but given its complexity, i.e., attributes,[5] it too has to be expressed as a `rdfs:Class`. The actual literal value is a property of given element of type `cmdm:hasElementValue`. For values that can be mapped to entities defined in external semantic resources, the references to these entities are expressed in parallel object properties of type `cmdm:hasElementEntity` (constituting outbound links). The containment relation between components and elements is expressed with a dedicated property `cmdm:contains`.

3.2. CMD profile and component definitions

These top-level classes and properties are subsequently used for modelling the actual profiles, components and elements as they are defined in the CR. For stand-alone components, the IRI is the (future) path into the CR to get the RDFS representation for the profile/component.[6] For "inner" components (that are defined as part of another component) and elements the identifier is a concatenation of the nearest ancestor stand-alone component's IRI and the dot-path to given component/element (e.g., Actor: `cr:clarin.eu:cr1:c_1271859438197/rdf#Actor.Actor_Languages.Actor_Language`[7])

```
cmd:collection
  a                 cmdm:Profile ;
  rdfs:label        "collection" ;
  dc:identifier     cr:clarin.eu:cr1:p_1345561703620 .
cmd:Actor
  a                 cmdm:Component .
```

[5] Although the modelling work has been done, due to space considerations, we will not further discuss attributes.

[6] `http://catalog.clarin.eu/ds/ComponentRegistry/rest/registry/components/clarin.eu:cr1:c_1271859438125/rdf`

[7] For better readability, in the examples we collapse the component IRIs, using just the name, prefixed with `cmd:`

3.2.1. Data Categories

The primary concept registry used by CMDI is ISOcat. The recommended approach to link to the data categories is via an annotation property (Windhouwer and Wright, 2012).

```
dcr:datcat
  a                 owl:AnnotationProperty ;
  rdfs:label        "data category"@en .
```

Consequently, the `@ConceptLink` attribute on CMD elements and components referencing the data category can be modelled as:

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

Lateron, this information can be used, e.g., in combination with ontological relationships for these data categories available in the RELcat Relation Registry (Windhouwer, 2012), to map to other vocabularies.

3.3. CMD instances

In the next step, we want to express in RDF the individual CMD instances, the metadata records.
We provide a generic top level class for all resources (including metadata records), the `cmdm:Resource` class and the `cmdm:hasMimeType` predicate to type the resources.

```
<lr1>
  a                         cmdm:Resource ;
  cmdm:hasMimeType          "audio/wav" .
```

3.3.1. Resource Identifier

The PID of a Language Resource (`<lr1>`) is used as the IRI for the described resource in the RDF representation. The relationship between the resource and the metadata record can be expressed as an annotation using the OpenAnnotation vocabulary.[8] (Note, that one MD record

[8] `http://openannotation.org/spec/core/core.html`

```
@prefix cmdm: <http://www.clarin.eu/cmd/general.rdf#>.

# basic building blocks of CMD Model
cmdm:Component          a                   rdfs:Class .
cmdm:Profile            rdfs:subClassOf     cmdm:Component .
cmdm:Element            a                   rdfs:Class .

# basic CMD nesting
cmdm:contains           a                   rdf:Property ;
                        rdfs:domain         cmdm:Component ;
                        rdfs:range          cmdm:Component , cmdm:Element .
# values
cmdm:Value              a                   rdfs:Literal .

cmdm:hasElementValue    a                   rdf:Property ;
                        rdfs:domain         cmdm:Element ;
                        rdfs:range          cmdm:Value .

# add a parallel separate class/property for the resolved entities
cmdm:Entity             a                   rdfs:Class .

cmdm:hasElementEntity   a                   rdf:Property ;
                        rdfs:domain         cmdm:Element ;
                        rdfs:range          cmdm:Entity .
```

Figure 2: The CMD meta model in RDF

can describe multiple resources. This can be also easily accommodated in OpenAnnotation.)

```
_:anno1
 a                  oa:Annotation ;
 oa:hasTarget       <lr1a >, <lr1b>;
 oa:hasBody         _:topComponent1 ;
 oa:motivatedBy     oa:describing .
```

3.3.2. Provenance

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

```
<lr1.cmd >
 dc:creator         "John Doe" ;
 dc:publisher       <http://clarin.eu>;
 dc:created         "2014-02-05"^^xs:date .
```

3.3.3. Collection hierarchy

In CMD, there are dedicated generic elements – the `cmd:ResourceProxyList` structure – used to express both the collection hierarchy and to point to resource(s) described by the CMD record. The collection hierarchy can be modelled as an OAI-ORE Aggregation.[9] (The links to resources are handled by `oa:hasTarget`.) :

```
<lr0.cmd >      a               ore:ResourceMap .
<lr0.cmd>       ore:describes   _:agg0 .
_:agg0          a               ore:Aggregation ;
                ore:aggregates  <lr1.cmd>, <lr2.cmd>.
```

3.3.4. Components – nested structures

For expressing the tree structure of the CMD records, i.e., the containment relation between the components, a dedicated property `cmd:contains` is used:

```
_:actor1        a               cmd:Actor .
_:actor1lang1   a               cmd:Actor.Language .
_:actor1        cmd:contains    _:actor1lang1 .
```

3.3.5. Elements, Fields, Values

Finally, we want to integrate also the actual values in the CMD records into the linked data. As explained before, CMD elements have to be typed as `rdfs:Class`, the actual value expressed as `cmdm:ElementValue`, and they are related by a `cmdm:hasElementValue` property.

While generating triples with literal values seems straightforward, the more challenging but also more valuable aspect is to generate object property triples (predicate `cmdm:hasElementEntity`) with the literal values mapped to semantic entities. The example in Figure 3 shows the whole chain of statements from metamodel to literal value and corresponding semantic entity.

[9] `http://www.openarchives.org/ore/1.0/primer#Foundations`

```
cmd:Person                          a                                   cmdm:Component .
cmd:Person.Organisation             a                                   cmdm:Element .
cmd:hasPerson.OrganisationElementValue
                                    rdfs:subPropertyOf                  cmdm:hasElementValue ;
                                    rdfs:domain                         cmd:Person.Organisation ;
                                    rdfs:range                          xs:string .
cmd:hasPerson.OrganisationElementEntity
                                    rdfs:subPropertyOf                  cmdm:hasElementEntity ;
                                    rdfs:domain                         cmd:Person.Organisation ;
                                    rdfs:range                          cmd:Person.OrganisationElementEntity .
cmd:Person.OrganisationElementEntity
                                    a                                   cmdm:Entity .

  # person (mentioned in a MD record) has an affiliation (cmd:Person/cmd:Organisation)
_:pers                              a                                   cmd:Person ;
                                    cmdm:contains                       _:org .
_:org                               a                                   cmd:Person.Organisation ;
                                    cmd:hasPerson.OrganisationElementValue   'MPI'^^xs:string ;
                                    cmd:hasPerson.OrganisationElementEntity  <http://www.mpi.nl/>.
<http://www.mpi.nl/>                a                                   cmd:OrganisationElementEntity .
```

Figure 3: Chain of statements from metamodel to literal value and corresponding semantic entity

4. Implementation

The transformation of profiles and instances into RDF/XML is accomplished by a set of XSL-stylesheets. In the future, when the mapping has been tested extensively, they will be integrated into the CMD core infrastructure, e.g., the CR. A linked data representation of the CLARIN joint metadata domain can then be stored in a RDF triple store and exposed via a SPARQL endpoint. Currently, a prototype interface is available for testing as part of the Metadata repository developed at CLARIN Centre Vienna[10].

5. CMDI's future in the LOD Cloud

The main added value of LOD (Berners-Lee, 2006) is the interconnecting of disparate datasets in the so called LOD cloud (Cyganiak and Jentzsch, 2010).

The actual mapping process from CMDI values (see Section 3.3.5.) to entities is a complex and challenging task. The main idea is to find entities in selected reference datasets (controlled vocabularies, ontologies) corresponding to the literal values in the metadata records. The obtained entity identifiers are further used to generate new RDF triples, representing outbound links. Within CMDI the SKOS-based vocabulary service CLAVAS,[11] which will be supported in the upcoming new version of CMDI, can be used as a starting point, e.g., for organisations. In the broader context of LOD Cloud there is the Open Knowledge Foundations Working Group on Linked Data in Linguistics, that represents an obvious pool of candidate datasets to link the CMD data with.[12] Within these lexvo seems a most promising starting point, as it features URIs like `http://lexvo.org/id/term/eng/`, i.e., based on the ISO-639-3 language identifiers which are also used in CMD records. lexvo also seems suitable as it is already linked with a number of other LOD linguistic datasets like WALS, lingvoj and Glottolog. Of course, language is just one dimension to use for mapping. Step by step we will link other categories like countries, geographica, organisations, etc. to some of the central nodes of the LOD cloud, like dbpedia, Yago or geonames, but also to domain-specific semantic resource like the ontology for language technology LT-World (Jörg et al., 2010) developed at DFKI.

Next to entities also predicates can be shared across datasets. The CMD Infrastructure already provides facilities in the form of ISOcat and RELcat. RELcat, for example, has already sets to relate data categories to Dublin Core terms. This can be extended with the ontology for metadata concepts described in (Zinn et al., 2012), which does not provide common predicates but would allow to do more generic or specific searches.

6. Conclusions

In this paper, we sketched the work on encoding of the whole of the CMD data domain in RDF, with special focus on the core model – the general component schema. In the future we will extend this with mapping element values to semantic entities.

With this new enhanced dataset, the groundwork is laid for a full-blown *semantic search*, i.e., the possibility of exploring the dataset indirectly using external semantic resources (like vocabularies of organizations or taxonomies of resource types) to which the CMD data will then be linked.

7. References

Tim Berners-Lee. 2006. Linked data. online: http://www.w3.org/DesignIssues/LinkedData.html.

[10] `http://clarin.oeaw.ac.at/mdrepo/cmd?operation=cmd2rdf`

[11] `https://openskos.meertens.knaw.nl/`

[12] `http://linguistics.okfn.org/resources/llod/`

Daan Broeder, Marc Kemps-Snijders, et al. 2010. A data category registry- and component-based metadata framework. In Nicoletta Calzolari, Khalid Choukri, et al., editors, *LREC*, Valletta, May. ELRA.

Richard Cyganiak and Anja Jentzsch. 2010. The linking open data cloud diagram. online: http://lod-cloud.net/.

ISO/DIS 24622-1. 2013. Language resource management – component metadata infrastructure – part 1: The component metadata model (cmdi-1).

Brigitte Jörg, Hans Uszkoreit, and Alastair Burt. 2010. LT World: Ontology and reference information portal. In Nicoletta Calzolari and Khalid Choukri et al., editors, *LREC*, Valletta, Malta, May. ELRA.

Menzo Windhouwer and Sue Ellen Wright. 2012. Linking to linguistic data categories in ISOcat. In *Linked Data in Linguistics*, pages 99–107. Springer.

Menzo Windhouwer. 2012. RELcat: a relation registry for ISOcat data categories. In Nicoletta Calzolari, Khalid Choukri, et al., editors, *LREC*, Istanbul, May. ELRA.

Claus Zinn, Christina Hoppermann, and Thorsten Trippel. 2012. The isocat registry reloaded. In Elena Simperl, Philipp Cimiano, Axel Polleres, Oscar Corcho, and Valentina Presutti, editors, *The Semantic Web: Research and Applications*, volume 7295 of *Lecture Notes in Computer Science*, pages 285–299. Springer Berlin Heidelberg.