An API for accessing the Data Category Registry

Marc Kemps-Snijders, Julien Ducret, Laurent Romary, Peter Wittenburg

An API for accessing the Data Category Registry

Marc Kemps-Snijders, Julien Ducret, Laurent Romary, Peter Wittenburg

MPI for Psycholinguistic, LORIA
Wundtlaan 1, 6525 XD Nijmegen, The Netherlands
{marc.kemps-snijders, peter.wittenburg}@mpi.nl, {julien.ducret, laurent.romary}@loria.fr

Abstract

Central Ontologies are increasingly important to manage interoperability between different types of language resources. This was the reason for ISO to set up a new committee ISO TC37/SC4 taking care of language resource management issues. Central to the work of this committee is the definition of a framework for a central registry of data categories that are important in the domain of language resources. This paper describes an application programming interface that was designed to request services from this data category registry. The DCR is operational and the described API has already been tested from a lexicon application.

1. Introduction

At the MPI for Psycholinguistics and many other places lexical resources are not only characterized by a large variety in structures and formats but also by a various lexical encodings to identify linguistic concepts. These can be explained by differences in language, theory of the researcher and tools used in the creation process. To allow operations across lexical resources, e.g. integration of lexica with other corpora, relations that exist between the linguistic concept encodings must be drawn. With a large number of resources this may become a lengthy and labor intensive process. For a number of linguistic domains well established linguistic concept registries are becoming available [1,2] which can greatly enhance interoperability between lexical resources. For this to have a real impact on the creation process of lexical resources it is essential that common application programming interfaces become available through which tool builders can connect to these registries and integrate usage of these standard concepts into their tools.

2. ISO 12620 Data Category Registry

The Data Category Registry currently being established within ISO TC37/SC4 [3,4] has the potential to revolutionize the way in which we will use linguistic concepts and achieve interoperability at the level of linguistic encoding. The ISO DCR basically consists of a flat list of linguistic concepts covering a range of linguistic domains. It is organized in profiles covering concepts such as metadata, morphosyntax, syntax etc. Each concept consists of a proper definition and may have a conceptual domain. Simple concepts that occur as values in the conceptual domain are atomic and don't have a value range of their own. Each concept has language sections that specify the value ranges in the different languages. Where applicable the definition of concepts can refer to a broader generic concept if this helps clarification. New concepts that a linguist may find essential may be added to the DCR. In general, such new concepts will first be part of a private workspace, but they can be subject of processes that may end up in an acceptance as part of the official DCR.

The DCR is available as an XML structure that is defined by a RelaxNG schema [5]. Definitions of linguistic concepts are widely agreed in the community and are available in machine readable form. This achieves interoperability with tools that can interact with it. They can make use of existing concepts and incorporate them in schema definitions. Schemas that define language resources such as metadata descriptions, annotations, lexica etc should include references that point to concepts in a unique way. When for example lexical attributes in two different lexica point to the same unique ID in the DCR we know that they share the same concept, i.e. search engines could easily extend their searches on these lexica.

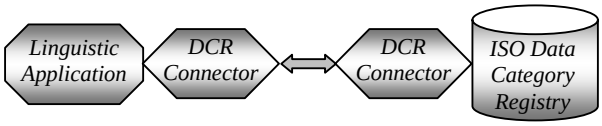

Figure 1 indicates the interaction between the DCR and a linguistic application such as LEXUS.

To establish a common entry point for tools interacting with the DCR it is crucial to define, publish and maintain a stable application programming interface (API) to the DCR so that tool builders can easily develop interaction schemes with it.

The following diagram may help to provide an overview of the domain where the API described in this document may be used. We assume that a researcher wants to create either manually or automatically a Part-Of-Speech annotation. We can further assume that he/she wants to build on existing linguistic knowledge and to integrate the emerging resource into the interoperable domain. Working with an annotation tool he will want to browse or search for an appropriate concept to encode Part-Of-Speech in the ISO DCR. Once found he/she will want to integrate all useful information into the schema and make a reference to the DCR entry. The information to be included could be the name of the concept to be included as the tier name, the definition of the concept to allow a quick look-up for other people and the value range (conceptual domain) to be included in menus to constrain the persons or algorithms carrying out the actual encoding. Having done so search engines or other type of linguistic tools could make use of the extracted information or contact the ISO DCR to request even more information.

The programming interface must support three general access points for accessing linguistic concepts. The first access point is by means of browsing the DCR. Tools guiding the user must be able to access the DCR and provide means for browsing profiles and linguistic concepts. The other access point is to provide direct search capabilities. Search capabilities must be provided over several fields of interest supported by the DCR model. Finally an access point must be created to allow direct access of the linguistic concept in the DCR.

3. Browsing the DCR

3.1. Profile browsing

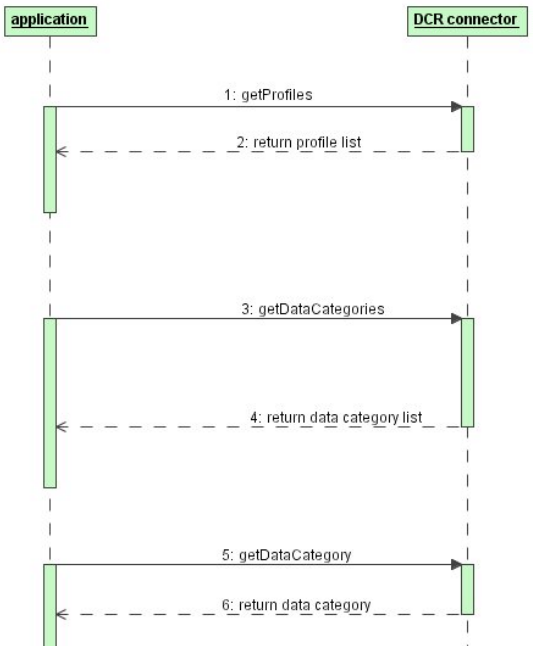

Figure 2: Profiles browsing interaction diagram.

Tools guiding the user through the selection process of a linguistic concept from the DCR must be able guide the user in the selection process of a linguistic concept. Rather than presenting the user with a list of all data categories present in the DCR the user selects a profile applicable to the linguistic domain they are working in. Then all linguistic concepts for the selected profile may be presented. The user may then proceed to view the details of the data category.

Browsing the DCR in this manner consists therefore of the following steps:

- A list of profiles is requested from the DCR connector.
- A list of data categories is requested from the DCR connector for a specific profile. The list of data categories may be returned in an abbreviated form.
- The details for a specific data category are requested from the DCR. All information of the data category is returned.

The sequence diagram above illustrates the interaction process.

3.1.1. ConceptGeneric Browsing

An alternative way of traversing the DCR is by means of the ConceptGeneric relations. These describe 'is_a' relations which may exist between linguistic concepts. An example of this is pronoun 'is a' noun. Two approaches are possible here, either top-down or bottom-up. In top-down browsing, all concepts are requested for which a given concept is a broader generic concept, in bottom-up browsing the concepts are requested for which the given concept is a narrower concept.

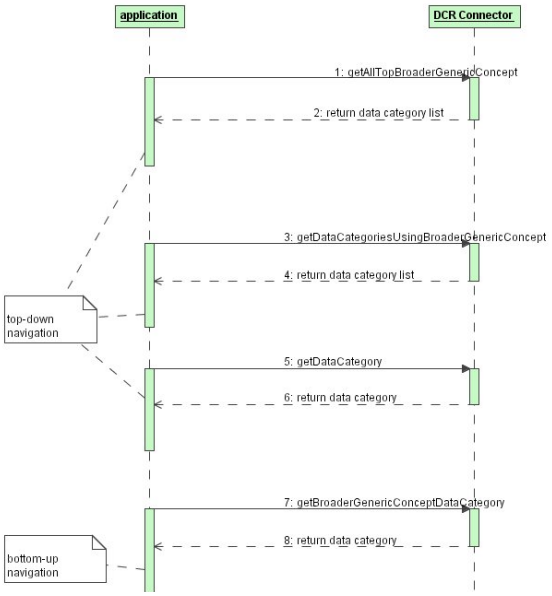

Figure 3: ConceptGeneric browsing interaction diagram.

Top-Down navigation consists of the following steps:

1. A list of all data categories is requested from the DCR which do not have a more generic concept. A list of data categories is returned.
2. The user selects a data category from the list and requests all data category for which the selected data category is a more generic concept. A list of data categories is returned. The list may be empty if no data categories are present.
3. The user selects a data category from the list to view its details

Bottom-up navigation is based on the assumption that a user has preselected a data category and is interested in the more generic data category. Bottom-up navigation consists of only a single step. Get the generic concept for this data category. The DCR will return all data category details or none if no generic concept is defined for the selected data category.

4. Searching the DCR

Tools assisting the user in the selection process must also have the possibility to search the DCR. The search must be restricted to a list of keywords to search for and a

list of fields in the DCR model to search through. Again a list of linguistic concepts should be retrieved from which a user may select a data category of interest.

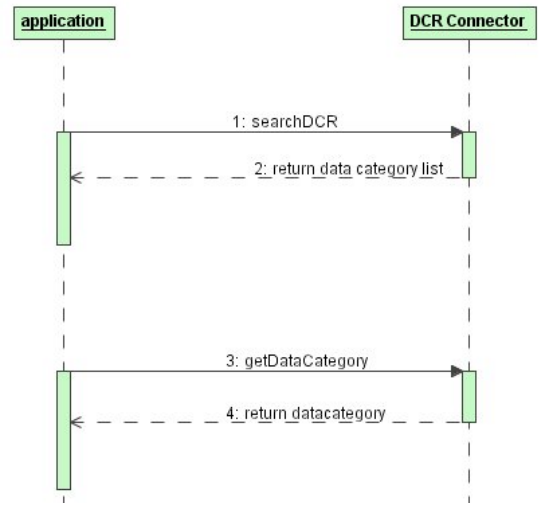

Figure 4: Search interaction diagram

The search process comprises of the following steps:

1. search the DCR using specified keywords and search parameters. The system will return a list of all data categories which mach the specified search criteria.

2. View details regarding one of the data categories from the presented result. List. The system will return all detailed information regarding the requested data category.

The sequence diagram above outlines this process.

5. Direct selection

When a data category is directly selected all details regarding that data category must be immediately accessible. This results in a single call (getDataCategory) to retrieve the details of a data category.

6. The DCR Application Programming Interface

From the interaction patterns described above the application programming interface may be derived. Development of the API is foreseen in 2 steps. In the first phase Profile browsing, search and direct access are implemented. Calls are implemented using simple HTTP requests to the DCR server. The DCR server returns an XML representation of the requested information. The user application can choose to use this XML directly or to convert it to a DCR API implementation. The latter approach is implemented as part of the interface module and a conversion library is available. The interface module also handles the connection to the DCR server. In the second phase, ConceptGeneric browsing is added to the API and the interface is made accessible as a web service. The DCR service returns an XML representation of the DCR objects. Again, the XML may be used directly or the conversion library may be used as part of the interface module.

The first phase is currently completed. All calls to the interface are stateless, meaning that all information needed to understand the call are present in the call's parameters. The API is currently integrated in LEXUS [6], a web based lexicon tool. Users are able to browse, search and select linguistic concepts form the DCR which are subsequently incorporated in LMF lexical models [7].

The following calls are supported. They are described in pseudo code for clarification purposes.

6.1. getProfiles()

Returns a list of profiles currently maintained by the DCR. The call takes no parameters and the result is returned in the RelaxNG format described below:

```
<grammar
xmlns="http://relaxng.org/ns/structure/1.0">
<start>
    <ref name="struct_listOfProfiles" />
    </start>
<define name="struct_listOfProfiles">
    <element name="struct">
        <attribute name="type">
            <value>ListofProfiles</value>
        </attribute>
        <zeroOrMore>
          <element name="feat">
            <attribute name="type">
                <value>profile</value>
            </attribute>
            <text />
          </element>
        </zeroOrMore>
    </element>
</define>
</grammar>
```

6.2. GetDataCategories(aProfile, aRegistrationStatus)

Returns a list of data categories registered for the specified profile. The registrationStatus is optional and refers to the registration status of the data category. Data categories are returned in a reduced form, containing the data category's ID, registration status, registration authority, identifier and version only (see listing 2).

6.3. getDataCategory(aURID)

Returns detailed data category information for the data category identified by the specified identifier. The results are delivered in the RelaxNG format of ISO 12620. A compressed model is shown in figure 5.

6.4. search(aKeywords, aFields, aProfile, aRegistrationStatus)

Searches the DCR for data categories matching the specified search criteria. An AND operator is assumed between the list of keywords. Wildcards may be used here. The search is performed over the specified list of fields. Possible field values are identifier, definition, explanation, example and note. An OR operator is assumed between the field elements. Profile and registration status are optional and indicate the profile and registration status of the data categories respectively. The list of data categories is returned in the same schema

format as for getDataCategories (aProfile, aRegistrationStatus).

```
<grammar
xmlns="http://relaxng.org/ns/structure/1.0">
 <start>
  <ref name="struct_DCS" />
  </start>
<define name="struct_DCS">
<element name="struct">
<attribute name="type">
  <value>DCS</value>
  </attribute>
<zeroOrMore>
  <ref name="struct_DC" />
  </zeroOrMore>
  </element>
  </define>
<define name="struct_DC">
<element name="struct">
<attribute name="type">
  <value>DC</value>
  </attribute>
<attribute name="id">
  <text />
  </attribute>
<element name="feat">
<attribute name="type">
  <value>registrationStatus</value>
  </attribute>
<choice>
  <value>standard</value>
  <value>qualified</value>
  <value>candidate</value>
  <value>retired</value>
  <value>superseded</value>
  </choice>
  </element>
<element name="feat">
<attribute name="type">
  <value>registrationAuthority</value>
  </attribute>
  <text />
  </element>
<element name="feat">
<attribute name="type">
  <value>identifier</value>
  </attribute>
  <text />
  </element>
<element name="feat">
<attribute name="type">
  <value>version</value>
  </attribute>
  <text />
  </element>
  </element>
  </define>
```

6.5. GetAllTopBroaderGenericConcepts()

Returns all data categories which are top level concepts, i.e. that have no broader generic concept. The list is returned in the same schema format as for getDataCategories(aProfile, aRegistrationStatus).

6.6. GetBroaderGenericConceptDataCategory (aURID)

Returns the list of data categories which are the broader generic concepts of the data category identified by the specified URID. The list is returned in the same schema format as for getDataCategories (aProfile, aRegistrationStatus).

Figure 5: compressed ISO DCR model

6.7. GetDataCategoriesUsingBroaderGenericConcept(aURID)

Returns all data categories referring to the data category identified by the specified URID as a broader generic concept. The list is returned in the same schema format as for getDataCategories (aProfile, aRegistrationStatus).

7. Conclusions

Using the ISO Data Category Registry improves interoperability of lexical resources by using well defined standardized linguistic concepts. To enable tool builders to interoperate with the DCR an application programming interface has been developed. The returned XML is well defined using RelaxNG schema's and a converter is available to convert the XML stream in an object representation.

8. References

[1] http://emeld.org/gold-ns
[2] http://syntax.inist.fr/
[3] http://www.tc37sc4.org
[4] http://syntax.inist.fr/page/home.RelaxNG.inc.php
[5] http://www.relaxng.org
[6] http://www.mpi.nl/lexus
[7] http://atoll.inria.fr/rnil/DCS_ELR.html