Originally published in: Calzolari, Nicoletta/Choukri, Khalid/Maegaard, Bente/Mariani, Joseph/Odijk, Jan/Piperidis, Stelios/Tapias, Daniel (Eds.): Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08). May 28-30, 2008, Marrakech, Morocco. – Paris: European Language Resources Association (ELRA), 2008. Pp. 3205-3212.

Lexicon Schemas and Related Data Models:when Standards Meet Users

Thorsten Trippel[1], Michael Maxwell[2], Greville Corbett[3], Cambell Prince[4], Christopher Manning[5], Stephen Grimes[6], and Steve Moran[7]

[1]Universität Bielefeld, Germany; [2]University of Maryland, USA; [3]University of Surrey, United Kingdom; [4]SIL, Thailand; [5]Stanford University, USA; [6]Indiana University, USA; [7]University of Washington, USA

E-mail: [1]thorsten.trippel@uni-bielefeld.de, [2]maxwell@umiacs.umd.edu, [3]g.corbett@surrey.ac.uk, [4]cambell_prince@sil.org, [5]manning@cs.stanford.edu, [6]stgrimes@indiana.edu, [7]stiv@u.washington.edu

Abstract

Lexicon schemas and their use are discussed in this paper from the perspective of lexicographers and field linguists. A variety of lexicon schemas have been developed, with goals ranging from computational lexicography (DATR) through archiving (LIFT, TEI) to standardization (LMF, FSR). A number of requirements for lexicon schemas are given. The lexicon schemas are introduced and compared to each other in terms of conversion and usability for this particular user group, using a common lexicon entry and providing examples for each schema under consideration. The formats are assessed and the final recommendation is given for the potential users, namely to request standard compliance from the developers of the tools used. This paper should foster a discussion between authors of standards, lexicographers and field linguists.

1. Introduction

This paper discusses requirements of lexicographers and people working with lexicographic resources on lexicon encodings and representation schemas. The paper is the result of a workshop of a group of lexicographers and lexicon theorists meeting at Stanford University in July 2007 to discuss which standard for lexicon encoding and representation should be recommended to people working in lexicography. It starts from the assumption that there is a large variety of lexicon formalisms used for the interchange and archiving of lexical resources. Many field workers face a problem of deciding which formalism is appropriate for their use and which should be taken into consideration. These lexicographers have an interest in sharing resources and archiving them, and would comply with a standard if possible to enhance interoperability. In this paper we discuss the requirements of lexicographers for representation schemas and evaluate which formalisms would be appropriate and could be used. This paper should foster a discussion between authors of standards, lexicographers and field linguists.

2. Requirements for Lexicon Schemas and Descriptions

Reviewing requirements of field linguists and lexicographers with different backgrounds we were able to create a list of 19 requirements, some of them mutually exclusive to others. However, we believe that by stating these requirements, it is possible to evaluate lexicon schemas and provide developers of lexicon software with a list of requirements of potential users.

2.1 Representation

2.1.1 Simplicity of the Lexicon Schema:

Lexicographers are not computer programmers hence the lexicon schema to be used should not be too like a programming language. Just as the rather simple versions of the hypertext markup language HTML have made the success of the world wide web possible, a lexicon schema has to be easy enough to allow a fast understanding of the basic concepts while allowing complex structures for advanced requirements. This does not mean that HTML should be the model for a lexicon schema, but

that a lexicographer with little programming background could use a specific lexicon schema.

2.1.2 Data category selection:

It seems to be impossible to fix all necessary lexical data categories (such as noun genders) for all languages. A lexicon schema has to allow for different data categories, as was discussed in the development of the new version of ISO 12620 (Data categories for use in the Terminology Markup Framework).

2.1.3 Make ambiguity explicit:

Lexical ambiguity such as homonymy and polysemy traditionally has been an issue for lexicographers. A lexicon model has to allow for this kind of ambiguity without imposing a disambiguation for example by enumeration. Disambiguation should be left to the lexicographer or an application.

2.2 Structure

2.2.1 Listing Lexical Data Categories:

Each lexicon representation needs to state which lexical data categories are available, as well as the categorization as mandatory or optional. Programs evaluating the lexical database can use the list for the syntactic evaluation of the resource.

2.2.2 Hierarchy of Lexical Data Categories:

Many lexical data categories are not independent of each other, but stand in a hierarchical relation. These hierarchies should be provided for, possibly with references to ontologies.

2.2.3 Explicit and Meaningful Structure Encoding:

For storing, archiving and interchange of lexicon data the interpretation of the structures is important. To allow this interchange, the structure of the lexicon schema has to be clearly defined and described explicitly. This becomes especially crucial in the future when interpreting tools may not be available any more.

2.2.4 Inheritance:

Lexical information can often be inherited from generalized patterns and rules, for example inflectional paradigm classes, etc. The lexicon schema should have the ability to represent inheritance models.

2.2.5 Definition of Relation to Grammar:

Lexicons and grammars are related to each other; they interact in the sense that the lexicon uses the lexical grammatical categories defined by the grammar. Grammatical implications should be made explicit in the documentation, which should be provided for in the schema.

2.3 Supported Information Types

2.3.1 Support for Different Writing Systems:

Many applications support Unicode and claim that they are enabled for different writing systems. However, applications often do not offer full Unicode support for changing writing systems and direction, or there is no font with the full set of characters needed. The lexicon schema should include ways of supporting different writing systems.

2.3.2 Support for Multiple Writing Systems:

A problem that appears artificial only at first glance is the use of multiple writing systems in the same lexicon article. Lexicon articles are the kind of text where such a change can happen frequently, for example in multilingual environments or with languages with more than one writing system.

2.3.3 Multimodal Data and Sign Language Representation:

For spoken language and for sign language representation the inclusion of multimodal data is obvious in a lexicon. This implies that idiosyncratic transcription systems and audio and video data have to be provided for.

2.3.4 Dependency Management:

For video formats it is necessary to specify which video encoding procedure (i.e. video codec) is used to interpret the signal. Another example of the dependency of a lexicon on external environments is the need for a font or specialized software to interpret relations modeled in a lexicon. Usually the data is not useless if those dependencies are not met, but users should know that they have to expect complications.

2.3.5 Include References:

A lexicon formalism needs to allow reference to material outside of the resource as such, e.g. bibliography, geographic locations, etc.

2.4 Conversion

2.4.1 Lossless Conversion:

As there are different lexicon schemas available already, a lexicon schema should state in which way a lexicon can be converted into a different format. That also means that a new lexicon schema has to be evaluated by testing it against lexical resources and by showing that these resources can be expressed by the schema. This also allows a user to take example lexicon encodings to understand the schema. The test of conversion can be demonstrated by converting from existing lexicon formats into the proposed schema and back. This conversion needs to be lossless, even over multiple cycles.

2.4.2 Common Format:

The lexicon standard format has to cover the core or intersection of all other data formats. If there is a lossless conversion, this is trivial, but it is not to be expected that lossless conversion will be possible in all instances

2.4.3 Flat Interchange Structure:

A flat table structure is the kind of structure most often used for interchanging lexical data, because it seems that it is the easiest to interpret and to map onto other structures. While flat structures appear to contradict the modeling of explicit hierarchies, it is in fact possible to map each hierarchical structures onto flat structures.

2.5 Applicability

2.5.1 Reuse of Existing Standards:

A lexicon schema should use and reuse institutional standards such as ISO standards and W3C standards wherever possible, e.g. for morphosyntactic features.

2.5.2 Existence of Applications:

An application which enables the use of a lexicon schema for linguists who are not programmers to foster acceptance of the schema.

2.5.3 Support for Existing Lexicons:

Lexicon encoding schemas will be useless for a user if the format does not support the lexicon used or designed by a this user. Hence it has to be tested and applied to multiple existing, well known lexical formats.

3. Characterization of Formalisms

3.1 DATR

DATR is a programming language for representing lexical information. It implements multiple non-monotonic inheritance. DATR is the oldest lexicon formalism looked at here, introduced by Evans and Gazdar (1996). They describe a modeling formalism and allow the representation of lexical knowledge in a text oriented format while also allowing the explicit inheritance of information from abstract or concrete lexical entries. The motivation for this is that the classification of a word according to some category implies a lot of additional information. For example, classifying an English word as a non-exceptional noun also means that the plural of this noun is created by adding the affix -s to the unmarked singular stem. Because English is not a highly inflecting language, this may not seem to be a major thing, but for highly inflecting languages this allows the creation of all possible word forms based on this form of generalization. Generalization is not restricted to morphology but can also be applied to syntactic roles, semantic representation, pronunciation, etc.

A basic idea of DATR is to define models for words and test them. Hence DATR defines theories for a given lexical item, and the DATR interpreter allows one checking if those theories are in fact correct. Since DATR originates from the computational linguistic and NLP tradition, there are numerous implementations available, which in turn means that the DATR formalism can be used and tested. Some implementations allow the use of models that are available in various encodings including Unicode, ASCII, ISO Latin 1, etc.

The lexicon data categories can be freely defined, if necessary even individually for each lexicon entry. The theories can be created by a text editor or with an appropriate user interface or generated by a program. This, however, poses one of the major problems of DATR in practical, lexicographic projects with users not overly familiar with

computational processes. Though the theories allow a wide flexibility of data categories, the theories themselves do not pose any linguistically based restrictions or data model for the individual lexicon entries.

Example of DATR lexicon entry with inheritance structures:

The starting point is the well understood DATR syntax, used for the Russian word romanized as 'komnata'.

In DATR the node label 'Komnata' is purely mnemonic and not used afterwards. A number or any other label could be substituted. The individual lexical data categories are represented in attribute value structure such as <> == N_II, which in this case means that by default all bits of information not specified elsewhere are to be inherited from the node N_II, which is a particular declension class of nouns. From that node there is a link to the node NOUN. From this part of the entry we can infer the lexical category, and since we know the morphological class we can infer almost all of the relevant inflections for six cases and two numbers. These are appended to the stem, which is given in the next line. That line gives no indication of any stem irregularity, so all the forms are taken to be the result of concatenating the stem plus a suffix. The reason that we cannot infer all the forms based on the information in just the first three lines of the entry is that there are well-known syncretisms in Russian based on animacy. The animacy must be inferred from the meaning, which is specified in the last line. Given that information, we can infer all twelve forms of Russian nouns. Note that gender is not specified. That too is inferred. There is no information given about sex, and so the gender is inferred from the inflectional class: nouns in N_II which are not sex-differentiable are feminine.

komnata:

<> == N_II

<gloss> == room

<infl_root> == komnat

<sem animacy> == inanimate.

3.2 Feature Structure Representation (FSR)

Feature Structures are frequently used in syntactic theories and other contexts to represent lexical information in a hierarchically structured format. The formalism as such does not provide a fixed set of data categories but a way of unifying two or more representations. One possible representation in a feature structure for a lexicon article would be the following:

$$\langle Komnata, \begin{bmatrix} N\ II & \\ Gloss & 'room' \\ sem & \begin{bmatrix} animacy & inanimate \end{bmatrix} \end{bmatrix} \rangle$$

The importance for language resources is underlined by the existence of an international standard for Feature Structure Representations (see ISO 24610-1:2006), which provides a normative way of encoding hierarchical structured data in XML format. The standard as such recommends the use of standardized data categories which are to be provided by a data category repository (see ISO 12620:2007 for a draft). The structure as such does not require the use of these data categories.

FSR representing inheritance structures:

```
<f name="komnata">
  <fs type="N_II">
    <f name="Gloss">
      <string>room</string>
    </f>
    <f name="infl_root">
      <string>komnat</string>
    </f>
    <f name="sem">
     <fs>
      <f name="animacy">
        <string>inanimate</string>
      </f>
     </fs>
    </f>
  </fs>
</f>
```

3.3 Lexical Markup Framework (LMF)

The Lexical Markup Framework (LMF, currently a draft international standard, ISO DIS 24613:2007) is a project of the International Organization of Standardization (ISO), Technical committee 37, Subcommittee 4 on language resources. Given the number of lexical resources produced in different fields and by different organizations, the need was felt to create a standard to describe, exchange and access electronically available lexical in-

formation. LMF intends to provide a 'meta-model' for creating lexicon formats, providing components to be combined to create a format. Based on the components used, the structure of lexical resources is supposed to be comparable and interchangeable. In this sense LMF is "the odd one out" in this list, in that there are code snippets of XML which are intended only as illustrations of how LMF would be applied. LMF is based on the assumption that a lexicon minimally contains a form-sense pair. It allows inheriting structures of data categories, which also implies the support of ontologies for lexical data categories. However, inheritance of lexical properties is not explicitly intended.

The draft includes a core specification for basic lexical entries, plus a number of annexes for syntactic information, morphological information (with distinct models for simple and complex morphologies), 'phrasal' entries, etc. As a meta-model, it defines the structures using UML, but does not give a required XML format for interchange.

Example of LMF implementation:

```
<Lexicon>
  <feat att="language" val="ru"/>
   <LexicalEntry>
    <feat att="partOfSpeech"
     val="NII"/>
    <Lemma>
     <feat att="writtenForm"
         val="komnata"/>
     </Lemma>
     <Sense>
      <feat att="animacy"
        val="inanimate"/>
      </Sense>
      <WordForm>
       <feat att="writtenForm"
        val="komnata"/>
       <feat att="grammaticalNumber"
        val="singular"/>
        <feat att="case" val="nom"/>
            </WordForm>
        </LexicalEntry>
    </Lexicon>
</LexicalResource>
```

3.4 Lexicon Interchange FormaT (LIFT)

The Lexicon Interchange Format (LIFT) as introduced by Hosken (2007) follows a different approach from DATR and LMF: The focus is not on lexical properties as inheritance, but on the lexical data as provided by tools from a fieldwork situation. The original motivation was that the Standard Format (SF), which is the data format of most SIL tools for field linguists (such as the widely used Shoebox/Toolbox programs), can contain structures which are only implicit and must be inferred based on the order of lexical data categories, such as data categories that are only relevant if there is a specific value in another data category.

An example for English would be the use of a data category (subject) person only if the word is a verb. Similarly, treating polysemy can be achieved in SF by repeating groups of data categories such as *definition, usage example*; the repetition of groups of categories hence implies that the word not only has more than one meaning, but also that those categories may depend on each other. In SF there is no way of expressing this (because there is no concept of a closing tag); in LIFT there is.

Additionally LIFT supports ontologies, which allows the defining not only of hierarchies of categories, but also the definition of a set of data categories as such, ensuring that consistency is achieved.

The underlying structure of LIFT entries also allows cross-references, i.e. links to other entries. Such cross-references are stated explicitly in SF files, but there is no guarantee that the target of such a cross-reference actually exists (and failed cross-references can, and typically do, arise in a number of ways). LIFT is ongoing work which is not yet fully stable. Grammatical information, for example, is not yet standardized. On the other hand, LIFT comes with tools that can be used in the fieldwork environment. One way of representing the previous example in LIFT is the following:

Example of LIFT entry:

```
<entry id="komnata"
     dateCreated="2008-04-01">
 <lexical-unit>
  <form lang="ru">
   <text>komnata</text>
  </form>
```

```
 </lexical-unit>
 <sense id="komnatasense">
  <gloss lang="en">
    <text>room</text>
  </gloss>
  <trait name="paradigm"
    value="N_II"></trait>
  <trait name="animacy"
    value="unanimate"></trait>
  <trait name="mor class"
 </sense>
</entry>
```

3.5 Text Encoding Initiative's Dictionary Format

The TEI encoding allows encoding dictionary articles of the traditional printed kind in structured, XML format. To allow for this, the lexical data categories appear in a structured way in each lexicon article. Some data categories are predefined in the specification, while others can be inserted as values of features to be specified.

Example of TEI entry:

```
<entry xml:lang="ru">
  <form type="lemma">Komnata:</form>
  <gramGrp>
  <gram type="inherits">N_II</gram>
   <gram type="animacy">inanimate
    </gram>
  </gramGrp>
  <sense xml:lang="en">
  <cit type="translation"
   xml:lang="en">
   <quote>room</quote>
  </cit>
 </sense>
</entry>
```

3.6 Tables in Character Separated Value (CSV) Format

The simplest formalism most associated with lexicons is that of a simple table. In a table format, each row represents a lexicon entry, and each column a lexical data category. Tables may be represented in simple database programs, or by means of Character Separated Values (CSVs), in which a particular character (often a comma or tab) is used to delimit fields.

This is the lexicon format with the flattest learning curve, but it does not support syntactic checking, any form of inheritance, ontology support etc. All of this would have to be provided at the time of lexicon structure design and implied in the lexicon structure.

CSV formats pose the same problems that originally started the design of LIFT, namely that dependencies between data categories are not explicitly given, and hierarchies are not part of the design. Furthermore, it does not readily support repeating fields (such as multiple usage examples for a single sense). In summary, while this format, together with some sort of prose description, is a simple form for humans to deal with, its limitations are clear.

Example of comma separated list entry:

komnata, N, II, fem, ‘room’

4. Conversion Between Different Lexicon Representation Formats

One way of investigating the expressive properties of a lexicon representation format is to show what needs to be done to convert one format into another. Possible options in this case are that information available in one format can be expressed by another format, either fully, with additional human logics, such as the interpretation of hierarchies of lexical data categories or naming of categories, or that some bit of information is lost on the way.

Table 1 shows the matrix of conversion from one format to another. All lexicon formats listed here are extensible in some way in order to provide additional data categories. Hence it is possible to represent the lexical information in all formalisms, with human intervention even without loss of information, though structural information may be lost on the way.

The described data formats seem to be restricted in one way or another: they either do not allow defining data category dependencies or hierarchies of categories within their structure (DATR in the data category names, CSV in the order of data categories) or they do not model inheritance of lexical information explicitly (LMF, CSV, LIFT). Some are open in respect of data categories

From ↓ to →	DATR	FSR	LMF	LIFT	TEI	CSV
DATR	-	Naming conventions of DATR theories used for hierarchies; inheritance structures can be expressed	Depends on the used data categories; LMF requires at least one form property.	Depends on the use of data categories; some data categories predefined in LIFT; tag misuse possible	Representation of fields possible; inheritance rules labelled as some kind of grammatical rules, danger of tag abuse	Full form lexicon: see DATR to FSR comment; for inheritance lexicons: similar but inheritance not explicit
FSR	Types of feature structures refer to inheritance, feature names to data categories; lossless	-	Depends on the used data categories; LMF requires at least one form property.	See DATR to LIFT comment	See DATR to TEI comment	Each data category is one column; multiple occurrences of same data category requires repetition of columns
LMF	Hierarchy of data categories representable in DATR category names; else simple	Hierarchy of data categories representable in FSR hierarchy names; else simple	-	Depends on the concrete implementation of LMF; examples in LMF are subject to the same problems as DATR conversion into LIFT	Depends on the concrete implementation of LMF; examples in LMF are subject to the same problems as DATR conversion into TEI	see LMF to FSR comment
LIFT	See FSR to DATR comment	See LMF to FSR comment	LIFT can be seen as one implementation of LMF	-	Different data category hierarchies	see LMF to FSR comment
TEI	Hierarchy of data categories representable in DATR category names or by abstract entries; lossless	See LMF to FSR comment	TEI can be seen as one implementation of LMF	Different data category hierarchies	-	see LMF to FSR comment
CSV	Each column is one data category in DATR, inheritance of DATR not used; lossless	Simple binary structure, lossless	See DATR to LMF comment	See DATR to LIFT comment	See DATR to TEI comment	-

Table 1: Converstion table for the different lexicon representation formats

(DATR, CSV, FSR), allowing multiple use of the same category within one element, while others have a core set of data categories and extensibility mechanisms (TEI, LIFT, LMF).

The availability of applications is another form of evaluation for the data formats, and it proves to be the most essential part: lexicographers and field workers require tools if they are to use a format. The text oriented formalisms discussed here can be edited in text editors, but this is not helpful for the average user. Though XML based editors can validate and assist users in the creation of lexical resources, these formats as such are not user oriented but implementation oriented. CSV formats can be exported from a spreadsheet program, while LIFT is

built in as export format in some SIL tools. For inserting data, these formats seem to be most relevant. For working with data, DATR provides more powerful interpretation functions with the DATR inference engines available (JDATR, ZDATR, etc.). The TEI format is best seen as a way of representing print dictionaries. LMF is a framework only. Feature Structure Representations are used for the interchange of lexical data for morphological and syntactic theories, but a front end for inserting data outside of the text editor or XML editor is currently unknown.

Formal models as the Lexicon Graph Model (Trippel, 2006) or other more general descriptions (for example discussed by Polguère, 2006) are not implemented for working in environments such as field work, laboratory lexicography, etc.

5. Summary and Recommendations for Users

The lexicon schemas and standards introduced here have the descriptive power for encoding the required lexical data categories. Some schemas are more open towards new data categories and do not discriminate against them, while others distinguish between a core set and extensions. The real problem for lexicographers and field linguists is hence not the lexicon schema but the application working with those schemas. With human interventions conversion is possible, so it remains for the user to ask software vendors for standard compliance and openness for data categories with reference to established standards or standards to be established.

6. Acknowledgements

The authors would like to thank the organizers of the workshop "Toward the Interoperability of Language Resources" organized at Stanford University in July 2007 and which allowed the authors to meet. The workshop was sponsored by The National Science Foundation (USA), the University of Kansas, Eastern Michigan University, and The Linguist List.

7. Bibliography

J. Crowther, editor (1995). Oxford Advanced Learner's Dictionary of Current English. Oxford University Press, Oxford, fifth edition.

R. Evans and G. Gazdar (1996). DATR : A language for lexical knowledge representation. Computational Linguistics, 22:167-216.

M. Hosken (2007). Lexicon interchange format: A description. URL: http://lift-standard.googlecode.com/files/lift_10.pdf, lookup 2008-03-31.

ISO 24610-1:2006. Language resource management -- feature structures - Part 1: Feature structure representation, International Standard.

ISO-DIS 12620:2007. Terminology and other content and language resources - data categories - specification of data categories and management of a data category registry for language resources, Draft International Standard.

ISO-DIS 24613:2007. Language resource management -- lexical markup framework (LMF), Draft International Standard.

A. Polguère (2006). Structural properties of Lexical Systems: Monolingual and Multilingual Perspectives. *Proceedings of the Workshop on Multilingual Language Resources and Interoperability* (COLING/ACL 2006), Sydney, 50-59

TEI P5. Dictionaries. In *TEI P5: Guidelines for Electronic Text Encoding and Interchange*. Feb. 2008. Section 9.

T. Trippel (2006). The Lexicon Graph Model: A generic Model for multimodal lexicon development. AQ-Verlag, Saarbrücken.