

Working Group 2

From EMELD

Working Group 2: Lexicon schemas and related data models

Members: Trippel (Co-Chair), Maxwell (Co-Chair), Corbett, Prince, Manning, Grimes, Moran, Mittelbach

Task: There are many proposed lexicons schemas and data models, but so far, no standard, de facto or otherwise. Arguably, this is the most salient gap in discipline-specific standards and one that only linguists can fill.

Test sets

An extended lexicon might also want to deal with grammatical information (cf. *constructicon* from Fillmore). Examples of entries might be (from Sri Lanka Malay):

- the suffix/clitic =*nang*
- relative clause structure ... (NP2 PRED2 NP2)(=POSTP1) ... PRED1
- finite adjunct clause (CLAUSE2)=POSTP1 SUBJ1 ... PRED1

A good test case for dictionary format could be Young & Morgan dictionary of Navaho (see for example [5] for and explanation of its complexity)

TILR Group 2: Lexicon schemas and related data models

Final Report

Overview

TILR group 2 was composed of Thorsten Trippel (Bielefeld University, Germany), Michael Maxwell (University of Maryland, USA), Greville Corbett (University of

Surrey, United Kingdom), Cambell Prince (SIL, Thailand), Christopher Manning (Stanford University Manning, USA), Stephen Grimes (Indiana University, USA), Steve Moran (University of Washington, USA). Group 2 was tasked with examining lexicon schemas as potential standards, in particular with

- Deciding what a standard schema should specify (and what it should not specify);
- Comparing the known schemas,
- Developing use cases to test them,
- Testing where the schemas deviate from each other by trying to transform a lexicon from one schema into another, and
- Recommending what standard would serve best for field linguistics.

In addition to schemas which have been developed for lexicons, there are several de facto standards for lexicons, which might be used to develop test cases or even be recommended as a potential standard.

Purpose

In the context of language description, particularly in the context of field linguistics, the potential uses of lexicon schemas include the following:

- Guideline for building lexicons
- Guideline for building lexicography tools
- Format for lexicon interchange (including merging of lexicons for a single language, and comparing different languages)
- Conversion of existing lexicons prior to importation into other tools
- Archival storage format for lexicons

What is a Lexicon?

In considering potential use cases by which lexicon schemas could be tested, the first question which needs to be addressed is: What is a lexicon? The answer may seem obvious, but in fact there are a number of ‘things’ which might be called lexicons, but which might stretch the definition:

1. Monolingual lexicons, bilingual or multilingual lexicons, which are collections of lexicographic information, often printed in books or brochures, containing wordforms and some sort of representation of their meaning, such as a form of definition, typical use cases, translations, but also containing various other types of lexical information such as grammatical information, morphological information, phonemic information, etc.
2. Web-based lexicons, hypertext encyclopedias, such as the famous web based examples where numerous lexicon entries are interlinked.

3. CD- or DVD-based lexicons, that are electronic versions of print dictionaries including the same type of information but often also additional audio-visual material.
4. OCRed versions of print dictionaries and printouts of electronic lexicons.
5. Lexicons created by machine learning (e.g. derived from a parallel corpus).
6. Lexicons with grammatical and phonological information, but no semantic information (e.g. for morphological parsing tools).
7. Lexicon management tools
8. Lexicons which have been formed by merging other lexicons
9. Speech lexicons (typically consisting of a transcribed word and one or more audio recordings of the word)
10. Term lists and other wordlists
11. Fieldwork lexicons, for example lexicons created with SIL's Shoebox/ Linguist's Toolbox/ Fieldworks or by the Multi Dictionary Formatter (MDF) or FLEX, often with inconsistencies, either due to adjustment of linguistic theories or due to free variation, errors, etc.
12. Lexical resources that exist in one of the conventional lexicon formats, such as the Lexicon Markup Framework (LMF, project by ISO TC 37 SC 4, proposed international standard), its predecessors such as the Multilingual ISLE Lexical entry (MILE), the Lexicon Interchange Format (LIFT, proposed by SIL).
13. Full form lexicons as generated by lexicon models, such as DATR, which allows to associate lexicon entries with explicit reference to paradigms such as syntactic classes to generate all possible wordforms based on one lexicon entry and the definition of the paradigm

We chose to view a minimal lexicon as being a computer-storable and readable list containing words and/or phrases, each associated with semantic information (one or more glosses, definitions, or translation equivalents). This covers most instances of items 1, 2, 3, 4, 5, 8, 11, 12, and 13, but not 6, 7, or 9. Instances of 10 may or may not be considered as lexicons under this definition.

Potential Schemas and de facto Standards

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 modelling formalism that allows the representation of lexical knowledge in a text oriented format and allows explicitly inheriting information from abstract or other lexical entries. The idea of 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 using the unmarked singular stem and adding the affix -s. Because English is not a highly inflecting language, this may not seem to be a major thing, but for high inflecting languages this allows actually to create all possible wordforms based on this form of generalization. However, this form of generalization is not restricted to morphology but can also be applied to syntactic roles, semantic representation, pronunciation, etc.

The 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 to check if those theories are in fact correct. For this reason for DATR as originating from the computational linguistic and NLP tradition there are numerous implementations available, which means that the DATR formalism actually works, 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 restrictions or data model for the individual lexicon entries.

Advantages:

- It has been tested and works on a wide variety of complex data.
- Mature.
- Exists in several implementations, new implementations could include statistical aspects of lexicons as well

Disadvantages:

- Steep learning curve; it is a programming language, and most field linguists are not programmers.
- Hypothesis testing requires “compilation”
- Unclear if a programming language is appropriate as an interchange format. However, it might be possible to create “flat dumps”, i.e. an output format in which all inheritance has been done to create fully instantiated lexical entries.
- Nested structures which can be created due to the inheritance model can make the model complex, which can cause confusion in an interchange process, especially when contradicting modes exist
- No check if mandatory data categories are defined or deprecated ones used.

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 some standard to describe, exchange and access electronically available lexical information. LMF does not provide a data format as such but intends to provide a 'meta-model' for creating such 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, as there are only a couple of snippets of XML as illustrations of how LMF would be applied. LMF is based on the assumption that a lexicon is always based on 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. (However, some of the examples include XML.)

Advantages:

- Covers a wide variety of lexical information, including grammatical information.

Disadvantages:

- The specification is difficult to understand in parts, e.g. the "intensional" morphology.
- LMF has not yet been tested on a wide variety of lexicons. In particular, it is unclear how some of the annexes would interact (e.g. phrasal lexical entries and intensional morphology).
- LMF does not specify an XML format; while this is probably intentional, an XML format would make interchange simpler.

LIFT

The Lexicon Interchange Format (LIFT) as introduced by Hosken (2007) follows a different approach from DATR and MILE: The focus is not on such lexical properties as inheritance, but on the lexical data as provided by tools from a fieldwork situation. The original motivation was that the Standard Data Format (SDF), which is the data format of most SIL tools for field linguists (such as the widely used Shoebox/ Toolbox programs), can contain implicit structures 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 SDF 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 SDF 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 defining not only 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 implicitly in SDF 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).

A disadvantage of LIFT in its current state is that it is a work in progress. Grammatical information, for example, is not yet standardized.

MILE

The Multilingual ISLE Lexicon Entry MILE (see for example Ide, et. al. 2003) is the result of a project working with morphosyntactic lexical information in the trans-atlantic project ISLE. MILE is based on an XML representation, giving the full advantages of tools available for XML, such as parsers and editors. It includes an inheritance model as well, which in combination with the XML syntax can be used to enforce syntactic checking. However, it is not as flexible in terms of data categories and substructures as DATR, as the concentration on morphosyntax resulted in constraints showing this focus.

The MILE format itself has a steep learning curve as well, requiring a user to understand XML (or in other implementations even RDF) in addition to the morphosyntactic models behind MILE.

MILE is not meant to be a format for producing lexicons, but a format for interchange of lexical resources. Tools which directly support MILE and could be used for general lexicographic work, for example in fieldwork situations, do not appear to exist; it is basically supported by in-house tools of the developers. (And it is not clear that there has been any development work since the 2003 paper.)

Tables

The simplest and most often with lexicons associated formalism 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.

However, 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.

Summary of available data formats

Available data formats seem to be restricted in one or the other way: they either do not allow the formulation of data category dependencies or hierarchies of categories (DATR, CSV) or they do not model inheritance of lexical information explicitly (MILE, LMF, CSV, LIFT). The learning curve is steep (DATR, MILE, LMF, LIFT). For all these models, support by existing programs is lacking.

Formal models as the Lexicon Graph Model (Trippel, 2006) that actually have a very simple representation and allow hierarchies and the restriction of data categories while remaining completely open, are not implemented for working in environments such as field work, laboratory lexicography, etc.

The following sections will try to model different use cases and implement them in those different formats, seeing where in the practical modeling problems occur.

Use cases

We believe a next step in deciding among models (or promoting improvements in models) is to design some test cases, by which models could be tested and then compared.

The definition of use cases for testing lexicon models is problematic. It seems that most of the designers of models have had their standard examples according to which the

formalisms work perfectly well, but they have not looked at other use cases which would allow direct comparison among models.

A first use case could be a morphological lexicon, to be reused by a morphological or syntactic parser. Another possible use case could be induction of sound change patterns from a lexicon base, for example upon affixation. This would be a test for inheritance and the power of inheritance facilities within a formalism. Use cases exercising various aspects of lexical semantics are also needed.

Issues in use case design

To allow lexical evaluation using use cases it is necessary to find minimal pairs of cases which can be used to precisely describe weaknesses and strengths of different approaches. For making sure that the entries are not artificial the proposed use cases are taken from real examples contributed by various authors. This test set should be designed to provide as precise cases as possible, from as many backgrounds as possible.

The test cases could be implemented in various models by the authors of those models, or by other researchers based on their understanding of the various models' specifications. In the latter case, part of the evaluation would be to document if and where there is a problem implementing those examples, and where the specifications are imprecise or misleading, as well as where the documentation actually helps to solve a problem. The idea would then be to approach the developers of the specifications, asking for their advice.

However, to allow the reimplementations in a different format and to make the information content of the examples more obvious it is required to also include a prose description of what a specific lexicon article actually states. For example it will not be sufficient to give an example in DATR, because developers of other models may not be familiar with DATR. Use cases should be provided in at least one format mentioned here, using Unicode. For documentation purposes examples should also come with a CSV list format, in order to better distinguish content from data categories. This will also help developers unfamiliar with some of the conventions to better understand the issues described in the prose text. Furthermore, the CSV format will provide a simple way of using cut and paste techniques.

Sample use cases

Several use cases are currently available in one or another format.

Russian lexical item 1: komnata

This use case is given as a DATR entry (from Corbett and Fraser, 1993); for fuller information see the paper and the fuller fragments at

<http://www.informatics.sussex.ac.uk/research/groups/nlp/datr/> (go to Index to the dtr archive, and then to Russian.doc and the files indicated there). The example here is given in a Latin script, rather than Cyrillic.

```
Komnata:  
<> == N_II      <gloss> == room  
<infl_root> == komnat  
<sem animacy> == inanimate.
```

Notes:

1. The node label 'Komnata' is purely mnemonic – the form Komnata is not used in what follows. A number or any other label could be substituted.
2. <> == N_II:
3. This indicates that by default all information is to be inherited from the node N_II. From that node there is a link to the node NOUN. So 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. That is, the 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.
4. 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 is inferable from the meaning, but here we specify it in the last line. Given that information, we can infer all twelve forms.
5. 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.

Thus the information contained in the entry is expanded out by the DATR compiler as follows:

```
Komnata: <gloss> = room.  
Komnata: <mor nom sg> = (komnat _a).  
Komnata: <mor acc sg> = (komnat _u).  
Komnata: <mor gen sg> = (komnat _i).  
Komnata: <mor dat sg> = (komnat _e).  
Komnata: <mor inst sg> = (komnat _oj).  
Komnata: <mor loc sg> = (komnat _e).  
Komnata: <mor nom pl> = (komnat _i).  
Komnata: <mor acc pl> = (komnat _i).  
Komnata: <mor gen pl> = komnat.  
Komnata: <mor dat pl> = (komnat _a _m).  
Komnata: <mor inst pl> = (komnat _a_m´i).  
Komnata: <mor loc pl> = (komnat _a _x).  
Komnata: <syn gender> = fem.  
Komnata: <syn animacy> = inanimate.
```

COMMA SEPARATED LIST

komnata, N, II, fem, 'room'

This same entry is represented as follows in LIFT: (some abbreviations)

```
<RNGSchema="file:/lift.rng" type="xml"?>
  <lift>
    <entry id="komnata274">
      <lexical-unit>
        <form lang="latin">
          <text>komnata</text>
        </form>
        <grammatical-info type="N"
stem="kamnat"gender="fem"></grammatical-info>
        <gloss
lang="en"><text>room</text></gloss>
        <trait name="mor class"
value="2"></trait>
        <reference type="m" ref="3">
          <range name="mor class"></range>
        </lexical-unit>
      </entry>
    </lift>
```

Russian lexical item 2: dama

Again, this is given as a DATR entry; see the fragments at <http://www.informatics.sussex.ac.uk/research/groups/nlp/datr/> (go to Index to the dtr archive, and then to Russian.doc and the files indicated there). Again, the example is given in a Latin script, rather than Cyrillic.

```
Dama :
<> == NOUN
<gloss> == lady
<infl_root all> == dam
<sem sex> == female.
```

Notes:

1. The node label 'Dama' is again just mnemonic.

2. The fact that sex is specified will lead to the inference that the noun is semantically animate. That in turn will predict the syncretism of accusative plural as genitive plural.
3. Note that gender is not specified, rather it is inferred: since information is given about sex, the inflectional class can be inferred (nouns denoting females belong to N_II unless otherwise specified) and gender can also be inferred.

Thus the information contained in the entry is expanded out by the DATR compiler as follows:

```
Dama: <gloss> = lady.
      Dama: <mor nom sg> = (dam _a).
      Dama: <mor acc sg> = (dam _u).
      Dama: <mor gen sg> = (dam _i).
      Dama: <mor dat sg> = (dam _e).
      Dama: <mor inst sg> = (dam _oj).
      Dama: <mor loc sg> = (dam _e).
      Dama: <mor nom pl> = (dam _i).
      Dama: <mor acc pl> = dam.
      Dama: <mor gen pl> = dam.
      Dama: <mor dat pl> = (dam _a _m).
      Dama: <mor inst pl> = (dam _a _m`i).
      Dama: <mor loc pl> = (dam _a _x).
      Dama: <syn gender> = fem.
      Dama: <syn animacy> = inanimate.
```

COMMA SEPARATED LIST

Dama, N, II, fem, animate, 'lady'

Russian lexical item 3: d'ad'a

Again, this is a DATR entry; see the fragments at <http://www.informatics.sussex.ac.uk/research/groups/nlp/datr/> (go to Index to the dtr archive, and then to Russian.doc and the files indicated there).

```
D'ad'a:
      <> == NOUN
      <declensional_class> == N_II:<>
      <gloss> == uncle
      <infl_root all> == d'ad'
      <phon stem hardness> == soft
      <mor gen pl> == MGP:<soft>
      <sem sex> == male.
```

Notes:

1. This is a phonological transcription, NOT a transliteration..

2. The fact that sex is specified will lead to the inference that the noun is semantically animate. That in turn will predict the syncretism of accusative plural as genitive plural.
3. Gender is not specified, because it is inferable from the <sem sex> tag.
4. The inflectional class is not that expected for males, and so is specified (it cannot be inferred from the inflectional root, and the entry is purely mnemonic, not used for inference).
5. The form of the genitive plural is inferred via a set of conditions, including the softness (palatalization) of the final consonant.
6. This item illustrates the need to be able to override the expectation that sex/gender/ inflectional class will line up, and the ability to specify one cell of the paradigm as requiring additional specification.

Thus the information contained in the entry is expanded out by the DATR compiler as follows:

```
D'ad'a: <gloss> = uncle.
      D'ad'a: <mor nom sg> = (d'ad' _a).
      D'ad'a: <mor acc sg> = (d'ad' _u).
      D'ad'a: <mor gen sg> = (d'ad' _i).
      D'ad'a: <mor dat sg> = (d'ad' _e).
      D'ad'a: <mor inst sg> = (d'ad' _oj).
      D'ad'a: <mor loc sg> = (d'ad' _e).
      D'ad'a: <mor nom pl> = (d'ad' _i).
      D'ad'a: <mor acc pl> = (d'ad' _ej).
      D'ad'a: <mor gen pl> = (d'ad' _ej).
      D'ad'a: <mor dat pl> = (d'ad' _a _m).
      D'ad'a: <mor inst pl> = (d'ad' _a _m`i).
      D'ad'a: <mor loc pl> = (d'ad' _a _x).
      D'ad'a: <syn gender> = masc.
      D'ad'a: <syn animacy> = animate.
```

COMMA SEPARATED LIST

```
D'ad'a, N, II, gen pl d'ad'ej, masc,animate, `uncle'
```

Complex morphology

- to be provided by Mike Maxwell

Semantic example: polysemy

Cherry:

```
<> == NOUN
<mor root> == cherry
<sem gloss i> == sweet red berry with pip
<sem gloss 2> == tree bearing <sem
gloss i>
<sem gloss 3> == wood from <sem gloss 2>.
```

Multiple gloss/multiple writing example

- to be provided by Cambell Price

One format has it all, or The egg-laying-wool-growing-milk-giving-meat-providing animal of lexicography

The impossible comes true if and when the complete wish list for a lexicon format are fulfilled. The following list is this wish list, different users will give priority in a different order, hence no priority is implied. Some of the items in this list seem to pose a contradiction which may be hard to overcome. At least by saying what is wanted and what is necessary it becomes possible to evaluate both existing standards and approaches under development.

1. Simplicity of the formalism: Taking hypertext as the example, the success story of the world wide web was probably due to the rather simple structure of HTML, which made it easy to understand and write. Of course, when looking at rather complex web pages today there are a lot of examples of very complex structures showing that the simple structure of HTML can express complex text structures. We are not proposing that HTML (which is a presentation mark-up scheme, rather than a content mark-up scheme) should be the structure for lexical information. Still, the lexical model has to be as easy for lexicographers and field linguists to understand, and the markup scheme must be easy for programmers to implement. That means that a rather short training for using the data format must be sufficient.
2. Data category selection: As in the context of the new version of ISO 12620 (Data categories for the use in the Terminology Markup Framework) it seems impossible to standardize the lexical data categories and a specific hierarchy of data categories. Experience shows that even defining a minimum independent of a concrete project poses a problem. Therefore it needs to be left open in the standard which categories need to be used.
3. Listing lexical data categories: For lexicon data interchange, it is necessary to list in the resource the data categories that are actually used, thereby informing users of the content of the lexical database. Based on this list, programs evaluating the lexical database can verify that mandatory data categories are used and that undefined data categories are not used.
4. Hierarchy of lexical data categories: A hierarchy of data categories, perhaps with reference to an ontology, should be given to explicitly express their interrelation.
5. Verbose and meaningful structure encoding: A lexicon data format should not only provide for interchange between different applications, but also for storage and archiving. This means that naming conventions of structures need to be verbose and clearly defined by extensive documentation with examples. This is especially relevant for long time archiving, when tools may no longer be available but the structures can be used to reconstruct the original information contained in the resource.
6. Support for different writing systems: Applications supporting Unicode could claim to already support different writing systems. However, there are still some

issues on character encodings that need to be solved. Though Unicode allows changing writing systems and directionality, tools may not (yet) provide full support; fonts may not be available for the full set of characters; and legacy data needs to be handled. The lexicon format needs to answer the encoding question.

7. Support for multiple writing systems: In some cases even within a lexicon article different writing systems need to be supported. For example, glossing may be done in more than one language or more than one writing system.
8. Multimodal data and sign language representation: Sign language representation and multimodal data may not seem to be closely related, but in fact they are: there are many partly idiosyncratic transcription systems for sign languages, and lexical resources for sign languages also include video sequences. The lexicon format has to provide for that.
9. Dependency management: A lexicon format needs to specify what is necessary to interpret it, for example a font, a video codec (see sign languages), special software to interpret relations, etc. That does not mean that the data is useless if those dependencies are not met, but at least a user would know the complications involved in using the data.
10. Lossless conversion: A lexicon formalism has to prove that its expressive power is as great as existing formats. This can be demonstrated by converting from those formats into the standard and back. This conversion needs to be lossless, even over multiple cycles.
11. Common format: The lexicon formalism has to cover the core or intersection of all other data formats. If there is a lossless conversion, this is trivial, but it means that all lexicon formalisms may intersect and this intersection should be mandatory. This is a clash with the requirement of being completely open in terms of data categories. However, it may be the case that no such intersection exists.
12. Flat interchange structure: For interchange purposes there should be a flat structure, because this is the easiest to interpret and map onto other structures. This contradicts the modeling of explicit hierarchies only superficially as it is possible to map each hierarchical structure on a flat structure. Such a flat structure then requires a prose description of the implied hierarchies to allow recreation of the hierarchy. It basically means also that for each hierarchical lexicon structure a flat structure is part of the documentation.
13. Include references: A lexicon formalism needs to allow reference to material outside of the resource as such.
14. Make ambiguity explicit: Traditional lexicographers have worried about cases of ambiguity such as homonymy and polysemy. It seems advisable to allow such ambiguity without restricting the model to disambiguating lexical information by enumeration. This should be left to the application.
15. Reuse of existing standards: ISO standards and W3C standards should be used wherever possible, e.g. for morphosyntactic features.
16. Existence of applications: A formalism as such is no good if there is no widespread acceptance and use from within applications.
17. Inheritance: For example, paradigms and classes of paradigms need to be allowed.

18. Define relation to grammar: Lexicons must be designed to interact with the grammar of the language, in the sense that the lexicon uses the lexical grammatical categories defined by the grammar. These grammatical implications should be made explicit in the documentation.
19. Support existing lexicons: The best formalism will lack acceptability if the format does not support the lexicon that a particular user actually uses. Hence it has to be tested and applied to multiple existing, well known and used lexical formats.

Summary

One of the principle problems seen with the implementation of the use cases is that most formalisms have been developed top-down. Here, this means based on a particular lexicon, a generalization was attempted without trying to test the generalizations on other, widely used lexicons. Hence the standard is overfitting to a specific case, rather than to general use. This may also be due to a lack of understanding of the expressiveness of the formalisms, but that often means that the documentation is unusable for someone who has not been working with the development of the standards. The solution is either to specifically show that the formalism has some abilities from the wish-list described here, or to enhance the formalism to create a general bottom up standard which actually maps what is being done and thereby elaborates on work that is being done.

Bibliography

- Corbett, Greville G.; and Fraser, Norman. 1993. Network Morphology: a DATR account of Russian nominal inflection. *Journal of Linguistics* 29 (1). 113-42.
- Evans, Roger and Gazdar, Gerald. 1996. "DATR: A Language for Lexical Knowledge Representation." *Computational Linguistics* 22.167-216. <http://acl.ldc.upenn.edu/J/J96/J96-2002.pdf>
- Hosken, Martin. 2007. "Lexicon Interchange Format (LIFT): A description." http://lift-standard.googlecode.com/files/lift_10.pdf
- Ide, N., Lenci, A., Calzolari, N. 2003. "RDF Instantiation of ISLE/MILE Lexical Entries." *Proceedings of ACL 03 Workshop on Linguistic Annotation: Getting the Model Right*, Sapporo, 30-37. <http://www.cs.vassar.edu/~ide/papers/ACL03-ws-ISLE.pdf>
- ISO DIS 24613 (2007): "Lexical Markup Framework", ISO, Geneva.
- Trippel, T. (2006) *The Lexicon Graph Model: A generic Model for multimodal lexicon development* AQ-Verlag, Saarbrücken