A knowledge-modeling approach for multilingual Regulus lexica

Marianne Santaholma and Nikos Chatzichrisafis
University of Geneva, ETI/TIM/ISSCO
40, bd du Pont-d'Arve, CH-1211 Geneva
E-mail: Marianne.Santaholma@eti.unige.ch; Nikos.Chatzichrisafis@vozzup.com

Abstract

Development of lexical resources is, along with grammar development, one of the main efforts when building multilingual NLP applications. In this paper, we present a tool-based approach for more efficient manual lexicon development for a spoken language translation system. The approach in particular addresses the common problems of multilingual lexica including the redundancy of encoded information and inconsistency of lexica of different languages. The general benefits of this practical tool-based approach are clear and user-friendly lexicon structure, inheritance of information inside of a language and between different system languages, and transparency and consistency of coverage between system languages. The visual tool-based approach is user-friendly to linguistic informants that don't have previous experience of lexicon development, while at the same time, it still is a powerful tool for expert system developers.

1. Introduction

One of the main efforts in developing multilingual applications that are based on linguistic knowledge is to build the required language resources. These resources include grammars and lexica. The grammar may contain rules for both analysis of system input, and for producing grammatical output by combining words into larger constituents. The lexica store the lexical entries including different type of linguistic information, like syntactico-semantic features. The information encoded in the lexicon depends on the grammar theory applied and the underlying grammar development environment. Due to their complexity, these resources are habitually constructed by hand, which makes maintenance labor-intensive and time-consuming task, especially in systems where multiple languages are developed.

Recent efforts to facilitate the construction of lexical resources have concentrated on exploiting the available digital resources like machine readable dictionaries and multilingual corpora. These automatic methods of lexical acquisition can currently help to derive lexical entries, but the learning of required complex linguistic information included in the entries remains problematic. Only some particular, well-defined linguistic properties, like verb subcategorization in a certain language can be defined automatically (e.g. Korhonen, 2002). Consequently, these data-based methods are primarily used to extend the already existing lexica (as in Baldwin, 2005) instead of building a lexicon for a new language from scratch. Despite of increasing amount of electronic data available, many languages and application domains are still scarce of resources required for this type of lexical acquisition.

As automatic methods do not yet provide a global solution we focus on assisting manual lexicon development with a tool that improves following aspects of multilingual system lexicon development:

Porting of lexica into new languages and domains. The manual creation of new system lexica commonly includes the involvement of expert developers in many stages (set-up, structuring, refactoring, entering data and maintenance). However, ideally populating the lexicon and its maintenance would be performed by linguistic informants who are familiar with the system specific expressions but do not necessarily have deep knowledge about language engineering. Hence extending the lexical coverage and maintaining it should be made simple. This favors visual tools over the basic coding environments.

Redundancy of information. Multilingual systems cover the equivalent expressions in several languages and in consequence include the equivalent lexical entries for all these languages. These entries are highly similar: they are written in the same application specific formalisms and they express the same type of information. Hence, there is a fair amount of repetition in lexica of different languages and even inside a lexicon of a language. Sharing the representations and information between different languages and inside of a language would reduce this redundancy and consequently facilitate and speed up the development of multilingual lexica.

Structure of multilingual lexica. Multilingual applications, as for example translation systems, often have the same coverage for all supported languages. A visually assisted browsing across system languages helps the system developer to verify the coverage and locate possible lexical gaps and inconsistencies between languages.

Focusing on above aspects we have implemented a development and management tool for writing multilingual Regulus lexica. Regulus is an Open-Source toolkit for developing feature grammars and lexica for spoken language (Rayner, Hockey & Bouillon, 2006; Regulus, 2008). These lexica are used, among others, in multilingual medical domain spoken language translator, MedSLT (Bouillon et al, 2005; MedSLT, 2008). MedSLT translates doctor-patient spoken dialog in medical diagnosis situation. The currently covered system languages include Arabic, Catalan, English, Finnish, French, Japanese, and Spanish.

The main components of the system (speech recognition, parsing and generation) are built on the linguistically motivated constraint-based Regulus grammars.

The lexical resources for MedSLT are not readily available for the supported domains and languages, and therefore they have to be separately developed. This becomes a tedious task for new system languages, as already existing coverage needs to be matched in a consistent matter. In order to accelerate system development and help with maintenance tasks we have implemented a tool-based solution for multilingual lexica. Instead of designing a new application for this, we use as starting point the open-source platform Protégé.

The remainder of this paper is structured as follows. The next section introduces Regulus lexica as they are used in MedSLT. Section 3 then describes how multilingual Regulus lexica can be represented in Protégé. Section 4 describes the implemented lexical hierarchy. The paper concludes with Section 5.

2. Regulus MedSLT lexica

The Regulus lexical entries are based on Prolog syntax, and are typically written in a text editor. The basic lexical entry has the format illustrated in Example 1. It consists of four parts:
(a) Lexical category name: `verb`
(b) Domain specific semantic representation:
`sem=[[state,sleep], [tense,present]]`
(c) Flat list of different constraints in form of attribute-value pairs:
`vform=infinitive,`
`agr=(sg/\3),subcat=intransitive,`
`sem_pp_type=duration`
(d) Surface form of lexical entry: `sleeps`.

```
verb:[sem=[[state,sleep],[tense,present]],
vform=infinitive,agr=(sg/\3),
subj_sem_n_type=person,
subcat=intransitive, sem_pp_type=duration,
takes_adv_type=(frequency\/duration)]
--> sleeps.
```

Example 1: Regulus lexical entry for ‘sleeps’.

Characteristic for Regulus lexical entries is the heavy use of sortal constraints. These features define the range of context in which each word can occur. For example the above illustrated verb ‘to sleep’ takes according to the lexical rule of Example 1 as subject a noun that represents the semantic ‘person’ (`subj_sem_n_type=person`). Furthermore the allowed prepositional phrase complement is of type “temporal” (`sem_pp_type=duration`). Regulus grammars and lexica are compiled into context free grammar (CFG) language models for the purposes of speech recognition. These sortal features help to constrain the representation to be suitable for this CFG compilation procedure.

Morphological tools are not integrated in Regulus. Currently the different forms of one lemma have to be enumerated in the lexicon. Hence, the amount of entries that only slightly differ from each other can be quite extensive. This means a significant amount of repetition of the same information in entries that are almost equal, like ‘sleep’ and ‘sleeps’. This type of redundancy is decreased in Regulus lexica by macros. Macros are used as templates that capture the common information. For example the common features (like subcategorization, subject type, allowed prepositional phrase complement type) of intransitive verbs ‘sleep’ and ‘walk’, can be generalized under one macro rule. This type of rule is illustrated in Example 2 for verb ‘*to sleep*’. Instead of enumerating the different surface forms of sleep (`sleep, sleeps, slept, slept, sleeping`) as separate entries in the lexicon, they are grouped in a single entry that begins with the `@v_intransitive` macro invocation. The macro rule `v_intransitive` assembles the information that is shared between all the similar intransitive verbs. Furthermore, this macro rule contains two other macro invocations, `@verb` and `@verb_sem`. This way lexical entries inherit information from several different sources and consequently the redundancy in rule writing is effectively reduced. Additionally required modifications during reengineering of grammars and lexical entries have to be introduced only in macro rules instead separately in all lexical entries.

```
(a) Lexical entry for ‘to sleep’
@v_intransitive
([sleep, sleeps, slept, slept, sleeping],
[action, sleep], [agent], [takes_time_pp=y,
takes_frequency_pp=y,takes_duration_pp=y].

(b) Macro rule v_intransitive
macro(v_intransitive
(SurfaceForms, [SemType, SemConstant],
[SubjSortalType], OtherFeats),
@verb(SurfaceForms, [@verb_sem(SemType,
SemConstant)], [subcat=intransitive, inv=n,
subj_sem_n_type=SubjSortalType|
OtherFeats])).
```

Example 2: Macro rule for intransitive verbs like ‘to sleep’

However, dealing with these macros, especially when multiple inheritance levels are involved, is not a trivial task. When different developers build several levels of macro invocations for different languages, acquiring an overview and performing simple maintenance tasks may become a complex endeavor. This can be especially demanding for inexperienced lexicon developers.

To make development easier for inexperienced users, and to increase transparency throughout all languages, we

considered using a visual development tool.

MedSLT lexica were customarily developed as monolingual resources. In the context of the multilingual shared grammar project (Santaholma, 2007), we required an easy way for capturing generalizations not only in one language but also between several languages.

3. Multilingual lexicon development and management tool

Instead of implementing a new multilingual lexicon toolkit from scratch, we based our development on the Protégé platform. Protégé is a popular open source ontology editor and knowledge base framework (Protégé, 2008). Protégé supports the export to standard ontology languages as OWL and RDF Schema, but it is easily extensible through its plug-in interface. This makes it a flexible base for a rapid prototyping and application development. We extended Protégé with the Regulus exporter plug-in (Chatzichrisafis & Santaholma, 2007). The plug-in exports Regulus-compatible files directly from the Protégé user interface. These files can then be included from Regulus grammars or from Regulus configuration files as part of Regulus based application.

Protégé has several built-in features that we found advantageous for multilingual lexicon development and management. These include a standardized graphical user interface (GUI) and flexible platform for knowledge-based domain modeling. The following sections describe the Protégé features in detail, and demonstrate how we used them for multilingual lexicon development.

3.1 Defining Regulus lexical entries with Protégé

The Protégé GUI consists of overlapping tabs that offer a 'browser' and 'form' for creation, viewing, editing, and saving different type of information. These tabs include '*classes*', '*slots*' and '*instances*' (Figure 1). Protégé *classes* represent originally the abstract domain concepts. Each of these abstract classes is described by a set of defined attributes. These are in Protégé called *slots*. The concrete class occurrences are represented in Protégé as *instances*. We use these three forms to enter the different type of information required in the Regulus lexical entries: lexical categories, semantic representation, attributes and their values, and the word form. Their associations and function are presented in the following.

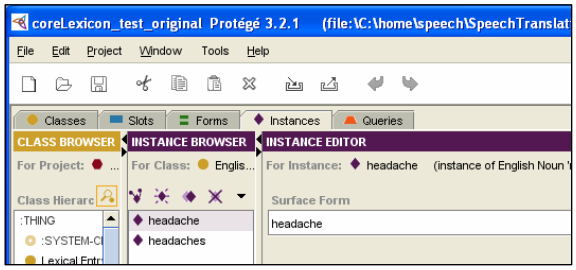

Figure 1: Protégé tabs.

Lexical categories. Lexical categories like (noun, adjective, pronoun, verb, etc) are introduced as classes.

Semantic representation and attribute-value pairs. The Regulus semantic representation and various features such as sortal constraints are introduced in Protégé Regulus lexicon as *slots*. Protégé slot form allows defining these attributes with a fixed set of possible values. It also supports a variety of values including the Boolean type values 'true' and 'false' (`takes_determinant=true`) and list of symbolic strings ('temporal' and 'duration' in `obj_sem_np_type=temporal\/duration`). Furthermore the slot form includes a facet where the permitted amount of different values for an attribute can be defined. The slot form provides a user-friendly interface for defining all attributes and their value types that are typical for the Regulus formalism.

Surface form. The word forms (like 'sleep' and sleeps') are introduced in Protégé as concrete *instances* of classes. The *instances* form displays a collection of fields that represent the attributes that are required for the lexical category in question. The display and options for each attribute-value field thus depend on the type of information that has been defined in the slot form. The instance field for Finnish noun 'kuume', *feverI*, is illustrated in Figure 2. The lexical entry has attribute fields for `surface_form`, `sem_np_type`, `N_type`, `can_be_pp`, `sem`, `takes_det_type` and `case`. In case the required value is a symbolic string like '`symptom`' in `sem_np_type =symptom`, the field includes a combo box that contains the possible value(s). The Boolean '`true`'/'`false`' value is displayed as check box as for `can_be_pp`.

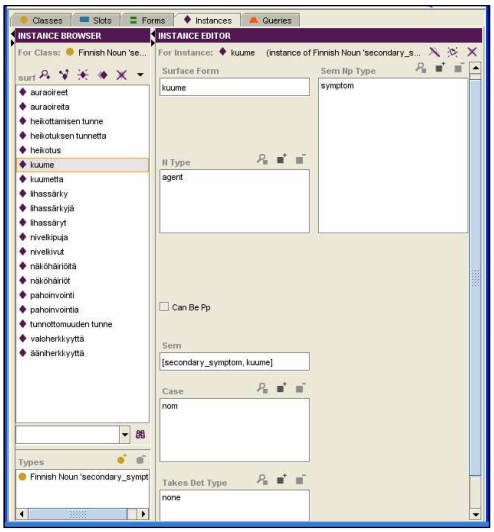

Figure 2: Instance form.

The developed Protégé Regulus lexicon is exported into Regulus format by iterating through all instances of relevant lexical classes and writing out lexical entries into target files. The output of this export procedure is a regular Regulus lexicon file, which can be included as part of Regulus grammars.

3.2 User-friendly tool for different user groups

By separating the different type of information in distinct tabs Protégé offers customized working environments for different type of lexicon developers, including expert developers and linguistic informants. The expert developers who are also responsible for the grammar development can focus on tasks that are more complex. These include the definition of set of attributes and their allowed values for each lexical category.

When these are in place, the linguistic informants can concentrate on introducing and modifying lexical entries on instance field. The fact that the possible set of allowed attributes and their values are ready in the place reduces the common errors such as orthographic mistakes and entry of incorrect values. Protégé also automatically controls and validates the entered instances. If no value is introduced for an attribute, the corresponding field of the instance form is outlined in red. The same for the values that violate the attribute conditions defined in the slot form. This on-line validation feature helps non-experts port lexica into new languages, while offering extended functionality to experienced developers, which would have otherwise be available through macro hierarchies.

The next section shows how Protégé is used for construction of multilingual lexica.

4. Multilingual inheritance lexicon

Inheritance hierarchies are commonly used in both monolingual and multilingual lexica as a way to capture generalizations about languages (for example Briscoe, de Paiva & Copestake, 1993; Cahill & Gazdar, 1999). The Regulus lexica typically use macros for this purpose (inside of a language). Here we show how we apply an inheritance-based approach for a multilingual MedSLT lexicon using Protégé.

Protégé allows the modeling of knowledge-domains. We make use of this feature to model a domain specific lexical entry hierarchy for MedSLT system. We do that in multilingual fashion so that the features can be shared between different types of languages. The general principle is that the hierarchy is implemented as a top-down inheritance hierarchy where siblings inherit information from parent nodes. The classes cannot inherit from different parent nodes. In consequence, the language independent information that is inherited by all the languages is stated always at higher point in the hierarchy. The language specific information is stated on the lowest level.

At the top of this MedSLT headache domain lexical hierarchy is the '*Lexical entry*' class (Figure 3). This contains the information that applies to all words of languages (here illustrated with English, Finnish, and Japanese). The information of this class is directly inherited by all its subclasses. These represent the basic lexical categories such as '*noun'*, *'verb'*, and *'adjective'*. The attributes that are common to categories of all these languages (like `sem` and `entry_type`) are introduced at this level.

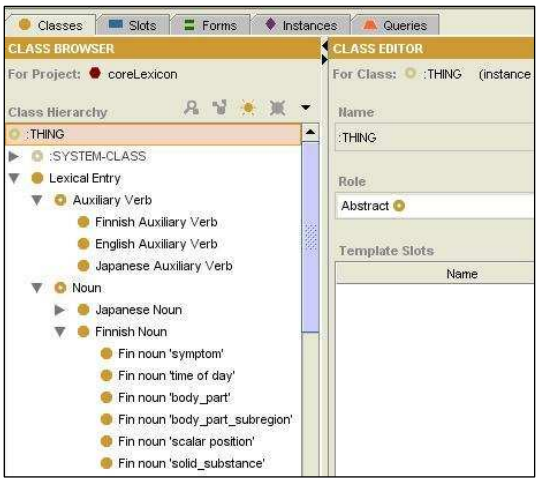

Figure 3: Multilingual domain specific lexical hierarchy.

Each of the lexical categories has one or several subclasses. These are defined based on the domain-specific syntactico-semantic features. These replace the Regulus (multilevel) macros illustrated in Section 2. For instance the verbs 'sleep' and 'walk' could here be classified in the same verb subclass. In this medical context these verbs are not only both intransitive verbs that take a 'person' as subject but in this particular medical context they both can express the 'cause' or 'relief' of the pain. These verbs act similarly in this context and thus share the same set of attribute-value pairs.

This sharing of constraints is not only limited within one language. Experience with the MedSLT system has shown that on restricted domains and context the entries of the same lexical category share features over languages borders (Bouillon et al, 2007). We can actually capture the generalizations about the different languages on the specific domain and share efficiently the linguistic information required by the system's lexical entries.

The benefits of this multilingual inheritance approach include the reduced redundancy of information within and between languages. The parallel hierarchic structure for different languages allows also to better detect the gaps and to keep the coverage consistent between the different system languages. Furthermore a new system language can be introduced in the multilingual lexicon simply by following the existing domain specific inheritance hierarchy and adding the required language specific classes and features.

5. Summary

We have described a lexicon development and management tool based on the Protégé platform that allows developers to address common shortcomings of declarative manual lexicon development approach.

The main difficulties of this approach are the required synchronization and mapping efforts for keeping the set of attributes and their allowed values consistent across languages, and complex macro hierarchies that are difficult to be developed and maintained by non-experts.

We have demonstrated how a centralized feature-value repository eliminates repetition and inconsistency of representations throughout the languages. While this could be equally implemented with the file-based declarative approach, the visual tool is preferable because of the user-friendly representation.

The described approach maintains the flexibility of the declarative approach, where expert developers define the lexical classes and required set of feature-value pairs, and further modify and remove them. This tool-based approach also simplifies the task for linguistic informants without prior exposure to language engineering allowing them to easily populate and maintain lexica.

Furthermore, this approach opens the door to the possibility of integration of already existing knowledge bases, which the knowledge-representation community develops within the same framework. For medical systems in particular, this approach could permit integration of built-in reasoning and dialog enhancements using readily available medical ontologies.

6. References

Baldwin, T. (2005). Bootstrapping Deep Lexical Resources: Resources for Courses. In *Proceedings of the ACL-SIGLEX 2005 Workshop on Deep Lexical Acquisition*, Ann Arbor, USA, pp. 67–76.

Bouillon P., Rayner M., Novellas Vall Br., Starlander M., Santaholma M., Nakao Y. & Chatzichrisafis N. (2007). Une grammaire partagée multi-tâche pour le traitement de la parole : application aux langues romanes. *In TAL (Traitement Automatique des Langues), Volume 47, 3/2006*, Hermes & Lavoisier.

Bouillon, P., Rayner, M., Chatzichrisafis, N., Hockey, B.A., Santaholma, M., Starlander, M., Isahara, H., Kanzaki, K. & Nakao, Y. (2005). A Generic Multi-Lingual Open Source Platform for Limited-Domain Medical Speech Translation. In *Proceedings of the Tenth Conference on European Association of Machine Translation*, Budapest, Hungary.

Briscoe T., Paiva, de V. and Copestake, A. (Eds.)(1993). *Inheritance, defaults and lexicon*. Cambridge University Press, Cambridge.

Cahill, L. J. and Gazdar, G. (1999). The Polylex architecture: Multilingual lexicons for related languages. *T.A.L.*, vol 40:1, pp.7-25.

Chatzichrisafis, N. and Santaholma, M. (2007). *Regulus Protégé Exporter. Manual, http://regulus.cvs.sourceforge.net/regulus/Regulus/doc/protege-regulus-exporter/regulusProtegePlugin.html?view=markup as of March 2008.*

Korhonen, A. (2002). Subcategorization Acquisition. Ph.D thesis, University of Cambridge.

MedSLT (2008) https://sourceforge.net/projects/medslt/. As of March 2008.

Protégé (2008). http://protege.stanford.edu. As of March 2008.

Rayner, M., Hockey, B.A. and Bouillon, P. (2006). *Regulus. Putting Linguistics into Speech recognition*. Stanford University Center for the Study of language and information, Stanford, California.

Regulus (2008). https://sourceforge.net/projects/regulus/. As of March 2008.

Santaholma, M. (2007). Grammar sharing techniques for rule-based multilingual NLP systems. In *Proceedings of NODALIDA 2007*, the 16th Nordic Conference of Computational Linguistics, Tartu, Estonia.