LexOnto: A Model for Ontology Lexicons for Ontology-based NLP

P. Cimiano[1], P. Haase[1], M. Herold[1], M. Mantel[1], P. Buitelaar[2]

[1]Institute AIFB, Universität Karlsruhe, Germany
{pci,pha}@aifb.uni-karlsruhe.de
matthias.mantel@gmx.net, mattonline@mac.de
[2]DFKI, Stuhlsatzenhausweg 3, Saarbrücken, Germany
paulb@dfki.de

Abstract. In this paper we present a model and a corresponding ontology for associating lexical information to entities of a given domain ontology. In particular, we focus on representing subcategorization frames as well as their mapping to ontological structures. The lexicon model and ontology can thus be used to specify the syntax-semantic interface for ontology-based NLP systems which are expected to produce output compliant with a specific domain ontology. Our lexicon ontology itself is formalized in the OWL language, thus allowing the lexica to be published together with the domain ontologies. We also discuss different ways how the creation of appropriate lexica can be supported.

1 Introduction

For many applications in a specific domain, NLP systems are required to produce output structures compliant with a specific domain ontology. In such cases, NLP systems need to "speak the language" of the underlying information system which stores the data in a knowledge base. In technical terms, the NLP system needs to produce structures which are compliant with the ontology on the basis of which the information system in question has been built. For this purpose, any ontology-based NLP system requires a lexicon which maps expressions in natural language to corresponding ontology structures. In this article we are concerned with the design of a rich lexicon model which allows for declaratively representing the mapping between language and a given domain ontology. Our view of ontology-based NLP is thus in line with the one of "Ontological Semantics" [1] in the sense that NLP systems are expected to produce ontological structures which have a well-defined meaning axiomatized by an underlying logical theory. An important requirement in our view is that these lexica are not specified in some proprietary form but in such a way that they can be (i) published together with the domain ontologies and (ii) reused across systems. Such interoperability can be achieved if lexica are described using ontologies and resorting to standard ontology languages such as OWL [2]. A further requirement on such a model is that it allows for the specification of more complex mappings than simple associations between nouns and classes as well as between verbs and properties.

For the representation of such simple mappings, the RDF(S) vocabulary [3] will certainly be enough. However, we need a model which allows to define more complex structures as lexicalizations for concepts and properties. In this paper we describe a lexicon ontology formalized in the OWL language which fulfills both requirements.

The paper is structured as follows: In Section 2 we first describe the lexicon ontology. Further, we discuss how the lexica can be actually created and maintained by advanced users relying on the FrameMapper tool we have presented earlier (see [4]) as well as by less experienced users relying on a wiki-based lexicon engineering tool. Both alternatives are discussed in Section 3. We also briefly describe how the lexicon ontology and the tools for lexicon engineering are used in the context of our ontology-based natural language interface ORAKEL, which has been described already in [4]. Before concluding, we discuss some related work in Section 4.

2 A Lexicon Model for Ontologies

In this section we present our lexicon ontology[1]. A major design criterion was to enable a tight integration of the lexical information with domain ontologies formalized in the Web Ontology (OWL) [2], but at the same time to ensure a clear separation between the ontological information in the knowledge base and the lexical information. To achieve this goal, we formalized the lexicon ontology itself in OWL. To relate the elements in the lexical ontology with the elements in the original domain ontology, we rely on a meta-ontology of OWL[2] that allows us to assert statements about the elements of an OWL ontology (c.f. [5])[3].

The main elements of our lexicon ontology are shown in Figure 1. In our lexicon model, we consider three different parts of speech: verbs, nouns and adjectives[4]. Overall, the lexicon model essentially represents open class words while closed class words (prepositions, determiners, pronouns etc.) are assumed to have a relatively fixed meaning across domains (compare [6]) and thus do not need to be represented in the domain lexicon. Prepositions are accounted for only indirectly through prepositional objects subcategorized by certain verbs.

In the most simple case, the relationship between `Lexemes` and `OntologyElements` (`classes`, `datatype` and `object properties`) can be realized using a direct relation `lexicalizes`. For example, a `Noun` may be used to lexicalize a particular `OWLClass`. However, in our model we are not particularly interested in simple nouns which map to a single class. While such a mapping is definitely necessary, it is relatively straightforward. However, it is not as straightforward if we want to model the morphological decomposition of the noun as in the LingInfo model [7].

[1] It can be downloaded at http://orakel.ontoware.org/LexOnto
[2] http://owlodm.ontoware.org/OWL1.0
[3] The idea is very similar to that of meta-modeling and meta-views in OWL1.1.
[4] While participles are currently considered, we do not discuss their treatment in this article.

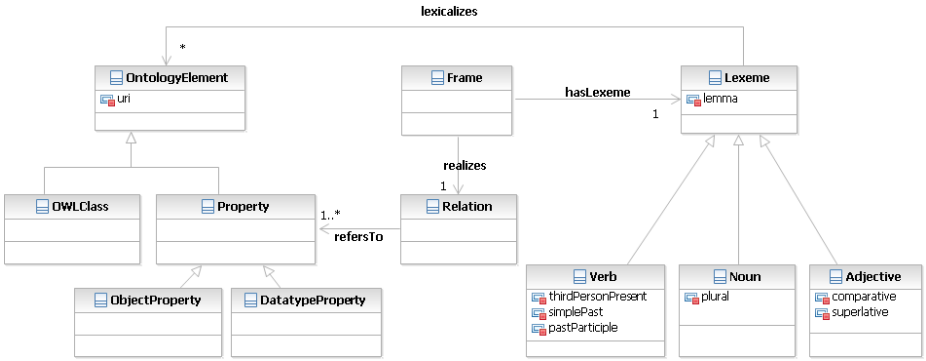

Fig. 1. Main Classes

In fact, as we will discuss further below, even for nouns, lexicalizations may be much more complex.

To describe more expressive lexicalizations, our lexicon ontology builds on the central notion of a *subcategorization frame*, i.e. linguistic predicate-argument structure. A basic distinction in the ontology is thus one between the class `Frame`, which represents subcategorization frames of different types, as well as the class `Relation`, which represents properties or joins between properties defined in the ontology. The abstraction `Relation` is necessary as it is not the case that linguistic argument structure can always be mapped to one (binary) property as defined in the OWL language. For sure, this does not hold for subcategorization frames with more than two arguments (e.g. transitive verbs with a prepositional complement). The class `Relation` is thus also introduced to represent joins between simple binary relations. For each (instantiated) `Frame`, we need to specify to which `Relation` in the ontology it refers to as well as how its arguments (e.g. its subject or object) map to the relation's arguments. Thus, there is a relation `refersTo` between a `Relation` (which can also represent a join between different properties) and a `Property` of the ontology. The chain `Frame` $\leftrightarrow$ `realizes` `Relation` $\leftrightarrow$ `refersTo` $\leftrightarrow$ `Property` allows to associate arbitrary complex lexical structures to properties in the ontology, in our case the subcategorization frames modeled as `Frame`s. This is the main mechanism which goes strictly beyond more simple models such as the one inherent in the RDF(S) model (compare Section 4). Finally, the lexical grounding with a specific `Lexeme` for a certain `Frame` is specified via the `hasLexeme` property.

In Figure 2 we show the various subclasses of `Frame`. The intention of these subclasses should be clear for the linguistically minded reader. For the sake of completeness, we briefly describe them. `TransitiveFrame` stands for a transitive verb (e.g. *love*) with the arguments subject and object, while `Intransitive_PP_Frame` represents an intransitive verb[5] with a prepositional complement as argument (e.g. *flow through*). `Transitive_PP_Frame` models a

[5] Intransitive verbs do not feature a direct object.

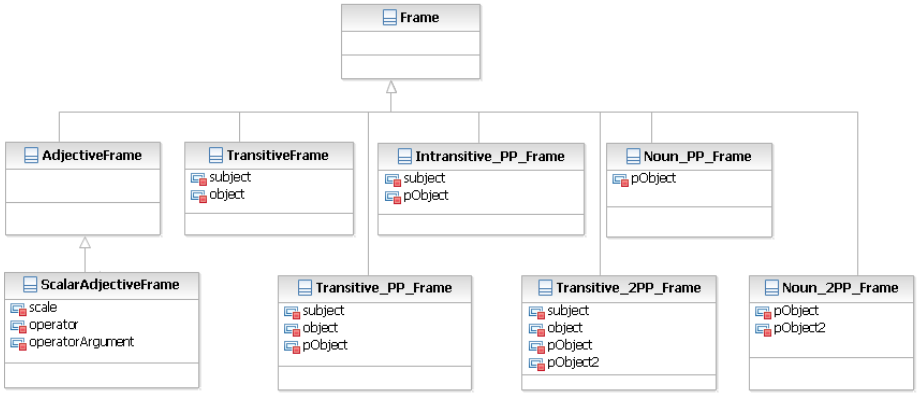

Fig. 2. Subclasses of Frame

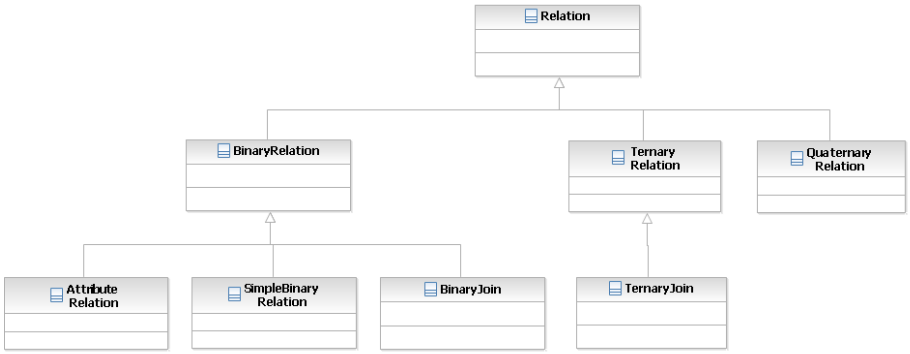

Fig. 3. Relations

subcategorization frame for a transitive verb with an additional prepositional complement (e.g. *move X to Y*), `Noun_PP_Frame` represents a noun with a prepositional complement (e.g. *capital of*) and `Noun_2PP_Frame` one with two prepositional complements (e.g. *distance from A to B*). We give illustrative examples for some of these classes below. Obviously, all these subclasses inherit the constraint that they should refer to exactly one instance of `Relation`.

Finally, in Figure 3 we show the various subclasses of `Relation`. For example, a `SimpleBinaryRelation` simply points to some `Property` defined in the metaontology of the corresponding domain ontology formalized in OWL via the `refersTo` property. Other structures (e.g. the `BinaryJoin`) point to more complex structures which we do not discuss in more detail here.

Figure 4 shows the modeling patterns for transitive verbs as well as nouns with a prepositional complement. They are both realized through a `BinaryRelation` and are associated through the `hasLexeme` property to a verb and noun, respectively. Further, the various grammatical roles `subject`, `object`

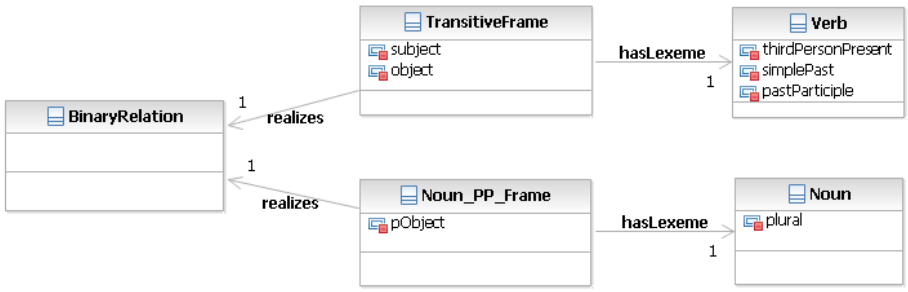

Fig. 4. Transitive and Noun_PP Frames

and `pObject` map to a String datatype restricted to the values *"domain"* and *"range"*.

The reason for applying this modeling pattern to nouns is that a simple mapping to a class (in the meta-ontology) is not sufficient in general. Nouns can be relational and feature prepositional complements. A relational noun is for example *capital* which does not denote a class but a relation between a country or state and the city associated with its government. In our work, we are thus mainly interested in nouns which subcategorize for prepositional complements, which can be modeled through the pattern shown in Figure 4. From these explanations, it is easy to extrapolate how other verb frames or a noun with two prepositional complements is modeled. Essentially, they have merely more arguments.

The third part of speech we consider in our lexicon model are adjectives. Adjectives are very interesting in the sense that they can map to very different ontological structures. Adjectives can be scalar adjectives and map to some property denoting a scale (typically a datatype property with an integer as range). This is for example the case for the adjective *long* which maps to a datatype property `height` with the range `integer` in the ontology. Second, adjectives can directly map to a class, i.e. a monadic predicate. This is for example the case for the adjective *German* which could map to a class `German` in some ontology. Third, an adjective could also map to a datatype property with a specific (string) value. For example. the adjective *German* could map to a property `nationality` with the value *"Germany"*. Possibly, other interpretations are conceivable, but we have given here what in our view are the main alternatives. Overall, it is important to emphasize that according to our ontology-based NLP view, adjectives (and other linguistic elements) get an interpretation by specifying their meaning in terms of classes and properties defined in the ontology. The formalization of the semantics of an adjective will ultimately depend on the way a certain phenomenon is modeled in the domain ontology. For example, considering the adjective *red* as an example, it could either map to a class `Red`, to a datatype property with a (string) value *"red"* or to an object property pointing to an instance `red`. Of course, this depends on the purposes and the granularity of the ontology in question. But in our lexicon model we should ideally cover all

the different possible interpretations. However, we currently only model scalar adjectives which are mapped to a datatype property with an integer as range. For this kind of adjectives, we need to specify whether the underlying scale measured is positive or negative. An example of a positive adjective which maps to the datatype property `length(object,integer)` is for example *long* and it is positive in the sense that the higher the integer value, the *longer* the object is. This is the other way round for an adjective with a negative scale such as *short*, which can me mapped to the same property `length` but with a negative scale. The scale is modeled via a datatype property `scale` restricted to the values *"positive"* and *"negative"* (compare Figure 2). Moreover, in order to be able to interpret the meaning of adjectives in base form, we assume that some threshold value is fixed which needs to be surpassed such that the property denoted by the adjective holds. This is modeled through datatype properties `operator`, which is restricted to the values $>$, $<$ and $=$, as well as `operatorArgument`, which has some datatype as range.

We have suggested above that adjectives can be treated as predicates with a delimited boundary. However, it is clear that such a treatment is problematic and can be seen as an approximation at best. The alternative of modeling the semantics of an adjective as a fuzzy set [8] is certainly appealing. However, it merely shifts the problem of determining a crisp extension to another level. While it would be nice to actually have a degree for the *bigness* of things, we would dare to claim that users, when asking a question such as *"Show me a list of big cities"* would not be particularly satisfied by a list of cities ranked by their degree of *bigness*, but would rather prefer a crisp set as results. Thus, in case we specify the semantics of an adjective as a non-crisp fuzzy function, we would still need to decide starting from which membership degree we will consider something as big for the purpose of returning answers to the above query.

In order to illustrate the above notions, we give some examples involving three properties from a geographical knowledge base[6]: `borders(location,location)`, `flows_through(river,city)`, `located_at_highway(city,highway)` and `inhabitants(city,integer)`.

The transitive verb *border* (represented as `Transitive_Frame`) could map to the `SimpleBinaryRelation` `borders` where the subject maps to the domain and the object to the range of the `border` relation.

The intransitive verb *flow* with a prepositional complement introduced by the preposition *through* (`Intransitive_PP_Frame`) could be mapped to a `SimpleBinaryRelation` pointing to the property `flows_through` of the ontology, whereby the subject maps to the domain and the prepositional object to the range position.

However, the order of arguments in a subcategorization frame does not always correspond to the order of arguments in the simple relation. This is for example the case for the verb *passes* with the preposition *through* (`Intransitive_PP_Frame`), for which we would specify that the subject maps

[6] to be downloaded at http://www.aifb.uni-karlsruhe.de/WBS/pci/orakel/

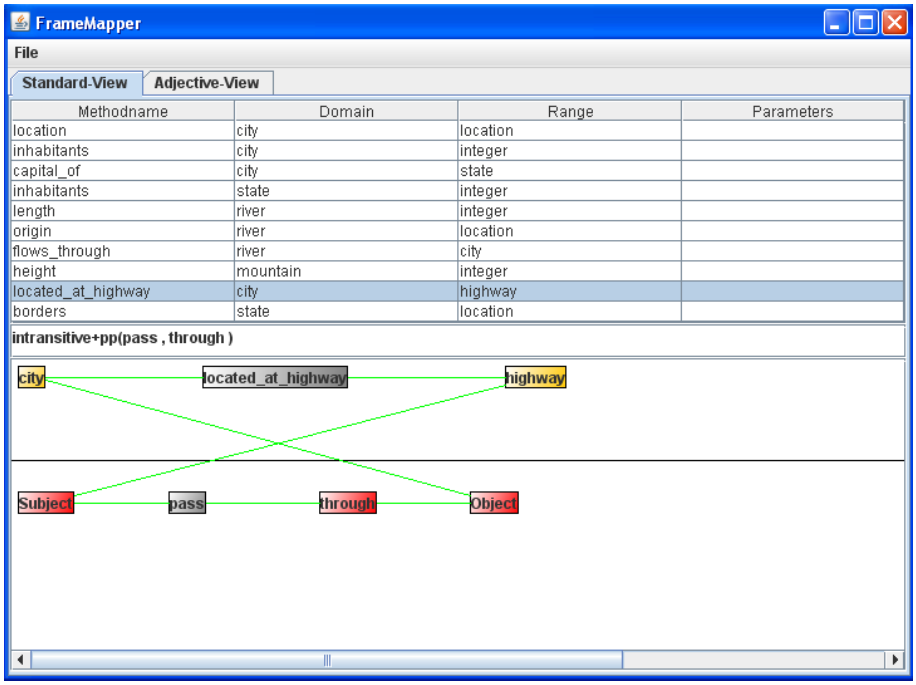

Fig. 5. GUI of FrameMapper showing a simple mapping for the geographical domain.

to the range of the property `located_at_highway` and the prepositional object to the domain of this property.

Finally, we give an example for the `Scalar_Adjective_Frame` *big*, which can be mapped to the `AttributeRelation` `inhabitants`. We would need to specify that the `scale` of the adjective is *positive* and that the threshold value for a city to be *big* is, say, 500,000 inhabitants. This can be done using the `operator` and `operatorArgument` datatype properties (compare Figure 2). In the corresponding instance of an `Adjective Lexeme`, we would specify that the base form of the adjective is *big*, the comparative is *bigger* and the superlative is *biggest*.

3 Lexicon Engineering and Application

In this section we discuss different ways how a lexicon for an ontology can be generated according to our model. On the one hand, we have designed a graphical tool called *FrameMapper* which can be used to create subcategorization frames and map these to properties (or joins) of these available in the domain ontology.

Figure 5 shows a screenshot of FrameMapper's graphical user interface. The figure shows the three main panes of *FrameMapper*. In the top pane, the user sees the relations specified in the ontology. In the second pane, s/he can see the different subcategorization frames assigned to the active relation. In the third pane, the user sees a graph visualization of the current subcategorization frame

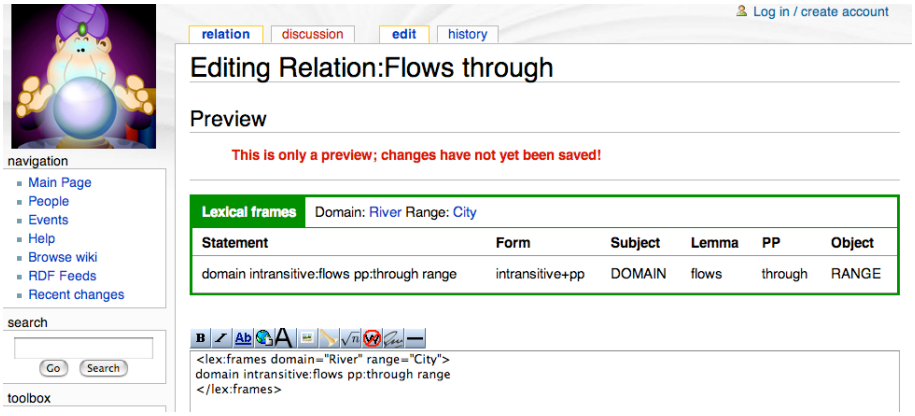

Fig. 6. Developing the lexicon using a wiki environment

and of the selected relations. They can then graphically map the arguments of the frame to the ones of the selected relation(s). In Figure 5, the user has for example selected the relation `located_at_highway(city,highway)` and created an `Intransitive_PP_Frame` for the verb *pass* which subcategorizes a subject and a prepositional complement introduced by the preposition *through*. The user has then mapped the subject position to the range position and the object to the domain position of the `located_at_highway` property.

Though we do not show this in more detail, the lexicon engineer can also use FrameMapper to create joins between relations to create mappings. Further, there is also an adjective view which allows for the creation of `Scalar_Adjective_Frame`s by specifying whether the underlying scale is *positive* or *negative*, specifying the datatype property it refers to, the lemma as well as the comparative and superlative forms of the adjective.

For less expert users, there is also the possibility of engineering the lexicon using a wiki environment, which in addition fosters the collaborative development of lexica. For this purpose we have designed a simple and restricted syntax which allows to create mappings. Assuming the user would like to express the above mentioned mapping for the `flows_through` relation, s/he would have to write down the following in the wiki page for the `flows_through` relation:

```
<lex:frames domain="River" range ="City">
domain intransitive:flow pp:through range
</lex:frames>
```

Figure 6 shows a screenshot of the wiki and how the mapping is rendered graphically.

The lexicon model described above has been applied in the context of the ORAKEL system (compare [4]). ORAKEL is a natural language interface to knowledge bases which translates users' natural language questions into a formal query which can be evaluated by an inference engine with respect to the knowl-

edge base in question. For this purpose, it relies on a lexicon model as specified above. From a specific lexicon, ORAKEL generates a complete domain-specific lexicalized grammar which realizes the syntax-semantics interface. In particular, the syntactic formalism used in ORAKEL are Logical Description Grammars (LDG) [9], a formalism which is very close to LTAG [10]. Semantic construction is performed compositionally on the basis of the lambda calculus. Thus, the subcategorization frames in the domain lexicon can be used to generate complete families of lexicalized trees featuring the correct semantic representations formulated in the lambda calculus using the classes and properties defined in the corresponding domain ontology. We refer the interested reader to ORAKEL's technical report for details [11].

In order to process lexica in the FrameMapper tool and in ORAKEL, we rely on the KAON2 knowledge repository which allows to store and query OWL ontologies[7]. In order to be able to "talk about" ontology elements (classes, properties etc.) in our lexicon model, we rely on the metamodeling framework described in [5]. Finally, the implementation of the WIKI interface has been performed on the basis of Semantic MediaWiki [12].

4 Discussion and Related Work

There has been already previous work on modeling lexical information for ontologies. On the one hand, RDFs [3] already allows to define labels for classes using the `rdf:label` property. However, the range of the `rdf:label` class is defined to be `Literal` and this considerably restricts the complexity of the structures that can be attached to classes as lexicalization. Another proposal for representing lexical information is SKOS [13]. The conceptual problem we see with SKOS is that it mixes linguistic and semantic knowledge. SKOS uses skos:broader and skos:narrower to express semantic relations without clearly stating the semantics of these relations intentionally, and defines the subproperties `skos:broaderGeneric` and `skos:narrowerGeneric` to have class subsumption semantics (i.e., they inherit the rdfs:subClassOf semantics from RDFS). We clearly keep the linguistic and semantic, ontology-based knowledge representations independent.

To our knowledge, the most elaborated current proposal to attach lexical information to ontologies is LingInfo [7]. LingInfo allows for the specification of lexical information for classes via so called metaclasses of type `ClassWithFeats` and to properties via the metaclass of type `PropertyWithFeats`. Via these classes, linguistic features `LingFeat` with respect to some language can be associated to classes and properties. These linguistic features include a morphosyntactic decomposition (`morphoSyntDecomp`) as property which points to phrases or word forms (`PhraseOrWordForm`). While `Phrases` point to complex noun or verb phrases, the `WordForms` represent part of speech types such as nouns, verbs, adjectives etc. For a certain word form we can specify gender and number features as well as the morphemes it is composed of (see [7] for more details).

[7] http://kaon2.semanticweb.org/

Our approach is similar to LingInfo in the sense that it strictly separates domain knowledge from lexical knowledge (in contrast to RDFS labels or SKOS). LingInfo provides a rich model how word forms or phrases can be associated to classes and properties, going down to the morphological level. Our approach is complementary in that it aims at associating higher-order lexical structures to classes and properties. In particular, our focus has been on designing a model in which subcategorization frames can be mapped to ontological structures. We also consider more complex structures than LingInfo in the sense that we also model joins of relations. Both models are in our view fairly elaborated and complementary, such that further work will aim at unifying both approaches into one combined model for representing lexical information for ontologies.

Also related to our approach is the SIMPLE lexicon model [14], which describes lexical entries also in terms of their semantic properties. The basic unit of the SIMPLE lexicon model are so called Semantic Units (SemUs). Each Semantic Unit gets assigned a semantic type from the SIMPLE ontology. In addition, the lexicon models the following additional information such as domain information, a lexicographic gloss, the argument structure for predicative SemUs, selectional restrictions on the arguments, the type of the event in question, a linking of the arguments to the syntactic subcategorization frames, an extended qualia structure entry, information about regular polysemous alternation into which a word sense may enter, cross-part-of-speech relations as well as synonyms. One of the crucial differences between SIMPLE and our ontology is that our lexicon model is assumed to be independent of any specific ontology, allowing to specify mappings from linguistic structure to an arbitrary ontology. SIMPLE, however, specifies the semantic types of SemUs with respect to a fixed ontology.

While we share the aim of making meaning explicit with respect to an ontology with the Ontological Semantics framework of Nirenburg and Raskin [1], our focus has been on creating a mechanism which allows to associate a lexicon with an arbitrary OWL ontology. Further, let us compare our treatment of adjectives with the one in the framework of Text Meaning Representation (TMR) / Ontological Semantics. Nirenburg and Raskin model the semantics of (scalar) adjectives in a similar way as in our approach, i.e. as denoting the value of some property of the modified noun. In particular, the semantics is essentially a function which is true in case the (normalized) value of the corresponding property for the modified is above or below a fixed threshold. For example, the semantics of *big* is a function which is true in case the value of the property denoting the size of the modified is over 0.75, whereby this value has to be understood as a normalized value which accounts for the fact that *big* can mean very different things depending on the property considered as well as the type of the modified noun, i.e. *big* can mean different things in absolute terms when comparing the sizes of elephants and mice. In our view, while using a normalized value is certainly elegant, at the very end it amounts to also specifying a relative but crisp boundary for when things are supposed to be regarded as big. Our model thus differs from this in that we require a user-specified absolute and crisp boundary for each adjective as well as a corresponding property of a specific concept it

refers. Thus, our model allows for defining completely different boundaries for one and the same adjective depending on the property and the corresponding concept. It is certainly an open question if this flexibility is indeed needed of if the semantic of scalar adjectives can be handled in a more generic way as proposed by Nirenburg and Raskin. In addition to scalar adjectives, Nirenburg and Raskin also discuss how to model the semantics of attitude adjectives (e.g. *good*), temporal adjectives (e.g. *occasional*), membership adjectives (e.g. *fake*), event-related adjectives (e.g. *abusive*) as well as denominal adjectives such as *medical*.

Finally, a few words seem necessary on the issue of how to deal with diathesis alternation. The answer is very simple, i.e. that diathesis alternations (and the changes in semantics they convey) are currently not accounted for in our proposals. Essentially, variants in the subcategorization structure of verbs are simply treated as different and unrelated entries (`Frames`) in the lexicon. Of course, the mapping to the ontology can be the same for different `Frames`, but these are then unrelated in the lexicon. However, in the case of diathesis alternation there is a direct relation between the different syntactic realizations of a verb and their semantics, which can not be accounted for in our approach. Modeling diathesis alternations would possibly lead to more compact lexicons, so it seems indeed an important issue for future research.

5 Conclusion

We have presented a model for attaching information about lexical realization to classes and in particular properties of a given domain ontology. In particular, our approach focuses on specifying how subcategorization frames map to complex ontological structures. We have presented our model in detail as well as described the tool support we have available so far for creating appropriate lexica. We have also discussed how our model is related to several other proposals in the literature, in particular to the LingInfo approach. Future work will aim at a tighter integration between our model and LingInfo. Finally, our model still needs a name for future reference. Let's call it *LexOnto*.

Acknowledgements This work has been partially funded by the German Ministry for Education and Research (BMBF) in the SmartWeb project as well as by the European Commission in the NeOn project (http://www.neon-project.org/) under grant number IST-2006-027595 and the X-Media project (http://www.x-media-project.org/) under grant number IST-FP6-026978. We would like to thank the anonymous reviewers for good and critical feedback as well as for pointing us to interesting and important issues for future work.

References

1. Nirenburg, S., Raskin, V.: Ontological Semantics. MIT Press (2004)

2. Bechhofer, S., van Harmelen, F., Hendler, J., Horrocks, I., McGuinees, D., Patel-Schneider, P., Stein, L.: OWL Web Ontology Language Reference. http://www.w3.org/TR/owl-ref (2004)
3. Brickley, D., Guha, R.: RDF Vocabulary Description Language 1.0: RDF Schema. Technical report, W3C (2002) W3C Working Draft. http://www.w3.org/TR/rdf-schema/.
4. Cimiano, P., Haase, P., Heizmann, J.: Porting natural language interfaces between domains – a case study with the ORAKEL system –. In: Proceedings of the International Conference on Intelligent User Interfaces (IUI). (2007) 180–189
5. Vrandecic, D., Völker, J., Haase, P., Tran, D., Cimiano, P.: A metamodel for annotations of ontology elements in OWL DL. In Sure, Y., Brockmans, S., eds.: Proceedings of the 2nd Workshop on Ontologies and Meta-Modeling. (2006)
6. Cimiano, P., Reyle, U.: Towards foundational semantics - ontological semantics revisited -. In: Proceedings of the International Conference on Formal Ontology in Information Systems (FOIS). Volume 150., IOS Press (2006) 51–62
7. Buitelaar, P., Sintek, M., Kiesel, M.: A lexicon model for multilingual/multimedia ontologies. In: Proceedings of the 3rd European Semantic Web Conference (ESWC06). (2006)
8. Zadeh, L.: The concept of a linguistic variable and its application to approximate reasoning. Information Sciences **8-9** (1975)
9. Muskens, R.: Talking about trees and truth-conditions. Journal of Logic, Language and Information **10** (2001) 417–455
10. Joshi, A., Schabes, Y.: Tree-adjoining grammars. In: Handbook of Formal Languages. Volume 3. Springer (1997) 69–124
11. Cimiano, P., Haase, P., Heizmann, J., Mantel, M.: Orakel: A portable natural language interface to knowledge bases. Technical report, Institut AIFB, Universität Karlsruhe (2007) http://www.aifb.uni-karlsruhe.de/WBS/pci/Publications/orakel_tech.pdf.
12. Krötzsch, M., Vrandecic, D., Völkel, M.: Semantic mediawiki. In Cruz, I., Tessaris, S., eds.: Proceedings of the 5th International Semantic Web Conference (ISWC06). Volume 4273 of Lecture Notes in Computer Science. (2006) 935–942
13. Miles, A., Brickley, D.: Skos core vocabulary specification. Technical report, W3C Working Draft (2005)
14. Lenci, A., Bell, N., Busa, F., Calzolari, N., Gola, E., Monachini, M., Ogonowsky, A., Peters, I., Peters, W., Ruimy, N., Villegas, M., Zampolli, A.: SIMPLE: A general framework for the development of multilignual lexicons. International Journal of Lexicography **13** (2000) 249–263