Harald Lüngen, Angelika Storrer

Domain ontologies and wordnets in OWL: Modelling options

1 Project framework and goals

Wordnets are lexical reference systems that follow the design principles of the Princeton WordNet project[1] (Fellbaum, 1998). Domain ontologies (or domain-specific ontologies such as GOLD[2], or the GENE Ontology[3]) represent knowledge about a specific domain in a format that supports automated reasoning about the objects in that domain and the relations between them (Erdmann, 2001). In this paper, we will discuss how the Web Ontology Language OWL can be used to represent and interrelate the entities and relations in both types of resources. Our special focus will be on the question, whether synsets should be modelled as individuals (we use *individual* and *instance* as synonyms and will refer to this option as *instance model*) or as classes (we will refer to this option as *class model*). We will present three OWL models, each of which offers different solutions to this question. These models were developed in the context of the research group "Text-technological Modelling of Information"[4] as a collaboration of the projects SemDok and HyTex. Since these projects are mainly concerned with German documents and with corpora that contain documents of a special technical or scientific domain, we used subsets of the German wordnet GermaNet (Kunze and Lemnitzer, 2002), henceforth referred to as GN, and the German domain ontology TermNet (Beißwenger et al., 2004), henceforth referred to as TN, to develop and evaluate the three models. To relate the general vocabulary of GN with the domain specific terms in TN, we developed an approach that was inspired by the plug-in model proposed by Magnini and Speranza (2002). In this approach, which has been developed in cooperation with the GermaNet research group (see Kunze et al. (2007) for details), we adapted the OWL model for the English Princeton WordNet suggested by van Assem et al. (2006) to GN, i.e. we modelled German synsets as instances of word-class-specific synset classes. For the reasons explained in section 3, we wanted to experiment with alternative models that implement the class model. In section 4 we will present three alternative OWL representations for GN and TN and discuss their benefits and drawbacks.

2 Basic entities and relations in GermaNet and TermNet

Wordnets and domain ontologies have been used in various applications of text processing (cf. Fellbaum, 1998; Kunze et al., 2003; Hirst, 2004, for an overview). Although

[1] http://wordnet.princeton.edu
[2] http://www.linguistics-ontology.org/gold.html
[3] http://www.geneontology.org/
[4] cf. http://www.text-technology.de

the Princeton WordNet was initially not conceived as an ontology but rather as a psychologically motivated model of lexical knowledge (Miller and Hristea, 2006, p.1), ontology textbooks often mention the Princeton WordNet as an ontological resource. (Sowa, 2000, p.497) distinguishes between *terminological ontologies*, the categories of which need not be fully specified by axioms and definitions, and *axiomatized ontologies*, the categories of which are distinguished by axioms and definitions stated in logic or in some computer-oriented language that could be automatically translated to logic. A similar distinction is drawn in Erdmann (2001), p.72: he differentiates between *light-weight ontologies*, which consist primarily of a representation schema providing means to specify taxonomies and to define additional features and relations, and *heavy-weight ontologies*, which are specified in a logic-based representation language. In this sense, PWN would be classified as a light-weight ontology; and PWN is, indeed, mentioned in the list of possible ontology resources (Erdmann, 2001, p.71).

Designing an OWL representation for a wordnet-style resource implies that one interprets the semantics of the entities and relations used in the original lexical resource with respect to the semantics of OWL.

In this interpretation process, the choice between one modelling option and the other is highly dependent upon the application context in which the ontology is to be used. The models discussed in section 4 have been developed with the following application framework in mind:

- The models are designed to be used in our research group's text processing applications, such as anaphora resolution (Goecke et al., this volume), discourse parsing (Bärenfänger et al., this volume), text-to-hypertext conversion (Holler et al., 2004; Storrer, 2008), and text classification (Mehler, this volume).

- Since some of these applications may deal with documents in a specific domain, we aim at a common representation format for domain-specific and general vocabulary.

For readers not familiar with wordnet-style lexical representations, the following paragraphs contain a brief introduction to the main types of entities and relations that have to be captured in our models.

The basic entities in GermaNet are disambiguated words, called *lexical units*.[5] Lexical units denoting the same or a very similar concept are grouped together in *synsets*, where the word *synset* is an abbreviation for *synonym set*. Lexical units and synsets are connected by two types of binary relationships: (1) *conceptual relations* like hyponymy and meronymy hold between synsets, and (2) *lexical-semantic relations* like antonymy hold between pairs of lexical units.

The basic entities in our domain ontology TermNet are technical terms, used to refer to well-defined concepts in the specialised domain. In many cases they form a taxonomy in which terms are represented as classes, and more specific terms are defined

[5] The OWL models of the Princeton WordNet discussed in section 3 use the term "word sense" for disambiguated words; the corresponding term in GermaNet is "lexical unit". Since our models are based on GermaNet data we will use the term "lexical unit" (abbreviated by LU) in this paper.

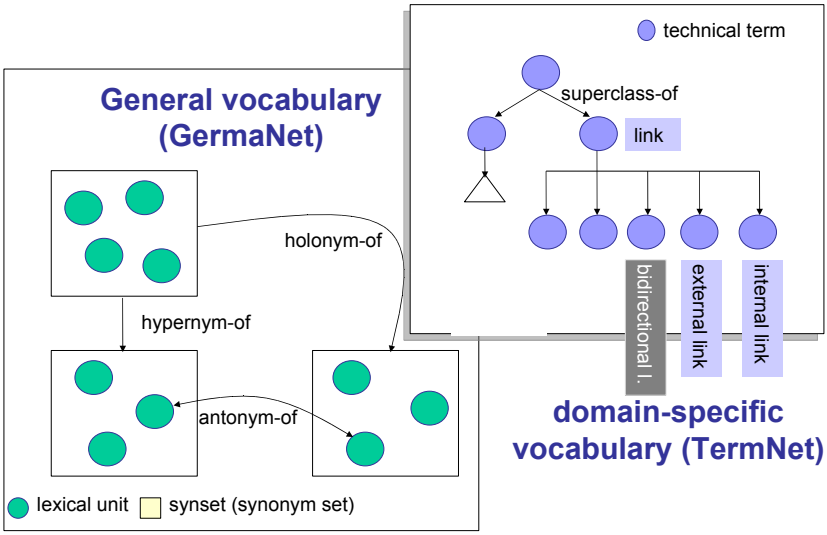

Figure 1: Entities and relations in GermaNet and TermNet.

as subclasses of broader terms. In the terminology of hypertext research, which is represented in TN for example, the technical terms *InternalLink*, *ExternalLink*, and *BidirectionalLink* are all subclasses of the broader term *Link*. Figure 1 illustrates these basic entities and relations in GN and TN.

3 OWL models for the Princeton WordNet

Although the Princeton WordNet was initially not conceived as an ontology, it has proven to be a useful resource for ontology-driven NLP applications. Following the Semantic Web initiative, several approaches for the conversion of the Princeton WordNet into OWL or RDFS have been put forward. We took three approaches that use OWL as their target representation language and examined them more closely: (1) The W3C approach ("W3C") (a working draft has been published by the Semantic Web Best Practices and Deployment Working Group, van Assem et al., 2006); (2) the "Neuchâtel approach" ("NCH") (Ciorăscu et al., 2003), and (3) the "Amsterdam approach" ("AMST") (van Assem et al., 2004). NCH has partly been considered in W3C, and the group of authors representing AMST overlaps with that of W3C; thus, AMST seems to be a predecessor of W3C. The three approaches differ in their goals: W3C aims at providing a standard conversion of the Princeton WordNet into OWL that can be used directly

by Semantic Web applications. In this approach the OWL version should not deviate from the original PWN, i.e. the PWN data model should be reflected in OWL without further interpretations. The goal of AMST was also to provide an OWL-encoded version of PWN. The main objective of NCH, though, was to create a test domain for the ontology-based information system *knOWLer* and to demonstrate its performance by means of sample queries in a document retrieval scenario. A discussion of modelling alternatives did not play a role in this effort. The W3C approach used version 2.0 of the Princeton WordNet, while NCH converted the version 1.7.1 of the Princeton WordNet in OWL.

In W3C and NCH, the actual ontology (i.e. the class hierarchy without the instances, cf. Erdmann, 2001, p.74), which is called the "WordNet RDF/OWL schema" in W3C, consists of the class of synsets (W3C: *Synset*, NCH: *LexicalConcept*) and its subclasses *Noun(Synset)*, *Adjective(Synset)*, *Adverb(Synset)*, and *Verb(Synset)*, where *Adjective(Synset)* has a further subclass called *AdjectiveSatellite(Synset)*. Lexical units are modelled by the class *WordSense* in W3C and by the class *WordObject* in NCH. In W3C, *WordSense* is further subdivided into part of speech-specific subclasses like *NounWordSense*; in NCH, it is not. Moreover, for purely formal units (not associated with a meaning), the class *Word* exists.[6] In NCH, a corresponding class called *StemObject* exists only in an external ontology which is used for document retrieval.

The single synsets – e.g. the synset {horse, nag, steed} – are modelled as individuals, i.e. as instances of *NounSynset*, *VerbSynset* etc., in all three approaches. Likewise, the single lexical units (e.g. *horse*) are modelled as individuals in W3C as well as in NCH. Consequently, the lexicalisation relation (the relation that connects synsets and lexical units) is an OWL `ObjectProperty` with the domain *Synset* (*LexicalConcept*) and with the range *WordSense* (*WordObject*; the relation is called *synsetContainsWordSense* in W3C and *wordForm* in NCH), thus connecting a synset individual to one or more lexical unit individuals. In AMST, lexical units are modelled neither as classes nor as individuals, but as literals which appear as values of the multiple-valued DatatypeProperty *wordForm* (domain: *Synset*). Furthermore, in all three approaches, further ObjectProperties with *Synset* as domain and range exist, which model the PWN conceptual relations (e.g. *hyponymOf*, *entails*, and partly their POS-specific restrictions) in OWL. In a similar fashion, the PWN lexical relations (e.g. *antonymOf*, *participleOf*) are represented as ObjectProperties *WordSense* in the OWL versions of W3C and NCH, with *WordSense* as domain and range. Moreover, the W3C approach contains instructions on how to interpret the PWN *hyponymOf* relation by declaring it a subproperty of the *subclassOf* property in OWL. In AMST, the lexical relations are encoded by dint of "helper classes" such as *SynSetVerb*.

Considering that the conversion mainly aims at preserving the original structure and providing an OWL representation that can be easily processed and integrated in SW applications, these models seem to be quite suitable.

[6] The homonymic lexical units *nArtefakt.248.Schloss* and *nOrt.862.Schloss* of GermaNet, for example, share certain formal (i.e. orthographic, phonological, and morphological) properties which could be represented as properties of one *Word* instance.

From a linguistic viewpoint, however, it is striking that all of the approaches model synsets, i.e. sets of quasi-synonymous units, and their members, the disambiguated lexical units, as individuals. This is striking because synsets are frequently considered to be concepts which can be referenced linguistically by the lexical units contained in the synsets; e.g. a synset formed by {horse, nag, steed} denotes the horse concept. This suggests that at least the synsets should be conceived as classes, the instances of which are individual objects (e.g. the horse "Fury"). However, in principle, the lexical unit "horse" can also generically refer to the whole class (e.g. in meaning postulates like "A sorrel is a reddish horse").

All in all, the decision to model synsets and lexical units as individuals, i.e. to implement the instance model, is not at all obvious. Instead, both options – for which we introduced the short terms *instance model* and *class model* in section 1 – capture two different perspectives that one may have on wordnets:

1. In the instance model, a wordnet is conceived primarily as a lexicon describing properties of lexical units. The categories of the model represent linguistic classes and subclasses. Thus, synsets and word senses are modelled as instances of word class categories, e.g. the classes *NounWordSense* or *NounSynset*.

2. In the class model, a wordnet is conceived primarily as an ontology describing properties of the concepts that are denoted by the lexical units. The categories captured in the ontology represent concepts and their properties. Thus, synsets and word senses are modelled as classes in which the instances are individual entities.

There are two arguments which motivate, in our view, the implementation of the class model:

1. The Princeton WordNet, in its version PWN 2.1, draws an explicit distinction between the relation of hyponymy on the one hand (e.g. the subordinate synset containing "peach" is a hyponym of the superordinate synset containing "drupe") and the class-instance relation on the other (e.g. the proper name "Berlin" is an instance of the synset containing "city").[7] Over 7,600 PWN synsets were manually classified as instances and tagged as such. Despite this introduction of the class-instance distinction, the PWN version 2.1 may still be converted to OWL using the instance model, e.g. by ignoring the class-instance distinction among synsets by skipping those synsets that are tagged as instances. However, Miller and Hristea (2006) (p.1) introduced this distinction with the aim to help ontologists "to distinguish between a concept-to-concept relation of subsumption and an individual-to-concept relation of instantiation". We believe that this aim implies that synsets (like "peach") are conceived as concepts denoting classes with numerous instances, while proper names (like "Berlin") denote instances of synset classes (in the case of "Berlin" the class synset containing "city"). In our opinion,

[7] Examples from (Miller and Hristea, 2006, p.3).

this perspective is captured more adequately by the class model than by the instance model, because when class synsets are already modelled as instances, the class-instance relations would have to be defined between pairs of instances; this is not a very intuitive interpretation of such relations.

2. The second argument is concerned with domain-specific vocabulary in domain ontologies. Domain ontologies often represent taxonomies of technical terms that mirror superclass-subclass-relations between their instances. This may be illustrated by the example in figure 1: the term *externalLink* is a subclass of the broader term *Link*. In the class model one may represent these relations using the <rdfs:subClassOf> property and benefit from its related mechanisms of feature inheritance. Another aspect that may be nicely captured by the class model is that in such taxonomies, subclasses with the same classification feature (e.g. *InternalLink* and *ExternalLink* in the example illustrated in figure 1) are disjoint: an individual link may either be an instance of *InternalLink* or an instance of *ExternalLink*. This restriction can be neatly represented using the OWL <owl:disjointWith> construct. Since <rdfs:subClassOf> and <owl:disjointWith> can only be defined for classes, the class model is better suited to represent taxonomies than the instance model. All in all, the class model seems to be more appropriate to capture domain-specific terminology in OWL than the instance model.

For our domain ontology TermNet, we developed an OWL model that implements the class model: the main entities of TN, the technical terms, are represented as classes. Specific terms are related to broader terms by means of the <rdfs:subClassOf> property. Disjointness of technical terms, e.g. between *internal link* and instance of *external link* in our example, is represented by using the OWL <owl:disjointWith> construct.[8]

If one chooses the class model for the domain ontology one still may follow the instance model when representing the general vocabulary of GermaNet. Indeed, in the approach described in Kunze et al. (2007) we related the class model of TN with an instance model of GN; this option will be described in section 4.1. In addition, we experimented with alternative models for GN: one that implements GN following the class model and one that combines both options in OWL Full. The three models as well as their respective combination with TermNet will be discussed and compared in the following section.

4 Three alternative models for representing GN and TN in OWL

In this section, we discuss three alternative representations of GN and its plug-in connections with TN in OWL: the first representation we call *The OWL DL Instance Model* (GN synsets and lexical units are OWL individuals), the second encoding we call *The OWL DL Class Model* (GN synsets and lexical units are classes), and the third

[8] The model is described in Kunze et al. (2007); the subset of TermNet considered in the model comprises 141 NounTerms from the domain of hypertext research.

encoding we call *The OWL Full Metaclass Model* (GN synsets and lexical units are both OWL classes *and* individuals).

For each model, a basic hierarchy of classes is declared using <owl:Class> and <rdfs:subClassOf> statements. The basic hierarchy includes *Synset* with its subclasses *NounSynset*, *AdjectiveSynset*, *VerbSynset*, and *AdverbSynset*, as well as the class *LexicalUnit* with its subclasses *NounUnit*, *AdjectiveUnit*, *VerbUnit*, and *AdverbUnit*, cf. also Kunze et al. (2007).

In each model, we define the general lexicalisation relation (describing the relation between one synset and its lexical units) as an OWL Object Property called *hasMember*. Listing 1 shows that this property has the general class *Synset* as its domain and the general class *LexicalUnit* as its range.

```
<owl:InverseFunctionalProperty rdf:about="#hasMember">
  <rdfs:range rdf:resource="#LexicalUnit"/>
  <rdfs:domain rdf:resource="#Synset"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  <owl:inverseOf rdf:resource="#memberOf"/>
</owl:InverseFunctionalProperty>
```

Listing 1: OWL code introducing the lexicalisation relation *hasMember* in all three models

Likewise in each model, the GN hyponym relation is an OWL Object Property called *isHyponymOf* and defined with *Synset* as both domain and range, cf. listing 2.

The POS-specific restrictions for *hasMember* and *isHyponymOf* are encoded in both models by use of the <owl:allValuesFrom> construction. Listing 3 illustrates how such restrictions are defined for the class*NounSynset*: the OWL code in this listing specifies the restriction of the range of the property *hasMember* to *NounUnit*, which is the POS-corresponding subclass of *LexicalUnit*.

In each model, we also wanted to relate GN synsets to terms of the domain ontology TermNet (TN). Since the OWL representation of TN remains constant (terms are always represented as OWL classes for reasons stated in section 3), each model implies a different way of encoding the plug-in relations between GN and TN.

In the following section, we will compare these three models and describe how they can be related to TermNet within OWL. This will be discussed by the examples of how the lexicalisation relation between individual synsets and lexical units, the hyponymy relation between individual synsets, and the plug-in relation *attachedToNearSynonym* between individual synsets in GN and terms in TN are encoded. Furthermore, for each model, we discuss our experiments with queries to the resulting knowledge bases formulated in Prolog and nRQL.

4.1 The OWL DL Instance Model

As discussed in section 3, previous approaches to the representation of the PWN in OWL conform to an instance model, where the single synsets are rendered as OWL

```
<owl:ObjectProperty rdf:about="#conceptualRelation">
  <rdfs:domain rdf:resource="#Synset"/>
  <rdfs:range rdf:resource="#Synset"/>
</owl:ObjectProperty>

<owl:TransitiveProperty rdf:about="#isHyponymOf">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  <rdfs:subPropertyOf rdf:resource="#conceptualRelation"/>
  <owl:inverseOf>
    <owl:TransitiveProperty rdf:about="#isHypernymOf"/>
  </owl:inverseOf>
</owl:TransitiveProperty>
```

Listing 2: OWL code introducing conceptual relations and the relation *isHyponymOf* in all three models

```
<owl:Class rdf:ID="NounSynset">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:allValuesFrom>
        <owl:Class rdf:about="#NounUnit"/>
      </owl:allValuesFrom>
      <owl:onProperty>
        <owl:InverseFunctionalProperty rdf:ID="hasMember"/>
      </owl:onProperty>
    </owl:Restriction>
  </rdfs:subClassOf>
  ...
</owl:Class>
```

Listing 3: OWL code for a restriction of the lexicalisation relation *hasMember* in all three models

individuals and where the conceptual and lexical-semantic relations are OWL property instances. The instance model for representing GermaNet in OWL, which adopted the W3C strategy (see section 3) for the representation of wordnets, was introduced in Kunze et al. (2007).

Relation instances of the *hasMember* relation are encoded as property instances of individual synsets. Listing 4 shows the XML code representing the individual that corresponds to the synset {eitel, selbstherrlich, selbstgefällig, selbstgerecht}.

A relation instance of *isHyponymOf* is also encoded as a property instance of an individual synset in XML, cf. listing 5.[9]

To check the consistency of the OWL DL Instance Model, the GermaNet ontology was populated with 37 synset and 83 lexical unit individuals and all relation instances that hold between them. The ontology was checked for consistency using the ontology editor Protégé 3.1.1[10] in connection with the reasoner/classifier software RacerPro 1.9.1[11].

[9] All characteristics of the OWL object and datatype properties as well as OWL code representing further lexical relations in the instance model are described in Kunze et al. (2007).

[10] http://protege.stanford.edu, visited 24 May 2007

```
<AdjectiveSynset rdf:ID="aVerhalten.235">
   <hasMember rdf:resource="#aVerhalten.235.eitel"/>
   <hasMember rdf:resource="#aVerhalten.235.selbstherrlich"/>
   <hasMember rdf:resource="#aVerhalten.235.selbstgefällig"/>
   <hasMember rdf:resource="#aVerhalten.235.selbstgerecht"/>
   <!-- ... (Further properties of aVerhalten.235] -->
<AdjectiveSynset>
```

Listing 4: OWL code for relation instances of *hasMember* in the Instance Model

```
<AdjectiveSynset rdf:ID="aVerhalten.235">
   <!-- ... (Further properties of aVerhalten.235] -->
   <isHyponymOf rdf:resource="#aVerhalten.225"/>
<AdjectiveSynset>
```

Listing 5: OWL code for relation instances of *isHyponymOf* in the Instance Model

To test how queries to the ontology can be formulated and processed, we parsed the owl file using the triple store SWI Prolog Semantic Web library[12] in combination with the Thea OWL library for Prolog (Vassiliadis, 2006). For the ontology, we subsequently implemented in Prolog several query types for the ontology, which are typical of text-technological applications:

- List the set of synsets that a given lexical unit is a member of
- List the set of lexical units a given synset has as its members
- List the set of synonyms of a given lexical unit
- List the set of direct hyponym (hypernym) synsets of a given synset
- List the set of direct hyponym (hypernym) lexical units of a given lexical unit
- List the set of direct or transitive hyponym (hypernym) synsets of a given synset
- List the set of direct or transitive hyponym (hypernym) lexical units of a given lexical unit

Queries of these types could be successfully run on the GN subset encoded as OWL DL Instance Model ontology.

In the approach described in Kunze et al. (2007), we related a subset of TN technical terms with a subset of GN synsets. Since that implies that an ontology which follows the instance model is related to an ontology which follows the class model, we call the approach to connect these a mixed model. We defined three plug-in relations called *attachedToNearSynonym*, *attachedToGeneralConcept*, and *attachedToHolonym* with domain *tn:Term* and range *gn:Synset*. Plug-ins are relations to connect the specialised

[11] http://www.racer-systems.com, visited 24 May 2007
[12] http://www.swi-prolog.org, visited 24 May 2007

concepts (concepts from domain-specific vocabulary or terminologies) of a domain ontology with the more general concepts of a PWN style lexical-semantic network. The original plug-in approach yields a common hierarchy in which the top concepts of the specialised ontology are eclipsed while the subordinate concepts, the terms, are imported into the general language ontology. The plug-ins defined in Kunze et al. (2007) are inspired by, but not identical to, the ones originally introduced in (cf. Magnini and Speranza, 2002). The plug-in relation *attachedToNearSynonym*, for example, is defined as an OWL Object Property in the mixed model with domain *tn:Term* and range *gn:Synset* as shown in listing 6. *Gn, tn*, and *plg* are namespace prefixes for the URIs of the three ontologies involved (GermaNet, TermNet, and plug-in relations), which are ideally kept in separate files.

```
<owl:ObjectProperty rdf:about="#plg:attachedToNearSynonym">
  <rdfs:domain rdf:resource="#tn:Term"/>
  <rdfs:range rdf:resource="#gn:Synset"/>
</owl:ObjectProperty>
```

Listing 6: OWL code for introducing the plug-in relation *attachedToNearSynonym* in the Instance Model

To plug the class *tn:Term_Link* into its corresponding synset *gn:Link*, the former is declared to be a subclass of a local restriction that assigns every individual of the class *tn:Term_Link* the individual *gn:Link* as the value on the property *plg:attachedToNearSynonym*, using the <owl:hasValue> construction (listing 7) (cf. also Kunze et al., 2007).

```
<owl:Class rdf:ID="tn:Term_Link">
  <rdfs:subClassOf rdf:resource="#tn:NounTerm"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:hasValue rdf:resource="#gn:Link"/>
      <owl:onProperty>
        <owl:ObjectProperty rdf:ID="plg:attachedToNearSynonym"/>
      </owl:onProperty>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

Listing 7: OWL code for relation instances of *attachedToNearSynonym* in the Instance Model

Since in this mixed model TermNet terms are modelled as classes and GermaNet synsets are modelled as instances and since in OWL DL classes cannot be specified as values of property instances, the plug-in relations proposed in Kunze et al. (2007) cannot have inverse properties, i.e. cannot be defined using the <owl:inverseOf> construction. A declaration of inverse relations is strictly speaking not necessary for making inferences, but it is still desirable because it can speed up processing.

4.2 The OWL DL Class Model

When modelling the conceptual relations between synsets according to a class model (synsets are OWL classes), our first preference would have been to relate classes by declaring pairs of them as relation instances of a conceptual relation like *isHyponymOf*. However, when classes are assigned as values of properties, they must function as individuals at the same time, which goes beyond the scope of OWL DL (cf. Smith et al., 2004). Thus, we decided to relate classes with one another by employing local property restrictions using the <owl:allValuesFrom> construction, such as in the example in listing 8, where the synset containing *Webdokument* is declared to be a hyponym of the synset containing *Hypertextsystem*. When a synset has more than one hypernym, we declare the <owl:allValuesFrom> restriction such that all values have to be taken from a *union* of classes.

```
<owl:Class rdf:about="#gn:Synset_Webdokument">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:allValuesFrom>
        <owl:Class>
          <owl:unionOf rdf:parseType="Collection">
            <owl:Class rdf:about="#gn:Synset_Dokument"/>
            <owl:Class rdf:about="#gn:Synset_Hypertextsystem"/>
          </owl:unionOf>
        </owl:Class>
      </owl:allValuesFrom>
      <owl:onProperty>
        <owl:TransitiveProperty rdf:about="#gn:isHyponymOf"/>
      </owl:onProperty>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:class>
```

Listing 8: OWL code for relation instances of *isHyponymOf* in the OWL DL Class Model

Within a class model, it is tempting to model lexical units as the instances of the synsets. However, we have indicated above that we want the instances of synsets to include individuals of a discourse model, such as *Berlin* (i.e. named entities) or *Horse_351*. Lexical units do not represent semantic units but linguistic expression types; thus, it is adequate to model lexical units as classes, too. (Their instances should represent the tokens (occurrences) of lexical units in text.) Accordingly, the relation instances of the lexicalisation relation *hasMember* and all of the lexical relations are encoded as local property restrictions on synset classes, too, cf. listing 9.

Again, we parsed the ontology using the Semantic Web library with the Thea OWL library for Prolog. To run the set of queries listed in section 4.1, a new set of Prolog predicates had to be implemented. Still, the queries could all be run in a straightforward manner, and Prolog provided the correct answer sets.

```
<owl:InverseFunctionalProperty rdf:about="#hasMember">
  <owl:inverseOf rdf:resource="#memberOf"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  <rdfs:range rdf:resource="#LexicalUnit"/>
  <rdfs:domain rdf:resource="#Synset"/>
</owl:InverseFunctionalProperty>
```

Listing 9: OWL code for relation instances of *hasMember* in the OWL DL Class Model

To test the consistency and general queryability of the combination of both GN and TN as class models in OWL, we also implemented a set of test queries using the query language nRQL (Haarslev et al., 2004) in connection with RacerPro. We tested the query types listed in section 4.1 as well as more complex queries across GN and TN, e.g. querying hyponymy also along the plug-in relation *attachedToGeneralConcept*. The results showed that, when using RacerPro, only information about (sets of) individuals can be queried, but not about classes as such, which would be necessary for a wordnet ontology in the class model. Thus, we had to first introduce one pseudo-individual for each synset and lexical unit. Consequently, the original class model became somewhat corrupted, but, unlike in the case of the SWI SemWeb Library, additional coding of query predicates was not required when using nRQL. We could formulate queries as listed in section 4.1 to the GN+TN Class Model represented in OWL.[13] Listing 10 shows nRQL code for a query for all hyponyms (TN terms or GN synsets) of the GN synset *Navigationshilfe*. RacerPro would infer the right answer, i.e. a list of synset IDs from both GermaNet and TermNet including those in listing 10.

Listing 10: nRQL query to Class Model

```
(retrieve (?y) (or (and (?x ?y |gn_isHyperonymOf|) (?x
  |gn_Synset_Navigationshilfe|)) (and (?x |gn_Synset_Navigationshilfe|)
  (?x ?z |gn_isHyperonymOf|) (?z ?y |plg_inverseOfAttachedToGeneralConcept|))))

((?Y |tn_ObjektiverLink|))
((?Y |tn_Eins-zu-n-Link|))
((?Y |tn_VerborgenerLink|))
((?Y |gn_SS_Link|)))
```

4.3 The OWL Full Metaclass Model

As already indicated, we tested a third modelling option for GermaNet in OWL. The Instance Model can be converted into a *Metaclass Model* simply by adding the following line to the definitions of the class *Synset*:[14]

[13] We would like to thank Bianca Selzam, who built the TermNet OWL model and conducted the query experiments using nRQL.

```
<rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Class"/>
```

This makes *Synset* a *metaclass*, i.e. a class in which the instances are classes, too. Accordingly, lexical units were declared as metaclasses. Thus, all of the single synsets and lexical units are simultaneously classes and instances in this ontology, which now seems to offer all advantages of the instance *and* the class model at the same time: since synsets and lexical units are instances, straightforward and simple XML element relation instances as in the instance model shown in section 4.1, are used to represent the conceptual and lexical-semantic relations, and the lexicalisation relation. On the other hand, since synsets and lexical units are classes, they can now be populated with token occurrence individuals as needed in text-technological applications, cf. figure 2. Interestingly, it seems that a metaclass mechanism for OWL could also be used in many other domains and applications, (cf. Schreiber, 2002; Noy (ed.), 2005).

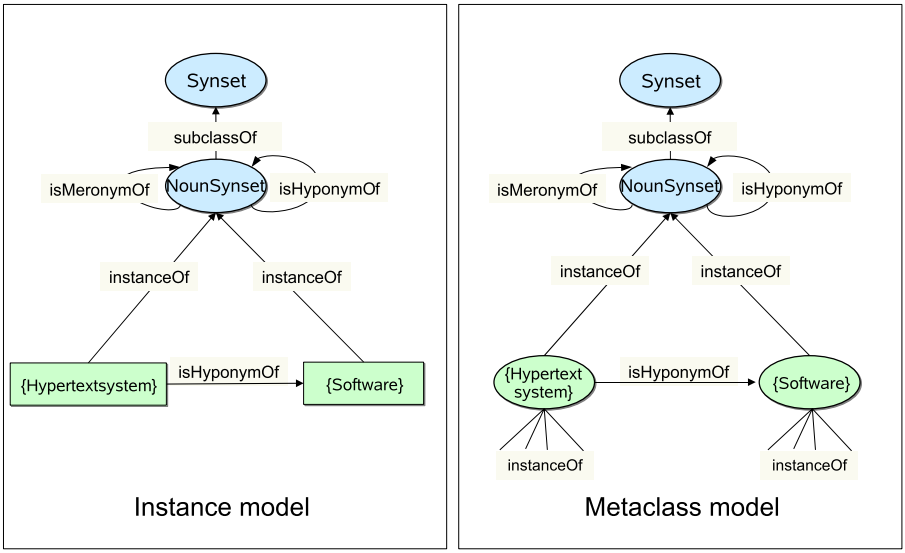

Figure 2: Instance vs. Metaclass Model.

Unfortunately the ontology described above lies outside the scope of OWL DL, i.e. it is in OWL Full (cf. Smith et al., 2004). Thus, standard DL-based reasoners cannot be used on it.[15]

[14] Line is shown as added using the Protégé editor.

[15] Outside the DL reasoning community, Pan et al. (2005) introduced a non-standard OWL variant called OWL FA. This is a well-defined *metamodelling* extension of OWL DL, and, unlike OWL Full, it is still decidable.

However, this metaclass model ontology of GermaNet in OWL Full could be parsed using the SWI Prolog Semantic Web library in combination with the Thea OWL library for Prolog and successfully queried with queries of the types listed in section 4.1. The queries had to be coded in Prolog and are basically the same as the ones one can code for the instance model. We implemented a set of queries in Prolog that can infer hyponym relationships in the OWL Full Metaclass representation of GN.[16] For querying the set of hyponyms of a given lexical unit, for example, the Prolog predicates shown in listing 11 were written: HasElements/2 and IsMemberof/2 are user-defined predicates that check the GN lexicalisation relation (*hasMember* and its inverse property *memberOf*, respectively), and owl_parser:uri_split/4 is a predicate provided by the Thea OWL library for deleting or adding a namespace URI. (For demonstration purposes, we provide a "readable" version of the predicate *unitIsHyperonymOf* here, i.e. one where the answer does not contain namespace URIs.) Listing 12 shows a query for the set of lexical unit hyponyms of the lexical unit *aVerhalten.235.eitel* and the response of the Prolog interpreter.

Listing 11: Prolog predicates for querying the Metaclass Model

```
unitIsHyperonymOf_readable(Input,Output) :-
        isMemberOf(Input,Set),
        setIsHyperonymHelper2(Set,Set2),
        hasElements(Set2,C),
        owl_parser:uri_split(C,_,Output,'#').

setIsHyperonymHelper2(Input,Output) :-
        individual(Input,_,_,PList),
        member(value('http://www.owl-ontologies.com/unnamed.owl#isHypernymOf',Output),
        PList).
```

Listing 12: Prolog query to Metaclass Model

```
?- unitIsHyperonymOf_readable('aVerhalten.235.eitel', A).

A = 'aVerhalten.231.geckenhaft' ;
A = 'aVerhalten.236.affektiert' ;
A = 'aVerhalten.236.geziert' ;
```

Further predicates that can be used for the hypernymy and hyponymy-related queries listed in section 4.1 are *unitIsDirectHyponymOf, setIsHyperonymOf* and *setIsDirectHyperonymOf*.

[16] We would like to thank Christian Kullmann, who implemented the query predicates and conducted the query experiments in Prolog using Thea.

5 Conclusions and outlook

Existing conversions of the Princeton WordNet into the Semantic Web ontology language OWL (Ciorăscu et al., 2003; van Assem et al., 2004, 2006) as well as a description to convert GermaNet to OWL (Kunze et al., 2007) apply what we have dubbed an instance model to the representation of wordnets in OWL: the single synsets and lexical units are rendered as OWL individuals, and relation instances of the conceptual and lexical-semantic relations appear as property instances. Nevertheless, from a linguistic point of view, synsets are concepts (classes) whose instances are individuals or discourse entities, and lexical units are types of linguistic expressions, whose instances can be interpreted as the token occurrences of these expressions in text, which seems appropriate at least in text-technological applications. Moreover, domain-specific ontologies or terminologies encoded in OWL such as TermNet (Kunze et al., 2007) or GOLD (Farrar and Langendoen, 2003) rather apply a *class model* of representation, in which terms are rendered as OWL classes, and a taxonomic hierarchy is imposed by means of the OWL *subClassOf* relation. In text-technological applications, it is often desirable to integrate such domain-specific ontologies with a general wordnet, and it would clearly ease the integration process if the two resource types corresponded to the same representation model.

Thus, in section 4, we described our evaluation of three OWL representation models for wordnets, using the example of GermaNet, and how they can be combined with the domain ontology TermNet, applying plug-in approach suggested by Magnini and Speranza (2002). Firstly, in the instance model of GN, conceptual and lexical-semantic relations as well as the lexicalisation relation appear as property instances, i.e. XML elements. Plug-in relations have as their domain a term class and as their values a GN individual synset. As a consequence, inverse relations of plug-ins could not be defined. Secondly, in the class model of GN, all conceptual and lexical-semantic relations, the lexicalisation relation, as well as plug-in relations were rendered as property restrictions on classes representing individual synsets and terms. Thirdly, in the metaclass model of GN, synsets and lexical units are both classes and individuals in OWL. All lexical-semantic and conceptual relations as well as the lexicalisation relation are thus XML property instances like in the instance model. Plug-in relations can either be represented as property instances like in the instance model or as local property restrictions like in the class model.

In our view, it would be clearly desirable to encode wordnets in a metaclass model in OWL: it allows for a linguistically adequate representation of synsets and lexical units as OWL classes while the somewhat clumsy representation of lexical-semantic and conceptual relations as local property restrictions, such as in the class model, are not necessary. Its only drawback is that DL-based reasoning software cannot be used on a metaclass model, because it is outside of the sublanguage OWL DL. As a possible way out, we found that the processing of queries for wordnet relations in the OWL metaclass model could be realised using the triple store SWI Prolog Semantic Web library[17] and

[17] We would like to thank Guus Schreiber for pointing out this possibility to us.

the Thea OWL library (Vassiliadis, 2006) for Prolog.

A total conversion of GermaNet into all of the three models examined is under way. We also aim at plugging in more domain ontologies in GermaNet based on their OWL representations; thus far, only connections from TermNet have been tested. The set of Prolog predicates for querying wordnets in OWL according to the metaclass model has to be extended for the remaining wordnet relations. Finally, we plan to develop a metaclass model for TermNet. On this basis, we aim to experiment with alternative plug-in properties that relate the GN metaclass model to the TN metaclass model (and possibly other domain ontologies represented according to this model). We will inform about future work on our project website, cf. `http://www.hytex.info/o4o_werkstatt/o3o_owlmodellierung`.

References

Beißwenger, M., Storrer, A., and Runte, M. (2004). Modellierung eines Terminologienetzes für das automatische Linking auf der Grundlage von WordNet. *LDV-Forum*, 19(1-2):113–125.

Ciorăscu, C., Ciorăscu, I., and Stoffel, K. (2003). knOWLer – Ontological support for information retrieval systems. In *Proceedings of the 26th Annual International ACM-SIGIR Conference, Workshop on Semantic Web*, Toronto, Canada.

Erdmann, M. (2001). *Ontologien zur konzeptuellen Modellierung der Semantik von XML*. Books on Demand, Karlsruhe.

Farrar, S. and Langendoen, D. T. (2003). A linguistic ontology for the semantic web. *GLOT International*, 7(3):97–100.

Fellbaum, C., editor (1998). *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.

Haarslev, V., Möller, R., and Wessel, M. (2004). Querying the Semantic Web with Racer + nRQL. In *Proceedings of the KI-2004 International Workshop on Applications of Description Logics (ADL'04), Ulm, Germany, September 24*, Ulm.

Hirst, G. (2004). Ontology and the lexicon. In Staab, S. and Studer, R., editors, *Handbook on Ontologies*, International Handbooks on Information Systems, pages 209–229. Springer, Heidelberg.

Holler, A., Maas, J.-F., and Storrer, A. (2004). Exploiting coreference annotations for text--to-hypertext conversion. In *Proceedings of LREC 2004, Volume II*, pages 651–654, Lisboa.

Kunze, C. and Lemnitzer, L. (2002). GermaNet - representation, visualization, application. In *Proceedings of LREC*, volume V, pages 1485–1491, Las Palmas.

Kunze, C., Lemnitzer, L., Lüngen, H., and Storrer, A. (2007). Repräsentation und Verknüpfung allgemeinsprachlicher und terminologischer Wortnetze in OWL. *Zeitschrift für Sprachwissenschaft*, 26(2).

Kunze, C., Lemnitzer, L., and Wagner, A. (2003). *Anwendungen des deutschen Wortnetzes in Theorie und Praxis. Sonderheft der Zeitschrift für Computerlinguistik und Sprachtechnologie*. Bonn.

Magnini, B. and Speranza, M. (2002). Merging Global and Specialized Linguistic Ontologies. In *Proceedings of Ontolex 2002*, pages 43–48, Las Palmas de Gran Canaria, Spain.

Miller, G. A. and Hristea, F. (2006). Word net nouns: Classes and instances. *Computational Linguistics*, 32(1):1–3.

Noy (ed.), N. (2005). Representing classes as property values on the semantic web. W3C Working Group Note. http://www.w3.org/TR/swbp-classes-as-values/, visited 23 May 2007.

Pan, J. Z., Horrocks, I., and Schreiber, G. (2005). OWL FA: A metamodeling extension of OWL DL. In *Proceedings of the Workshop OWL: Experiences and directions*, Galway, Ireland.

Schreiber, G. (2002). The web is not well-formed. *IEEE Intelligent Systems*, 17(2):79–80.

Smith, M. K., Welty, C., and Deborah L. McGuinness, e. (2004). OWL Web Ontology Language – Guide. Technical report, W3C (World Wide Web) Consortium. http://www.w3.org/TR/2004/REC-owl-guide-20040210/, visited 24 May 2007.

Sowa, J. F. (2000). *Knowledge Representation. Logical, Philosophical, and Computational Foundations.* Brooks/Cole, Pacific Grove.

Storrer, A. (2008). Mark-up driven strategies for text-to-hypertext conversion. In Metzing, D. and Witt, A., editors, *Linguistic Modelling of Information and Markup Languages. Contributions to Language Technology*, Text, Speech and Language Technology. Kluwer, Dordrecht. To appear.

van Assem, M., Gangemi, A., and Schreiber, G. (2006). Conversion of WordNet to a standard RDF/OWL representation. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC'06)*, Genoa, Italy.

van Assem, M., Menken, M. R., Schreiber, G., Wielemaker, J., and Wielinga, B. (2004). A method for converting thesauri to RDF/OWL. In *Proceedings of the 3rd International Semantic Web Conference (ISWC 2004)*, number 3298 in Lecture Notes in Computer Science, Hiroshima, Japan.

Vassiliadis, V. (2006). Thea. A web ontology language - OWL library for [SWI] Prolog. Web-published manual, http://www.semanticweb.gr/TheaOWLLib/index.htm, visited 15 July 2006.