Modelling and Processing Wordnets in OWL

Harald Lüngen
Michael Beißwenger
Bianca Selzam
Angelika Storrer

Preprint.

To appear in: Alexander Mehler, Kai-Uwe Kühnberger, Hennig Lobin, Harald Lüngen, Angelika Storrer & Andreas Witt (Eds., in press): **Modeling, Learning and Processing of Text Technological Data Structures**. Dordrecht: Springer.

Modelling and Processing Wordnets in OWL

Harald Lüngen, Michael Beißwenger, Bianca Selzam and Angelika Storrer

Abstract In this contribution, we discuss and compare alternative options of modelling the entities and relations of wordnet-like resources in the Web Ontology Language OWL. Based on different modelling options, we developed three models of representing wordnets in OWL, i.e. the instance model, the class model, and the metaclass model. These OWL models mainly differ with respect to the ontological status of lexical units (word senses) and the synsets. While in the instance model lexical units and synsets are represented as individuals, in the class model they are represented as classes; both model types can be encoded in the dialect OWL DL. As a third alternative, we developed a metaclass model in OWL FULL, in which LexicalUnit and Synset are defined as metaclasses, the individuals of which are classes themselves. We apply the three OWL models to each of three wordnet-style resources: (1) a subset of the German wordnet GermaNet, (2) the wordnet-style domain ontology TermNet, and (3) GermaTermNet, in which plugin relations between TermNet technical terms and GermaNet synsets are defined. We report on the results of several experiments in which the performance of querying and processing the different models in two application contexts was evaluated: (1) A comparison of all three OWL models (class, instance, and metaclass model) of TermNet in the context of automatic hyperlinking, (2) an investigation of the potential of the GermaTermNet resource by the example of a wordnet-based semantic relatedness calculation.

Harald Lüngen
Justus-Liebig-Universität Gießen
Zentrum für Medien und Interaktivität
Ludwigstraße 34
D-35390 Gießen
e-mail: luengen@uni-giessen.de

Michael Beißwenger, Bianca Selzam, Angelika Storrer
Technische Universität Dortmund
e-mail: michael.beisswenger|bianca.stockrahm|angelika.storrer@uni-dortmund.de

1 Research Context and Motivation

Wordnets are lexical resources that follow the design principles of the English Princeton WordNet (Fellbaum, 1998). Many applications of natural language processing and information retrieval use wordnets not only as a lexical database but also as an ontological resource representing conceptual knowledge. In order to make use of this conceptual knowledge in the context of semantic-web-related research, several approaches have been put forward that aim at representing the English Princeton WordNet[1] in the Web Ontology Language OWL or RDFS (cf. Section 3.2).

The goal of this paper is to discuss and compare different options to represent the basic entities and relations of wordnets using OWL. For this purpose, we developed alternative OWL models on the same wordnet-style resources: (1) a subset of the German wordnet GermaNet[2], (2) the wordnet-style thesaurus TermNet[3] representing technical terms from the domains of text-technology and hypertext research, and (3) the GermaTermNet representing relations between TermNet technical terms and GermaNet synsets. In Section 2, we briefly describe these resources; in Section 3 we present and compare the alternative OWL models that we created for these resources. The main difference between these models lies in the ontological status of the two main entity types of wordnets: the lexical units (LU) (words) and the synsets (collections of synonymous or near-synonymous lexical units). While the instance models of our resources represent LUs and synsets as individuals, the class models represent these entities as classes. Class and instance models are in the scope of OWL DL and may thus be processed by description logic-based reasoners. As a third alternative, we developed metaclass models which define the two entity types LexicalUnit and Synset as metaclasses. The individuals of these metaclasses – the particular lexical units and synsets – are classes themselves, i.e. these models combine both the instance and the class perspective. However, the metaclass models are outside the scope of OWL DL and can thus not be processed by DL-based reasoners.

Clearly, the comparison and evaluation of the models has to refer to their deployment in concrete applications. In Section 4, we report the results of some experiments in the application context of text-technological information processing: in 4.1 we compare the performance of the three OWL models in the context of automatic hyperlinking using the DL reasoner RACER Pro and the Thea OWL library for SWI Prolog. In 4.2 we used the latter to calculate semantic relatedness measures on the GermaTermNet metaclass model. In the same section, we test the performance and feasability of our plug-in-approach that connects the general language resource GermaNet with the domain-specific TermNet. On the basis of the results reported in Section 4 we discuss the advantages and drawbacks of the three model types in the application contexts[4].

[1] http://wordnet.princeton.edu

[2] http://www.sfs.uni-tuebingen.de/GermaNet/

[3] http://www.hytex.tu-dortmund.de/ressourcen.html#wortnetze

[4] The OWL resources are available for download on the website http://www.wordnets-in-owl.de.

2 Resources

2.1 GermaNet

GermaNet is a lexical-semantic wordnet for German which was developed and is maintained at the University of Tübingen (cf. Kunze and Lemnitzer, 2002). It was designed largely according to the principles of the Princeton WordNet (PWN, cf. Fellbaum, 1998) and covers the most important and frequent general language concepts and relations between the concepts and lexical units like hyponymy, meronymy and antonymy.

The central unit of representation in a wordnet is the *synonym set* (synset), in which those synonymous *lexical units* that are interchangeable in a given context are combined. The synset {*öffnen, aufmachen*}, by example, is represented as a concept node with a number of relational links: to its hyperonym synset {*wandeln, verändern*}, and to its several hyponym synsets e.g. {*aufstoßen*}. Moreover, a causation relation with the synset {*(sich) öffnen, aufgehen*} holds. Finally, the lexical unit *öffnen* is related to its antonym *schließen*.

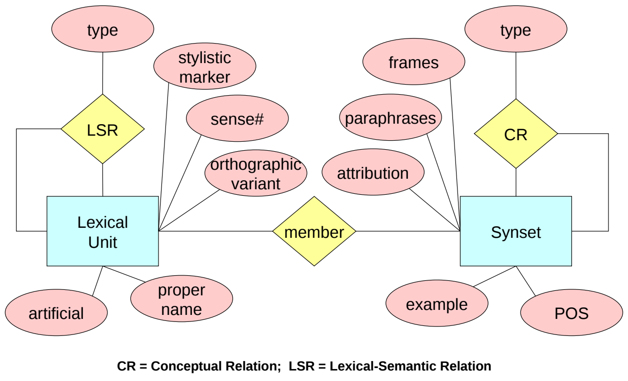

Fig. 1 E-R graph for GermaNet

The data model for GermaNet is shown in the entity-relationship graph in Figure 1[5]. The lexical objects are shown in rectangles, and the attributes that characterise them are shown in ellipses. Relations between the objects are marked as diamonds: conceptual relations (CR) like hyponymy hold between synsets, and lexical-semantic relations (LSR) like antonymy hold between lexical units, and the lexicalisation relation ("member") holds between lexical units and synsets.

GermaNet 5.0 contains about 53000 synsets and 76000 lexical units. To develop the different OWL models for wordnets and the query implementations described in this chapter, we employed a subset of GermaNet containing 54 synsets and 104 lexical units.

[5] from Kunze et al. (2007)

2.2 TermNet

TermNet is a lexical resource that was developed in the HyTex project on automated text-to-hypertext conversion[6]. TermNet represents technical terms occurring in a German corpus on the domains "text-technology" and "hypertext research"[7]. The two basic entities of the TermNet model are *terms* and *termsets*. The entity type *term* corresponds to the entity type *lexical unit* in GermaNet: *terms* denote well-defined concepts in the above-mentioned domains – in many cases they are part of a taxonomy in which more specific terms are defined as subclasses of broader terms. *Termsets* collect terms that denote similar concepts in different taxonomies, e.g. in German hypertext research, the terms "bidirektionaler Verweis" and "bidirektionale Verknüpfung" denote similar concepts in different taxonomies on hyperlinks. Termsets correspond to the entity type *synset* in GermaNet with one significant difference: lexical units in the same synsets are regarded as near-synonyms – they are exchangeable in at least one context. Such an exchange is not possible for different terms in a termset because the semantics of terms is well-defined by their position in the taxonomy. Different authors or scientific schools may develop different and non-isomorphic taxonomies for the same domain. Therefore, it is not possible to exchange a technical term in a document for a technical term defined in a different taxonomy, even if the two terms denote quite similar concepts and therefore belong to the same termset. Termsets represent categorical correspondences between terms of competing taxonomies.[8] This representation of categorical correspondence opens up two modes for searching for term occurrences in a corpus: (1) the *narrow search mode* (based on the entity type *term*) for occurrences of a term in a specific taxonomy, and (2) the *broader search mode* (based on the entity type *termset*) for all occurrences of terms that denote a similar concept in the domain.

Apart from these specific characteristics of termsets, the TermNet data model builds on the entity and relation types of the Princeton WordNet model and the GermaNet model. Whereas lexical semantic relations (LSR) are defined between terms, termsets are related by conceptual relations (CR). The following LSR and CR are used in the TermNet data model:

- LSR: *isAbbreviationOf* (inverse: *isExpansionOf*)
- CR: *isHyponymOf* (inverse: *isHypernymOf*), *isMeronymOf*, *isHolonymOf*

In addition, we introduced a symmetrical relation *isDisjointWith* to describe the relation between terms with disjoint extensions (which is the case with terms which denote contrary subconcepts of one and the same superconcept, e.g., *externer Verweis* and *interner Verweis*).

[6] http://www.hytex.info, and Storrer (2010)

[7] The current version of TermNet contains 423 technical terms in 206 termsets.

[8] A concrete example: in German hypertext research the terms *externer Verweis* and *extratextuelle Verknüpfung* both denote hyperlinks leading to other "external" websites. However the definition of *extratextuelle Verknüpfung* is opposed to two disjunctive terms (*intertextuelle Verknüpfung*, *intratextuelle Verknüpfung*) whereas *externer Verweis* is opposed to only one disjunctive term (*interner Verweis*).

In order to represent the relation between terms and termsets, we furthermore introduced a *membership*-relation which relates terms that denote the same category in different taxonomies with the respective termset (*isMemberOf*) and, inversely, the termset with one or several terms (*hasMember*).

As an extension to the standard WordNet model, TermNet represents subclass relations between terms of the same taxonomy. The data model is illustrated by the ER-diagram in Figure 2; further details are described in Beißwenger et al. (2004).

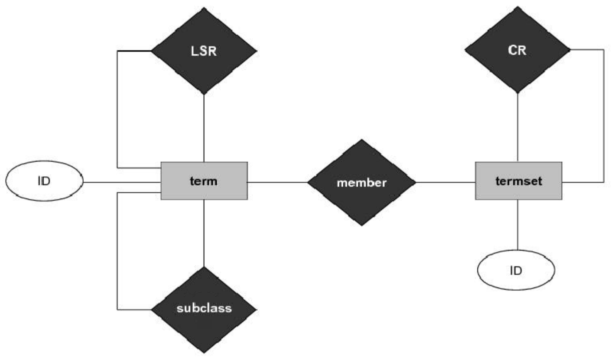

Fig. 2 E-R graph for TermNet

2.3 GermaTermNet

GermaTermNet connects the two above-described resources – the GermaNet subset and TermNet – following an approach inspired by Magnini and Speranza (2002). The basic idea of this approach is to supplement GermaNet and TermNet by so-called plugin-relations[9]. The "plugin" metaphor is motivated by the fact that these relations connect entities from the one with entities from the other resource without the necessity of modifying or merging the two resources. The strength of such a "plugin" approach, thus, lies in the fact that resources which are independent of one another can be linked in order to process them in combination.

The current version of GermaTermNet distinguishes three types of plugin-relations which describe particular correspondences between general language concepts and their terminological counterparts:

1. *attachedToNearSynonym*: This relation describes correspondences between TermNet terms and GermaNet synsets; e.g. between the TermNet term *Link* and the

[9] This approach has been developed in close cooperation with Claudia Kunze and Lothar Lemnitzer of the GermaNet group, see Kunze et al. (2007) and Lüngen et al. (2008) for details.

GermaNet synset *Link*. Since we do not assume pure synonymy for a corresponding term-synset pair, *attachedToNearSynonym* is the closest sense-relation between entities of the two resources.

2. *attachedToGeneralConcept*: This relation is used to relate TermNet terms to GermaNet synsets which stand in an *attachedToNearSynonym* relation with a superordinate term.[10] The *attachedToGeneralConcept* relations serve to reduce the path length between semantically similar concepts for applications in which semantic distance measures are calculated.
3. *attachedToHolonym*: This relation is used to relate a TermNet term t_1 to a synset s_1 in GermaNet when the following conditions are fulfilled: (1) t_1 is a member of a termset A, (2) this termset is a meronym of termset B, and (3) the members of B (i.e., a collection of terms t_2, t_3, ..., t_n) are connected with s_1 through the relation *attachedToNearSynonym*. An example: the term *arc*, a member of the termset *Kante*, is linked to the GermaNet synset *link* by an *attachedToHolonym* relation because *Kante* is a meronym of the termset *Link*, and one of its members, the term *link*, is attached to the GermaNet synset *link* via *attachedToNearSynonym*.

The current version of GermaTermNet represents 150 plugin relation instances: 27 *attachedToNearSynonym*, 103 *attachedToGeneralConcept*, and 20 *attachedToHolonym* instances.

3 Modelling Wordnet-like Resources in OWL

3.1 Basic Options

3.1.1 OWL Dialects

In the line of Gruber (1993), Staab and Studer (2004) characterise an ontology as a "formal explicit specification of a shared conceptualisation for a domain of interest". It is controversial whether wordnets constitute proper ontologies; according to Erdmann (2001), a wordnet may count as a *light-weight ontology*, i.e. an "ontology primarily consisting of a schema (a concept taxononomy with attribute and relation definitions)" (Erdmann, 2001, p. 27, original in German). Sowa (2000) mentions PWN as an example of a "terminological ontology", i.e. an ontology that is not completely formalised.

The Web Ontology Language OWL is an XML application for the representation of ontologies in the semantic web, a standard created by the W3C Web Ontology working group. OWL comes in three sublanguages (dialects), OWL Full, OWL DL, and OWL Lite, which differ in their expressive power. When only constructs from

[10] An example: the term *externer Verweis* is connected to the GermaNet synset *Link* by an *attachedToGeneralConcept* relation because it is a subclass of the term *Verweis* which itself is connected to *Link* by an *attachedToNearSynonym* relation.

the sublanguage OWL DL are used in an ontology, its semantics correspond to description logic (Baader et al., 2004). Most available reasoning software is based on description logic, and reasoners such as Racer (Haarslev and Möller, 2004) support consistency checking and inferring new facts and relations, i.e. automatically extending OWL DL ontologies. As a consequence, most ontologies for the semantic web are encoded in OWL DL.

OWL Full is known to be undecidable. Probably no reasoning services will ever be implemented for ontologies that make full use of the expressive power of OWL Full. One important difference between OWL DL and OWL Full is that in an OWL DL ontology, the three sets of the classes, individuals, and properties must be disjunct, e.g. an entity in the ontology may not be a class and an individual at the same time. This, however, is permitted in an OWL Full ontology.

3.1.2 Synsets and Lexical Units as Classes or Individuals?

A fundamental decision which has to be made when representing Wordnet-like resources in OWL concerns the question whether synsets and lexical units are to be described as classes or as individuals. From a linguistic point of view, synsets are *concepts (classes)* whose instances are discourse entities, and lexical units are types of linguistic expressions whose instances can be interpreted as the token occurrences of these expressions in documents. The decision to treat synsets and lexical units as OWL individuals conceives a wordnet primarily as a lexicon describing properties of individual lexical units while disregarding that nouns, in the traditional view of lexical semantics, denote concept classes, with concept classes being ordered hierarchically, superclasses including subclasses and subclasses in some cases (especially in terminologies) being pairwise disjoint.

Treating synsets and lexical items as *individuals* implies that the subconcept-superconcept relation and disjointness can only be described by non-standard individual-to-individual relations. An individual representing a subconcept does not automatically inherit the properties defined for the individual representing the superconcept.

Treating synsets and lexical items as *classes* allows for describing them as concepts and word senses, with entities of the real world or word occurrences in documents being described as their individuals. The "class model" of a Wordnet-like resource thus describes an ontology, whereas the "instance model" – much like a lexicon – describes synsets and lexical units as instances of linguistic categories such as *NounSynset* or *NounWordSense*.

3.2 Related Work

What OWL models of wordnets have been designed previously, and how are these related to the ones presented in this chapter? In 2006, the *Semantic Web Best Practices and Deployment Working Group* issued a working draft for a standard conversion of the PWN in an RDF/OWL (in fact OWL DL) representation that can be used by semantic web applications (van Assem et al., 2006a,b). Its top-level classes are *Synset*, *WordSense*, and *Word*, where *Synset* represents synsets and has the subclasses *NounSynset*, *VerbSynset*, *AdjectiveSynset*, and *AdverbSynset*. The individual synsets are individuals of these classes. CRs are defined over the synset class, e.g. the OWL object property *hyponymOf* has *Synset* as its domain and its range. Likewise, the class *WordSense* has the subclasses *NounWordSense*, *AdjectiveWordSense*, *VerbWordSense*, and *AdverbWordSense*. Lexical units are modelled as the individuals of these classes. LSRs like *antonymOf* are OWL object properties with *WordSense* as their domain and their range. The lexicalisation relation is an OWL object property called *synsetContainsWordSense* and has the *Synset* as its domain and *WordSense* as its range. The third top-level class, *Word*, represents a purely formal unit, i.e. not associated with a meaning. It is connected with the class *WordSense* via the OWL object property *word*, which has *WordSense* as its domain and *Word* as its range. Properties with an attribute-like character including *lexicalForm* or *synsetId* are rendered as OWL datatype properties with XML Schema datatypes such as xsd:string as their range.

There are two earlier approaches to representing wordnets in OWL, the "Neuchâtel" approach (Ciorăscu et al., 2003b,a) and the "Amsterdam" approach (van Assem et al., 2004). Neuchâtel has been explicitly acknowledged by van Assem et al. (2006b), and Amsterdam and W3C partly have identical authors. The W3C approach is fairly close to Neuchâtel; one difference is that the *StemObject* class in Neuchâtel which corresponds to the *Word* class in the W3C approach is only introduced in a wordnet-external ontology for a document retrieval application. In the Amsterdam model, a different top-level structure of the ontology is specified, mostly in that lexical units are neither classes nor individuals but literal values of the OWL datatype property *wordForm* with the domain *Synset*. As a consequence, the encoding of LSRs is achieved by a different mechanism. We took the W3C approach to convert PWN into OWL as a model for a first GermaNet representation in OWL (Kunze et al., 2007), which is briefly described below in Section 3.3.1 as the "OWL DL Instance Model" for GN. The W3C approach was also adopted by De Luca et al. (2007) for a conversion of EuroWordNet (Vossen, 1999) into OWL.

In (van Assem et al., 2006b), an alternative model is briefly discussed. For some purposes it might be useful to treat the hyponymy relation as a class hierarchy by declaring *Synset* a subclass of rdfs:class (thus make each individual synset a class and an individual at the same time) and *hyponymOf* a subproperty of rdfs:subClassOf as suggested in Wielemaker et al. (2003).

3.3 OWL Models for GermaNet, TermNet, and GermaTermNet

In this section, we describe the three alternative models that we developed for encoding wordnets in OWL, namely the *OWL DL Instance Model*, the *OWL DL class* *model*, and an *OWL Full Model*. They are based on different outcomes for the following modelling decisions:

1. the representation of synsets and lexical units as OWL classes, individuals, or both
2. the encoding of relation instances in OWL as property restrictions or assignments of individuals as property values (as a consequence of 1.)
3. the conformance with the dialect OWL DL or OWL Full (as a consequence of 1. and 2.)
4. the way of linking word occurrences in XML documents to lexical units (strictly speaking not part of the ontology)

We describe the implementation of the three models for GermaNet in Section 3.3.1 and for TermNet in 3.3.2. OWL Models for GermaTermNet are described in Section 3.3.3.

3.3.1 GermaNet

For GermaNet, the top-level hierarchy of classes is defined using the `<owl:class>` and `<rdfs:subClassOf>` statements and is the same in all three models. It corresponds to the indented list representation shown in Figure 3, and is similar to the class hierarchy of the W3C model.

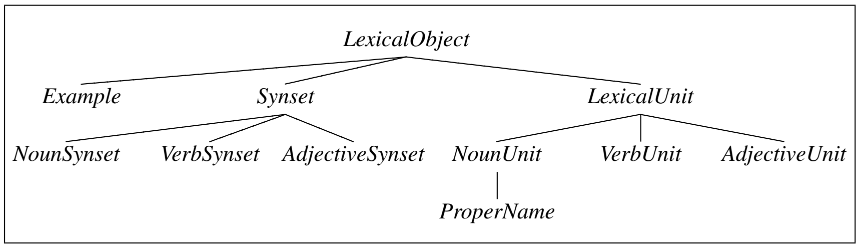

Fig. 3 Class hierarchy for OWL models of GN

The GN relations shown as diamonds in the data model in Figure 1 are modelled as OWL object properties. To capture the commonalities of CRs and LSRs, we introduced the two superproperties *conceptualRelation* (domain and range: *Synset*) and *lexicalSemanticRelation* (domain and range: *LexicalUnit*).

Listing 1 shows the OWL code introducing the lexicalisation relation *hasMember* as an OWL inverse functional property (a special type of object property). Listing 2 shows the code for introducing *isHypernymOf* as an OWL transitive property (also a special type of object property). All other object properties are declared similarly in OWL. Most attributes in the data model of GermaNet (the ellipses in the E-R-graph in Figure 1) are represented as OWL datatype properties (cf. Lüngen et al., 2008).

Property	Domain	Range	Characteristics	Inverse Property	Local Restrictions
Conceptual relations (CR)					
conceptualRelation	Synset	Synset			
isHypernymOf	Synset	Synset	transitive	isHyponymOf	*pos-related*
isHolonymOf	NounSynset	NounSynset			
isMeronymOf	NounSynset	NounSynset			
isAssociatedWith	Synset	Synset			
entails	VerbSynset	VerbSynset			
causes	VerbSynset $\sqcup$ AdjectiveSynset	VerbSynset			
Lexical-semantic relations (LSR)					
lexicalSemanticRelation	LexicalUnit	LexicalUnit			
hasAntonym	LexicalUnit	LexicalUnit	symmetric	hasAntonym	*pos-related*
hasPertonym	LexicalUnit	LexicalUnit			
isParticipleOf	VerbUnit	AdjectiveUnit			
lexicalisation relations					
hasMember	Synset	LexicalUnit	inverse--functional	memberOf	*pos-related*

Table 1 Characteristics of the ObjectProperties for GermaNet

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

Listing 1 OWL code introducing the lexicalisation relation *hasMember* for GermaNet

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

Listing 2 OWL code introducing the hypernymy relation *isHypernymOf* for GermaNet

OWL DL Instance Model Our first modelling variant closely follows the principles of the W3C approach to converting PWN to OWL: the individual synsets and lexical units are modelled as OWL individuals and the encoding of CRs, LSRs and the lexicalisation relation as an assignment of individuals as property values. In the Instance Model for GN, there is an additional class *LUOccurrence*, the individuals of which have (pseudo-) URIs as IDs to simulate the token occurrences of lexemes in documents. *LUOccurrence* is related to its LU individual through the object property *isOccurrenceOf* (domain: *LUOccurrence*, range: *LexicalUnit*).[11]

In Listing 3, OWL code examples of assignments for CRs (*isHypernymOf*) and the lexicalisation relation (*hasMember*) are shown for the synset *vVeraenderung.119*. Examples of assignments for LSRs and datatype properties are shown for the lexical unit *vVeraenderung.199.wandeln*. Finally, there is the occurrence individual *URI_LUnitInstance_vVeraenderung.119.wandeln_1* with an assignment of its LU for the property *isOccurrenceOf*.

```
<VerbSynset rdf:ID="vVeraenderung.119">
   <hasMember rdf:resource="#vVeraenderung.119.wandeln"/>
   <hasMember rdf:resource="#vVeraenderung.119.ändern"/>
   <hasMember rdf:resource="#vVeraenderung.119.mutieren"/>
   <hasMember rdf:resource="#vVeraenderung.119.verändern"/>
   <isHypernymOf rdf:resource="#vVeraenderung.421"/>
   <isHypernymOf rdf:resource="#vVeraenderung.517"/>
</VerbSynset>

<VerbUnit rdf:ID="vVeraenderung.119.wandeln">
   <hasOrthographicForm
      rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
      wandeln</hasOrthographicForm>
   ...
</VerbUnit>

<LUOccurrence rdf:ID="URI_LUnitInstance_vVeraenderung.119.wandeln_1">
    <isOccurrenceOf rdf:resource="#vVeraenderung.119.wandeln"/>
</LUOccurrence>
```

Listing 3 OWL code for a synset, lexical unit, and occurrence individual in the instance model for GN. Note the encoding of the *hasMember* and *isHypernymOf* relation instances.

OWL DL Class Model For the reasons discussed in Section 3.1.2, we alternatively devised an *OWL DL Class Model* for GermaNet. In the class model, each individual synset and lexical unit is a subclass of one of the top-level classes *NounSynset, VerbSynset, NounUnit* etc. Since within OWL DL property value assignments can only be defined between individuals, the GermaNet relation instances are modelled as property restrictions over the single synset or unit classes using *owl:Restriction* in combination with the *owl:allValuesFrom* construct. This holds for CRs, LSRs, as well as the lexicalisation relation, cf. the example of hyponym declarations for the synset *vVeraenderung.119* in Listing 5.

In the class model, the place of individuals of the LU classes can be straighforwardly taken by their textual occurrences. Unlike in the case of the instance model, the GN top-level ontology does not need to be extended to accommodate them.

OWL Full Model In Lüngen and Storrer (2008), we put forward some criticism of the instance model for wordnets in OWL (summarised in Section 3.1.2 above).

[11] Although we think that *LUOccurrence* and *TermOccurrence* (see Section 3.3.2) and their properties are strictly speaking not part of wordnets or the lexicon, we have included them in our ontologies for the purpose of linking our annotated text corpora to them.

```
<owl:Class rdf:ID="vVeraenderung.119">
  <rdfs:subClassOf rdf:resource="#VerbSynset"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:TransitiveProperty rdf:about="#isHypernymOf"/>
      </owl:onProperty>
      <owl:allValuesFrom>
        <owl:Class>
          <owl:unionOf rdf:parseType="Collection">
            <owl:Class rdf:about="#vVeraenderung.421"/>
            <owl:Class rdf:about="#vVeraenderung.517"/>
          </owl:unionOf>
        </owl:Class>
      </owl:allValuesFrom>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

Listing 4 OWL code for the *isHypernymOf* relations in the class model for GN using restrictions

```
<vVeraenderung.119.verändern
      rdf:ID="URI_LUnit_Instance_vVeraenderung.119.verändern_1"/>
```

Listing 5 Occurrence individual in the class model

In the class model, on the other hand, the relation assignments via property restrictions seem counter-intuitive; moreover, although occurrence instances of LUs are conveniently included as individuals of their LU classes, it does not seem adequate that they inherit all property specifications by virtue of their instancehood (CRs and LSRs are *lexical* relations and are not supposed to hold between word occurrences in a document).

Thus, as a third option we converted the OWL instance model of GermaNet into an *OWL Full Metaclass Model* by adding the following line to the definitions of the classes *Synset* and *LexicalUnit*.

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

This makes *Synset* and *LexicalUnit* metaclasses, i.e. their individuals, the single synsets and lexical units, are also classes. Consequently, an LU class can be populated with occurrence individuals that do *not* inherit lexical properties, *and* CRs, LSRs and the lexicalisation relation can be added as simple property value assignments (the code for which looks just like that for the instance model in Listing 3). Unfortunately, the ontology now lies outside the scope of OWL DL, i.e. is in OWL Full (cf. Smith et al., 2004), and DL-based reasoners cannot be applied to it.

A point of criticism one could formulate against the OWL Full model is that when single synsets (and LUs) are classes besides being individuals, then one would also expect these classes to be part of another (second) class hierarchy. In fact, in Noy (2005), Motik (2005), and in the third example in Schreiber (2002), a major motivation for introducing metaclasses is the fact that certain individuals of an ontology should be subclasses in a second class hierarchy, e.g. an *ArchetypalOrangUtan* individual should be an individual of an *Archetype* class and at the same time subclass of an *OrangUtan* class. On the other hand, in the metamodelling examples in Schreiber (2002) (first two examples) and in Pan et al. (2005) (dealing with the PWN), there is no such second hierarchy; the motivation given is simply that some entities are both a class and an individual. In van Assem et al. (2006b), where a metaclass approach to modelling the PWN in OWL is proposed as a variant, the motivation is to interpret the hyponym relation as a (second) class hierarchy.

3.3.2 TermNet

For each one of the three modelling options of TermNet a separate top-level hierarchy of classes is defined using the <*owl:class*> and <*rdfs:subclassOf*> statements. The hierarchy for the OWL DL instance model is shown in Figure 4, the hierarchy for the OWL DL class model and the OWL Full model in Figure 5.

As in our GermaNet models, the CRs and LSRs in TN (diamonds in the data model in Figure 2) are defined as OWL object properties. Listing 6 shows the OWL code defining the *isMemberOf* relation between the term concept *TermConcept_Relation* and the termset *TermSet_Link* in the OWL DL class model. All other object properties in our TN models (as listed in Table 2) are declared similarly.

```
 <owl:Class rdf:ID="TermConcept_Relation">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:ObjectProperty rdf:ID="isMemberOf"/>
      </owl:onProperty>
      <owl:allValuesFrom>
        <owl:Class rdf:ID="TermSet_Link"/>
      </owl:allValuesFrom>
    </owl:Restriction>
  </rdfs:subClassOf>
  ...
</owl:Class>
```

Listing 6 OWL code describing the relation between the term concept *TermConcept_Relation* and the termset *TermSet_Link* in the OWL DL class model of TermNet

OWL DL Instance Model Following the principles of the W3C approach to converting PWN to OWL, the term concepts and termsets of the TN instance model are realised as OWL individuals of the top-level classes *TermConcept* and *TermSet*, and all of the relations mentioned in table 2 are described as object properties between instances. Occurrences of technical terms in documents from our corpus are given as URI references and are described as individuals of the top-level class *TermOccurrence*, and the mutually inverse object properties *occursIn* – *isOccurrenceOf* are used to link individuals of term occurrences with individuals of the *TermConcept* class. In order to describe the lexicalisation relation between term forms and

Property	Domain	Range	Characteristics	Inverse Property	CM	IM	MM
Membership relations							
hasMember	TermSet	TermConcept		isMemberOf	X	X	X
Type/token relations							
isTypeOf	TermForm	TermOccurrence		isTokenOf	X	X	X
Lexicalisation relations							
lexicalizes	TermForm	TermConcept		isLexicalizedAs		X	
Occurrence relations							
occursIn	TermConcept	TermOccurrence		isOccurrenceOf		X	
Lexical-semantic relations (LSR)							
isAbbreviationOf	TermConcept	TermConcept		isExpansionOf	X	X	X
Disjointness							
isDisjointWith	TermConcept	TermConcept	symmetric	isDisjointWith		X	
Conceptual hierarchy between term concepts							
isNarrowerTermOf	TermConcept	TermConcept	transitive	isBroaderTermOf		X	
Conceptual relations (CR) between termsets							
isHyponymOf	TermSet	TermSet	transitive	isHypernymOf	X	X	X
isMeronymOf	TermSet	TermSet			X	X	X
isHolonymOf	TermSet	TermSet			X	X	X

Table 2 Characteristics of the ObjectProperties for TermNet; *CM* = OWL DL class model, *IM* = OWL DL instance model, *MM* = OWL Full metaclass model

term concepts, we additionally introduced the relations *isLexicalizedAs – lexicalizes* which relate instances of *TermConcept* with instances of *TermForm* (cf. Figure 4).

In order to represent the further class hierarchy below the top-level class *Term-Concept*, inclusion between general and more specific terminological concepts is described through the mutually inverse relations *isBroaderTerm – isNarrowerTerm*. Since modelling individual terminological concepts as individuals does not allow for using the OWL construct <*owl:disjointwith*>, disjointness between instances of *TermConcept* was described through the object property *isDisjointWith*.

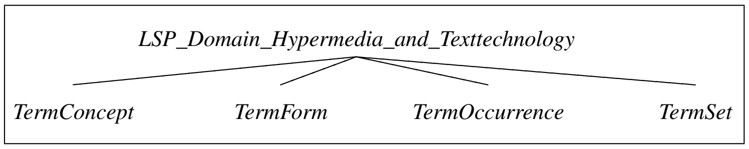

Fig. 4 Class hierarchy for the OWL DL Instance Model of TermNet

OWL DL Class Model The OWL DL Class Model of TermNet was created due to the same reasons as its GermaNet counterpart (see chapter 3.3.1). In the class model, each individual term concept and termset is a subclass of one of the top-level classes *TermConcept* and *TermSet*. Given that two concepts A and B belong to one and the same terminological system with the extension of B representing a subclass of the extension of A, the classes representing A and B are connected through a *superclass-subclass* relation. Given that two subclasses of *TermConcept* can be regarded as being disjoint, we labelled A and B with the OWL construct <*owl:disjointwith*>.

URI references to occurrences of individual technical terms in our corpus are described as OWL individuals of the corresponding specific *TermConcept* classes.

The top-level class *TermForm* bundles the entirety of orthographic forms with which the terminological concepts can be instantiated when used in discourse (in our case in one of our corpus documents). *TermForm* has no subclasses; instead, every single form is described as an individual of this class. Each individual of *TermForm* is linked to a specific URI reference by a *type/token* relation which describes the respective form (e.g. the *TermForm* instance *Link*) as being the orthographic type for the written realisation which can be found under the respective URI in the corpus.

Since in OWL DL classes cannot be assigned as property values for classes, we could establish the CRs, LSRs, and membership relations only indirectly using *owl:Restriction* in combination with the <*owl:allValuesFrom*> construct (cf. Listing 6). Even though restrictions are defined for classes, they do not describe a pair of classes as an instance of a relation. Instead, they determine that the individuals of the respective classes may be connected by OWL object property value assignments. Modelling relation instances between classes would imply a transition to OWL Full. The restrictions in the class model are thus rather workarounds, mainly serving consistency purposes.

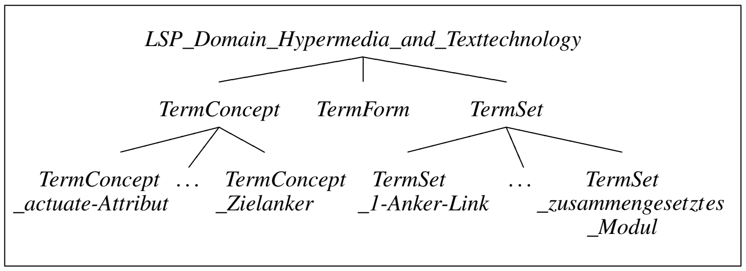
LSP_Domain_Hypermedia_and_Texttechnology
TermConcept
TermForm
TermSet
TermConcept _actuate-Attribut
...
TermConcept _Zielanker
TermSet _1-Anker-Link
...
TermSet _zusammengesetztes _Modul

Fig. 5 Class hierarchy for the OWL DL Class Model and OWL Full Model of TermNet

OWL Full Model By adding the line of code mentioned in section 3.3.1 into the definitions of the classes *TermConcept* and *TermSet*, the OWL DL instance model of TermNet was converted into the OWL Full Metaclass Model. Through this modification, *TermConcept* and *TermSet* become metaclasses; this means that their instances (the individual term concepts and termsets) are both instances and classes at the same time. As a consequence and in contrast to the OWL DL class model, relation instances between subclasses of *TermConcept* and *TermSet* could be explicitly described as property value assignments between classes. Nonetheless and as mentioned in section 3.1.1, OWL Full ontologies cannot be queried and processed by reasoning services which are based on description logics, but e.g. by using a logic programming language such as Prolog.

3.3.3 GermaTermNet

Theoretically, based on the three OWL models of GermaNet and TermNet described above, 3 x 3 = 9 differing OWL models for GermaTermNet may be derived. In the following, we describe the implementation of the plug-in relations of GermaTermNet for:

- the combinations of the respective OWL DL instance models (henceforth referred to as the *GermaTermNet instance-instance model*);
- the combinations of the respective OWL DL class models (henceforth referred to as the *GermaTermNet class-class model*);
- the combination of the OWL DL instance model of GermaNet with the OWL DL class model of TermNet (henceforth referred to as the *GermaTermNet "hybrid model"*);
- the combinations of the respective OWL Full models.

In the instance-instance model, the plugin relations described in Section 2.3 as well as their inverses have been established directly between individuals of TermNet and GermaNet (cf. Listing 7, below). In the class-class model these relations and their inverses have been indirectly realised through establishing *allValuesFrom*-restrictions which apply to the TermNet subclasses of *TermConcept* and the GermaNet subclasses of *Synset*.

```
<rdf:Description
  rdf:about="http://www.owl-ontologies.com/TermNet.owl#TermConcept_Knoten">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:hasValue
       rdf:resource=
       "http://www.owl-ontologies.com/GermaNet-instances.owl#nArtefakt.5816"/>
      <owl:onProperty rdf:resource="#attachedToNearSynonym"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</rdf:Description>
```

```
<rdf:Description
  rdf:about="http://www.owl-ontologies.com/TermNet.owl#TermConcept_Knoten">
  <attachedToNearSynonym
    rdf:resource=
    "http://www.owl-ontologies.com/GermaNet-instances.owl#nArtefakt.5816"/>
</rdf:Description>
```

Listing 7 Connecting resources: Instance of the *attachedToNearSynonym*-relation in the "hybrid model" (above) and in the "instance-instance model" (below); in both examples the TermNet term concept *Knoten* is linked to the GermaNet synset individual *nArtefakt.5816* which comprises the lexical unit *Knoten*

The plugin relations of the "hybrid model" (Kunze et al. (2007)) have been realised through establishing *hasValue*-restrictions which link subclasses of *TermConcept* with individuals of *Synset* (cf. Listing 7, above). Since OWL DL does not allow for specifying inverse relations between classes and instances, the restrictions are defined only one-way from classes to instances, and not vice-versa. The plugin relations of the OWL Full model have been modelled in the same way as in the instance-instance model, namely by establishing direct relations between *TermConcept* and *Synset* individuals.

4 Processing WordNet-like Resources in OWL

In this section, we will discuss the advantages and drawbacks of the three OWL models for wordnets described above when it comes to using them as lexical-semantic resources (4.1) in the context of hypermedia applications and (4.2) in the context of semantic relatedness calculations. Section 4.1 describes, with the example of the different models of the TermNet resource, how the decision for a certain OWL sublanguage – OWL DL or OWL Full – as well as for one of the three models – OWL DL instance model, OWL DL class model or OWL Full Metaclass model – leads to different conditions for querying the resource. We discuss which of the three OWL models can be processed more efficiently and compare three sample queries with respect to the complexity of the query terms needed to implement them on the resource and with respect to the elapsed time between sending the query to the server and receiving the query results. In Section 4.2, we firstly investigate ways to process OWL Full models of wordnets and secondly the potential of a connection of a general language with a specialised wordnet within OWL by the example of wordnet-based semantic relatedness calculation.

4.1 Processing the OWL Models of TermNet

Application scenario: For the comparison of the three OWL models of the TermNet resource, we defined three sample queries. The queries have been defined with respect to the application scenario of the HyTex project, which describes a hypermedia application that supports the selective reading of and the intellectual information retrieval from documents with scientific texts from special domains. The application is designed especially with respect to users who are not experts in the scientific domain to which the documents belong – instead, it is meant to support especially those users who just have basic knowledge of the concepts which are relevant to the domain (user groups such as e.g. students, journalists, or scholars from other scientific disciplines). The design of the hypermedia application is meant to provide these users with all information that they individually need for a proper understanding when selectively browsing through the documents.

Since most of the documents were originally written for a linear medium (print) and converted into the hypertext format ex-retrospect, the text given in the individual hypertext nodes may contain references to passages and concepts introduced in other hypertext nodes which had preceded the current one in the author's original paper plan; or they main contain, on the linguistic level, cohesive devices whose antecedents are not given in the same hypertext node so that the reader needs information either about the associated referent or about the hypertext node or URI where the respective antecedent can be found. Since these prerequisites for understanding mainly derive from the process of hypertextualisation of linearly organised documents, non-expert readers of scientific texts also need support when it comes to the terminological concepts which are relevant in the respective domain and which are addressed or referred to in the documents by the use of technical terms.

This support may be given on the one hand by providing the readers with hyperlinks to definitions of the respective technical terms given in other hypertext nodes or in a glossary and which are relevant for the understanding of the occurrence of the respective term in the hypertext nodes currently being read. But a definition may not help the reader in any context; quite often, a definition may contain other technical terms which also have to be known in order to understand the occurrence of the term defined by the definition. In addition, the comprehension of a terminological concept often requires an at least rudimentary knowledge of the lexical-semantic or conceptual neighbourhood of the respective term. Especially readers with a non-expert status often lack this orientation and therefore also need support for the exploration of the conceptual structure of the respective domain and its representation in the terminologies used in the scientific community.

In our application scenario, TermNet serves as a resource to provide users with the possibility to explore the terminological context of a given technical term (e.g. after having found this term in one of the hypertext nodes) as well as to explore the conceptual structure of the domain irrespective of a specific terminological representation of it (e.g. as given in the terminological system of a certain author or scientific paradigm). Our three sample queries represent needs which can be considered typical for non-expert users when selectively browsing through scientific documents:

User scenario 1: Making use of the TermNet component of the application is driven by the motivation to view all text passages in the corpus that have to do with one particular terminological concept.
Sample scenario: A lexicographer who has to compose the entry “Hyperlink” for a specialised dictionary on information technology should not only be provided with all text passages in which the TermForm(s) *hyperlink* and *link* (as lexicalised forms of the TermConcept *Link*) occur, but in addition with all text passages in which other TermForms occur that do not lexicalise the TermConcept Link, but one of its subconcepts (e.g., *bidirektionaler Link*, *1:1-Link*, *Inhalts-Link*).
→ **Query 1:** “Search for all term occurrences in the corpus which denote a given term concept or a more specific one.” (e.g., “Search for all URIs that have to do with links.”).

User scenario 2: Making use of the TermNet component of the application is driven by the motivation to view all text passages in the corpus that have to to with a particular scientific category.
Sample scenario: A student who has to write a term paper on the hyperlink concept in hypermedia research should not only be provided with all text passages in which

the TermForm(s) *hyperlink* and *link* occur, but in addition with all text passages in which different TermForms occur that relate to the same category, but do not lexicalise the same TermConcept (e.g., also text passages which contain occurrences of the TermForms *Verknüpfung* and *Kante* as well as of TermForms which lexicalise subconcepts of the TermConcepts *Verknüpfung* and *Kante*).
→ **Query 2:** "Search for all term occurrences in the corpus that have to do with links *or similar concepts*." (e.g., "Search for all URIs that relate to TermConcepts which belong to a given termset.").

User scenario 3: While browsing the hypertext nodes of the application, a non-expert user is suddenly faced with the occurrence of a term which seems to be fundamental for the comprehension of the text; s/he decides that s/he should learn more about the category to which this term refers, in order to improve his/her knowledge prerequisites for a proper understanding.
Sample scenario: A student who is planning to attend a course "Introduction to Hypermedia Research" in the upcoming semester wants to get a first overview of the field of hypermedia research. He browses some hypertext nodes of the application and is faced with an occurrence of the TermForm *link*. He already has an intuitive concept of *link* from his everyday internet use but has the intuition that in the scientific literature, the concept area associated with the form *link* might be more differentiated. He thus wants to view other hypertext nodes that have to do with *links*. Since he is motivated to gather information about a certain conceptual area, or category, he should not only be led to hypertext nodes that contain further occurrences of the TermForm link, but also nodes that contain occurrences of different TermForms which are not associated with the same TermConcept Link but with TermConcepts which are members of the same termset to which the TermConcept Link belongs.
→ **Query 3:** "Start from a given term occurrence in the corpus and search for occurrences of any technical terms which denote the same or a similar concept." (e.g., "Search for all URIs that have to do with the same or a similar concept as the term occurrence link in node #127.").

Implementation: For the OWL DL models (TN-IM, TN-CM), we implemented the queries using the query language *nRQL* in combination with RacerPro[12] for processing. Due to the different models, the complexity of the queries vary between TN-CM and TN-IM (as can be seen from the examples in Listing 8 which give the nRQL expressions for query 1). This is, on the one hand, due to the lack of class hierarchies between term concepts in TN-IM: since in TN-IM no superclass-subclass relations with implicit class inclusion are available, the inclusion of specific concepts under general concepts can only be considered by explicitly querying the object property *isNarrowerTermOf*. On the other hand, URIs in TN-IM are modelled as individuals

[12] The acronym *RacerPro* stands for *Renamed ABox and Concept Expression Reasoner Professional* and has been developed by Racer Systems GmbH & Co. KG (http://www.racer-systems.com). In Racer, knowledge bases can be queried using the *new Racer Query Language* (nRQL) which is described in Haarslev et al. (2004).

of a top-level class *TermOccurrence* and not as individuals of term concepts. This is due to the fact that term concepts are already individuals themselves and, thus, in OWL-DL can not have further individuals.

Reasoners such as *RacerPro* are based on description logics and cannot be applied to OWL Full ontologies. However, OWL Full ontologies can be parsed and queried using the Thea OWL Library for *SWI Prolog* (Vassiliadis (2006)) which in turn utilises the SWI-Prolog Semantic Web Library (Wielemaker (2005)). We thus used Prolog for querying the OWL Full Metaclass model (TN-MM). The Prolog predicates for query 1 are given in Listing 8. It searches for all term occurrences of a certain concept or one of its subconcepts.

```
(retrieve
  (?x)
  (?x |TermConcept_Link|)
)
```

```
(retrieve
  (?x)
  (or
    (and
      (?x |TermOccurrence|)
      (|TermConcept_Link| ?x |occursIn|)
    )
    (and
      (?y |TermConcept|)
      (?y |TermConcept_Link| |isNarrowerTermOf|)
      (?x |TermOccurrence|)
      (?y ?x |occursIn|)
    )
  )
)
```

```
termOccurrenceForConceptOrSubConcept(Concept, URI):-
  termOccurrenceForConcept(Concept, URI).

termOccurrenceForConceptOrSubConcept(Concept, URI):-
  transitive_subclassOf(SubConcept, Concept),
  termOccurrenceForConcept(SubConcept, URI).

findOccurrencesForConceptOrSubconcept(Concept, L):-
  findall(URI, termOccurrenceForConceptOrSubConcept(Concept, URI), L).
```

Listing 8 Query 1 as expressed for the different TermNet models: in nRQL for TN-CM (above), in nRQL for TN-IM (middle), and the Prolog predicates for TN-MM (below)

```
(retrieve
  (?x)
  (and
    (?x |http://www.owl-ontologies.com/TermNet.owl#TermConcept|)
    (?x |http://www.owl-ontologies.com/TermNet.owl#TS_Link|
      |http://www.owl-ontologies.com/TermNet.owl#isMemberOf|)
  )
)
```

```
(retrieve
  (?x)
  (and
    (?x |http://www.owl-ontologies.com/TermNet.owl#TermOccurrence|)
    (?y |http://www.owl-ontologies.com/TermNet.owl#TermConcept|)
    (?y |http://www.owl-ontologies.com/TermNet.owl#TermSet_Link|
      |http://www.owl-ontologies.com/TermNet.owl#isMemberOf|)
    (?y ?x |http://www.owl-ontologies.com/TermNet.owl#occursIn|)
  )
)
```

```
termOccurrenceForSet(Set, URI):-
  termConceptForSet(Set, Concept),
  termOccurrenceForConcept(Concept, URI).

findOccurrencesForSet(Set, L):-
  findall(URI, termOccurrenceForSet(Set, URI), L).
```

Listing 9 Query 2 as expressed for the different TermNet models: in nRQL syntax for TN-CM (above), in nRQL for TN-IM (middle), and the Prolog predicates for TN-MM (below)

In order to evaluate the efficiency with which the three models can be processed, we compared the queries 1 and 2 according to (a) the complexity of the query expression and (b) the average time elapsed between sending the query and receiving its answer. We defined "query complexity" as the number of conjuncts, i.e. the number of atoms which are connected through AND-relations, in the query body. We measured the average elapsed time by averaging over the results from executing each query one hundred times per model on one and the same machine. This experiment was conducted on a 32-bit Windows XP machine (AMD Turion 64 X2 2.2 GHz with 2.4 GB RAM). As a reasoner, we used *RacerPro 1.9.0* running at *localhost*. The prolog queries had been processed with *SWI Prolog*[13] in combination with the semweb.pl and Thea libraries for parsing and querying OWL documents.

Table 3 shows the results of the evaluation of TN-CM, TN-IM, and TN-MM with the two queries. The results show that the nRQL expressions needed to query TN-IM are significantly more complex than the expressions needed for TN-CM. Concerning the time needed to process the queries, the findings are converse: although the query expressions for TN-CM are less complex than the expressions for TN-IM, the elapsed time between sending the query and receiving the answer is 46.7% higher at average for TN-CM than for TN-IM. Since the TN-CM resource consists of 72% axioms and 28% facts whereas the TN-IM resource consists of 1% axioms and 99% facts (cf. Table 4), this may be due to the fact that axioms have to go through several preprocessing steps before being able to be handled by a reasoner (cf. Sirin

[13] http://www.swi-prolog.org/

et al. (2007)) so that knowledge bases with large numbers of axioms may degrade performance while being queried (Horrocks and Tobies (2000)).

The complexity of the queries to TM-MM in Prolog+Thea measured by the number of conjunct predicates called is higher because Thea does not store the OWL facts as simple triples of atoms but uses a list representation for the types of an individual. Besides, a predicate for namespace expansion needs to be called several times, and the transitivity of the subclass needed to be implemented and called specifically. The number in brackets in Table 3 shows the complexity when the member, namespace expansion, and transitivity calls are subtracted and is supposedly better comparable with the nRQL/SWRL complexities given for the TM-IM and TM-CM.

Query	**Model**	**Complexity**	**Elapsed time**		
			average (seconds)	*variance*	*standard deviation*
1	CM	1	3.19	0.0428	0.20688
	IM	6	2.10	0.0027	0.05196
	MM	7 (3)	2.85	0.0016	0.04000
2	CM	2	3.04	0.0031	0.05568
	IM	4	2.15	0.0014	0.03742
	MM	11 (3)	2.85	0.0014	0.03742

Table 3 Differences in processing the three TN models with queries 1 and 2 according to query complexity and elapsed time

Even though they are more complex than the nRQL queries for TN-CM, the Prolog queries for TN-MM are processed faster than the queries for TN-CM. While *SWI Prolog* needed 0.95 s per conjunction for processing the TN-MM queries, Racer Pro needed 3.19 s (query 1) and 1.52 s (query 2) per conjunction for processing the TN-CM queries. Nevertheless, the best results were obtained for the TN-IM queries (0.35 s and 0.5375 s per conjunction) which might be due to the fact that when generating TN-MM from TN-IM, class inclusion between term concept classes had been re-introduced so that TN-MM contains 56 times more axioms than TN-IM (cf. Table 4).

	TN-CM	**TN-CM (expanded)**	**TN-IM**	**TN-IM (expanded)**	**TN-MM**
Axioms	5,511	5,516	82	87	4,595
Facts	2,145	2,754	6,153	7,814	5,459

Table 4 Number of axioms and facts in the different TN models (for TN-CM and TN-IM before and after applying the SWRL rules given in Listing 10)

Besides the processing speed, TN-MM has the advantage that Prolog is a much more powerful querying device than nRQL. This can be illustrated by means of query 3: While it is no problem to express this query in Prolog, the nRQL syntax is restricted to the definition of simple inference patterns on the basis of class inclusion, superclass-subclass relations, or object properties. In order to define an nRQL expression to process this query for the two OWL DL models with *RacerPro*, we therefore first had to expand our resources by a new symmetrical object property which connects URI individuals which are occurrences of term concepts that are members of one and the same termset. For this purpose, we used the *Semantic Web Rule Language*[14] (*SWRL*) in order to automatically infer instances of a new object property named *instantiatesSimilarConceptAs* into TN-CM and TN-IM. *SWRL* allows for the augmentation of ontologies by inferring new facts (information about individuals) through the definition of implications between antecedents and consequents.

For the definition of our SWRL rule, we used the *Rules Tab* module[15] of the Protégé editor. For the application of SWRL rules to OWL resources, *Rules Tab* uses the *Jess* reasoner[16]. The application of the SWRL rule then automatically added new facts (information about individuals) to the resources, namely 609 instances of the property to TN-CM and 1,661 instances to TN-IM (cf. Table 4). The difference in the number of properties added to TN-CM and to TN-IM is due to the different modelling of term concepts and termsets as described in chapter 3.3. In order to test querying relatedness between classes in TN-CM anyway, we added dummy instances to some (but not all) of our TermSet classes. Thus, the number of object properties added by applying the SWRL rule is less for TN-CM than for TN-IM. The SWRL rule for TN-CM and TN-IM is given in Listing 10.

```
TermConcept(?y) ∧ TermSet(?z) ∧ isMemberOf(?y, ?z) ∧ TermConcept(?a) ∧
  hasMember(?z, ?a) → instantiatesSimilarConceptAs(?y, ?a)
```

```
TermOccurrence(?x) ∧ TermConcept(?y) ∧ occursIn(?y, ?x) ∧ TermSet(?z) ∧
  isMemberOf(?y, ?z) ∧ TermConcept(?a) ∧ isMemberOf(?a, ?z) ∧
  TermOccurrence(?b) ∧ occursIn(?a, ?b) → instantiatesSimilarConceptAs(?x, ?b)
```

Listing 10 SWRL rules used for query 3 as specified for TN-CM (above) and TN-IM (below)

4.2 Processing the OWL Full version of GermaTermNet

Previously, we implemented a set of basic queries to the GermaNet OWL Full model in Prolog on top of Thea, reported in Lüngen and Storrer (2008). That implementation mostly allowed for querying hyponym sets and could be applied as well to the OWL DL instance model of GN. For the present study we aimed at a more complex querying scenario for wordnets. It should

[14] http://www.w3.org/Submission/SWRL/

[15] http://protege.cim3.net/cgi-bin/wiki.pl?SWRLTab

[16] http://herzberg.ca.sandia.gov/jess/

- include different kinds of elementary queries and ways of combining them
- utilise genuine OWL Full features, i.e. the class+individual status of lexical units in GN and of term concepts in TN.
- utilise GermaTermNet, i.e. analyse the plug-in relations connecting the general language GN with the domain-specific TN
- start from or involve the occurrence individuals which emulate word token occurrences in documents
- constitute a component in a typical text-technological application

We chose the area of wordnet-based calculation of semantic distance/relatedness for querying the GermaTermNet OWL Full model. Determining the semantic relatedness of words/concepts is a task in many computational linguistic and text-technological applications such as word sense disambiguation, lexical chaining, automatic hypertextualisation, discourse parsing, or information retrieval.

According to Budanitsky and Hirst (2001) and also cited in Cramer and Finthammer (2008), *semantic relatedness* should be distinguished from *semantic similarity*. Two concepts (or via lexicalisation, lexemes) are semantically *similar* only if a synonymy or hypernymy relation holds between them. W.r.t the concepts and relations defined for TermNet and the plug-in relations defined for GermaTermNet in this chapter, we suggest to also count membership in one TermSet and an *attachedToNearSynonym* link as a sufficient criterion for semantic similarity. Examples of this are the pairs *aufgehen-sich öffnen*, *Gewässer-Meer*, *Link-Relation*, and *Dokument (GN)-Dokument (TN)*.

Two concepts are semantically *related* if any systematic relation such as synonymy, antonymy, hypernymy, holonymy, or any unsystematic, functional relation or frequent association holds, as in *day-night*, *tree-branch*, or *flower-gardener* (cf. Budanitsky and Hirst, 2001; Cramer and Finthammer, 2008). Semantic similarity is thus a subconcept of semantic relatedness.

How best to approximate semantic relatedness computationally is an area of current research. Several semantic relatedness measures have been defined, partly using WordNet (or more general, a wordnet) as a lexical resource. Overviews and evaluations of semantic relatedness measures are included in the recent publications by Budanitsky and Hirst (2001, 2006), and Cramer and Finthammer (2008). To test the queriability of our OWL Full model of (merged) wordnets, we implemented Wu and Palmer's *conceptual similarity* (Wu and Palmer, 1994). It is defined as in the following notation found in Budanitsky and Hirst (2006).

$$wp(c_1,c_2) = \frac{2*depth(lcs(c_1,c_2))}{len(c_1,lcs(c_1,c_2))+len(c_2,lcs(c_1,c_2))+2depth(lcs(c_1,c_2))}$$

where $c_{1,2}$ are concepts in a wordnet, *lcs* is their lowest common superconcept, *len* is the length of a path along the hypernymy hierarchy, and *depth* is the length of the path to a given concept from the root node of the hypernymy tree.

Consequently, the implementation of *wp* in Prolog using the Thea library consists of three main functions: first, the construction of the hypernymy (or more general, superconcept) tree; second, the calculation of the depth of one concept node; and third, the calculation of the *lcs* of two concept nodes.

Generating the hypernymy (or rather, superconcept) tree consists of inserting a root node and connecting it to those concepts that had no superconcepts in the first place. In GermaTermNet, the superconcept hierarchy is formed by four different path types:

1. the usual *isHyponymOf* relation between synsets in GermaNet
2. the *subclassOf* relation between term concepts in TermNet
3. the plug-in relation *attachedToNearSynonym* between term concepts in TN and synsets in GN
4. the plug-in relation *attachedToGeneralConcept* between term concepts in TN and synsets in GN

We implemented the calculation of depth and path lengths (*len*) in the *wp* formula such that an *attachedToNearSynonym* link counts 0, and the other three types of link count 1 (cf. Listing 11).

Concerning the *subclassOf* links between term concepts in TN, only those below the attachment points to GN are taken into account, i.e. the upper part of the term concept hierarchy is "eclipsed" in this application (cf. Magnini and Speranza, 2002; Lüngen and Storrer, 2008); otherwise, certain specialised terms would become situated too close to the root node and distort *wp* calculation.

Because the superconcept hierarchy includes multiple inheritance, there may be multiple depth values for one concept. We disambiguate this naturally by considering only the minimum depth in the calculation of *wp*. For the same reason, two concepts may have more than one *lcs*; in the implementation, we choose the one that is found first. Furthermore, the Thea OWL parser does not seem to be able to deal with *owl:import* specifications, so we worked with a version where the components of GermaTermNet were merged in one file.

The present implementation has some limitations: the GermaTermNet OWL Full ontology that we tested does not contain the complete GermaNet but only the representative subset mentioned in Section 2.1. The complete GermaNet can probably not be parsed and loaded into Prolog in reasonable time on the type of PCs that we used. We also noticed that the multiple *owl:disjointWith* specifications of TermNet delayed the parsing of GermaTermNet considerably. We thus removed them for *wp* calculation.

The Prolog coding of *wp* was straightforward using the OWL predicates that are made available by Thea. Examples of queries for a Wu-Palmer calculation of a concept pair and an occurrence pair are given in Listing 12.

Thus, the OWL Full model of a wordnet merged in OWL according to the plug-in approach can be processed in the Prolog-plus-semantic-web-libraries framework. A number of further interesting observations was made as well:

In the implementation, it is fairly easy to extend or reduce the set of link types used for the calculation of the superconcept hierarchy and to specify their respective distance counts. This allows for e.g. including other relations like the holonymy relation in the calculation of the superconcept hierarchy or for a different weighting of plug-in links as opposed to GermaNet links as opposed to TermNet links.

```
% direct_isSubConceptOf_GTN/3

direct_isSubConceptOf_GTN(A, B, 1):-
        direct_isHyponymOf_S(A, B).

direct_isSubConceptOf_GTN(A, B, 1):-
        attachedToGeneralConcept(A, B).

direct_isSubConceptOf_GTN(A, B, 0):-
        attachedToNearSynonym(A, B).

direct_isSubConceptOf_GTN(A, B, 1):-
        subclassOf(A, B).
%---
attachedToGeneralConcept(S, T):-
        individual(S, _, _, PrValueList),
        member(value('attachedToGeneralConcept', T), PrValueList).
```

Listing 11 Prolog code for calculating relatedness. *subclassOf/2*, *individual/4*, and *value/2* are predicates from the Thea library

```
?- wp('tn:TermConcept_extensional_definierter_Verweis',
      'tn:TermConcept_Beziehung', W).
   W = 0.571429

?- wpo('tn:URI_TermInstance_extensional_definierter_Verweis_1',
       'gn:URI_LUnit_Instance_aVerhalten.239.arrogant_2', W).
   W = 0.222222.
```

Listing 12 Semantic relatedness queries in Prolog, wp = query with concepts, wpo = query with occurrences

It was also straightforward to “eclipse” a part of the GermaTermNet ontology dynamically as part of the *wp* application (*eclipse* was originally described as a static effect in the merger of a general with a specialised wordnet in Magnini and Speranza (2002)).

Furthermore, *wp* calculation for a pair of concepts could be projected to *wp* calculation for a pair of occurrences easily by making reference to the properties that represented occurrence in Prolog. This is not an advantage of the OWL Full model, though, as the implementation would be similar for the instance model. However, processing property value specifications in the OWL Full model is a true advantage over processing the OWL class restrictions of the class model.

5 Conclusion

In this paper, we discussed modelling and processing issues of wordnets represented in the web ontology language *OWL*. We introduced the features of an Instance Model, a Class Model, and a Metaclass Model for rendering wordnets in OWL by the example of the three resources GermaNet, TermNet, and GermaTermNet.

In the Instance Model, which is favoured for the Princeton WordNet for English by the W3C Semantic Web Best Practices and Deployment Group, synsets and lexical units are rendered as OWL individuals, and the links representing lexical-semantic relations are rendered as simple property assignments. A drawback of this model is that occurrence individuals cannot be modelled as instances of their lexical units. Instead, an additional occurrence class must be introduced, and occurrences are linked in a distinguished *occursAs* relation. Moreover, the class/individual distinction in OWL cannot be used to model wordnet instances as introduced for PWN 2.1 (cf. Miller and Hristea (2006)).

In the Class Model, on the other hand, synsets and lexical units are rendered as OWL classes. Many domain ontologies or terminologies in OWL are represented according to this model (e.g. the GOLD ontology for linguistic description, cf. Farrar and Langendoen (2003), or TermNet, cf. Lüngen et al. (2008)). With regard to lexical-semantic networks, the Class Model is best compatible with the traditional view of lexical semantics according to which a noun semantically denotes a concept class. That means that even wordnet instances as described in Miller and Hristea (2006) could be incorporated naturally as OWL individuals of synset classes. However, lexical-semantic and conceptual relations as well as the lexicalisation relation and the plug-in relations of GermaTermNet must all be encoded using OWL property restrictions on synset and LU classes, which is a less than ideal solution when modelling simple wordnet relation assignments. Moreover, when introducing occurrence individuals of lexical unit classes (Section 3.3.2), it seems inadequate that these also inherit all the lexical properties of the LUs such as style marking or the lexical-semantic relations relation assignments.

Thus, in view of the drawbacks of the Instance and Class Models, we introduced the Metaclass Model for wordnets in OWL. It offers a combination of the advantages of the Class and Instance Model because synsets and lexical units are viewed as classes and individuals at the same time. On the other hand, it introduces a new practical problem as a wordnet encoded according the the Metaclass Model is in the dialect OWL Full and thus not processible by most standard DL-based reasoning software.

In Section 4, we examined ways to process the different models for wordnets in OWL in two scenarios. In the first scenario, we evaluated the performance of queries for related term occurrences in documents such as they are typically queried on TermNet in the hypertextualisation application. For querying the Class Model and the Instance Model, we employed the the query language nRQL with the DL reasoner software RacerPro, and the rule language SWRL with the java-based rule engine Jess. The rules and queries for the Metaclass Model were all formulated in Prolog, using SWI Prolog and the semweb.pl and Thea libraries.

In general, the formulation of queries in nRQL was less complex on the Class Model than on the Instance Model, mainly due to the rendering of occurrence individuals as instances of lexical unit classes in the Class Model. However, it turned out that the Class Model still had to be extended by dummy individuals to get the desired results for our queries. Also the processing of all three queries took about one-third longer than processing the corresponding queries on the Instance Model.

For querying the Metaclass Model, DL reasoners such as RacerPro could not be employed; thus we used Prolog and its Semantic Web Library as a rule and query language. Formulating the queries in the first scenario was straightforward but still somewhat more complex than querying the Class Model in nRQL due to the way property assignments are stored in Prolog facts for individuals by the Thea OWL parser. The time it took to execute the queries lay between the times for querying the Class Model and the Instance Model using nRQL and RacerPro.

In the second scenario, we evaluated the feasability of implementing the more complex text-technological task of calculating semantic similarity on the combined ontology GermaTermNet in the Instance+Instance Model (using SWRL rules) and the Metaclass+Metaclass Model (using SWI Prolog and its semantic web libraries).

Unlike in the Class Model (where OWL property restrictions must be checked, cf. Lüngen and Storrer (2008)), combinations of path types can easily be analysed on the Thea representation of the Metaclass+Metaclass Model of GermaTermNet in Prolog (and the same would hold for the Instance+Instance Model).

Although SWRL is a logical rule language based on Horn logic like Prolog and also includes a built-in function library, it was impossible to approximate semantic similarity calculation on the Instance+Instance Model using only SWRL. The reason is that SWRL lacks all the general programming language features of Prolog, such as binding temporary results to a variable and changing the variable content, or a maximum function, which is needed in the calculation of semantic similarity. To implement this on the basis of SWRL, one would have to embed the OWL ontology+SWRL rules in a programming language environment as well, e.g. a Java application using the SWRL-Java API[17].

Based on the findings of the experiments reported in this chapter, we favour the Metaclass Model for wordnets in OWL and process them using the Prolog semantic web libraries semweb and Thea. However, one has to keep in mind that Thea is not a W3C-endorsed effort and that the semantics of pure Prolog differ from the semantics of OWL in various respects, such as Prolog's closed world assumption vs. an open world assumption for OWL.

Using SWRL, on the other hand, may be more sustainable as it has the status of a W3C member submission and is more easily combinable with OWL. More and more implementations of SWRL features and APIs are likely to emerge in the future so that it might become easier to integrate OWL+SWRL ontologies into procedural programming applications as well.

References

Franz Baader, Ian Horrocks, and Ulrike Sattler. Description logics. In Steffen Staab and Rudi Studer, editors, *Handbook on Ontologies*, International Handbooks on Information Systems, pages 3–28. Springer, Heidelberg, 2004.

[17] http://www.daml.org/rules/proposal/jaxb/

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

Alexander Budanitsky and Graeme Hirst. Semantic distance in WordNet: an experimental, application-oriented evaluation of five measures. In *Proceedings of the Workshop on WordNet and Other Lexical Resources, Second meeting of the North American Chapter of the Association for Computational Linguistics (NAACL)*, Pittsburgh, 2001.

Alexander Budanitsky and Graeme Hirst. Evaluation wordnet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32(1):13–47, 2006.

Claudia Ciorăscu, Iulian Ciorăscu, and Kilian Stoffel. knOWLer – Ontological support for information retrieval systems. In *Proceedings of the 26th Annual International ACM-SIGIR Conference, Workshop on Semantic Web*, Toronto, Canada, 2003a.

Iulian Ciorăscu, Claudia Ciorăscu, and Kilian Stoffel. Scalable ontology implementation based on knOWLer. In *Proceedings of the 2nd International Semantic Web Conference (ISWC2003), Workshop on Practical and Scalable Semantic Systems*, Sanibel Island, Florida, 2003b.

Irene Cramer and Marc Finthammer. An evluation procedure for word net based lexical chaining: Methods and issues. In *Proceedings of the Global Wordnet Conference - GWC 2008*, pages 121–146, Szeged, Hungary, 2008.

Ernesto William De Luca, Martin Eul, and Andreas Nürnberger. Converting EuroWordNet in OWL and Extending it with Domain Ontologies. In Claudia Kunze, Lothar Lemnitzer, and Rainer Osswald, editors, *Proceedings of the GLDV-2007 Workshop on Lexical-Semantic and Ontological Resources*, volume 336 of *Informatik. Berichte*, pages 39–48, Hagen, 2007. Fern-Universität Hagen.

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

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

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

Thomas R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220, 1993.

Volker Haarslev and Ralf Möller. RACER User's Guide and Reference Manual Version 1.7.19. Technical report, Technische Universität Hamburg-Harburg, 2004. With contributions from Michael Wessel. `http://www.sts.tu-harburg.de/ r.f.moeller/racer/`, visited 15 January 2010.

Volker Haarslev, Ralf Möller, and Michael Wessel. 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, 2004.

Ian Horrocks and Stephan Tobies. Reasoning with axioms: Theory and practice. In *Proceedings of the 7th International Conference on Principles of Knowledge Representation and Reasoning (KR 2000*, pages 285–296. Morgan Kaufmann, 2000.

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

Claudia Kunze, Lothar Lemnitzer, Harald Lüngen, and Angelika Storrer. Repräsentation und Verknüpfung allgemeinsprachlicher und terminologischer Wortnetze in OWL. *Zeitschrift für Sprachwissenschaft*, 26(2), 2007.

Harald Lüngen and Angelika Storrer. Domain ontologies and wordnets in OWL: Modelling options. *LDV-Forum. GLDV-Journal for Computational Lingjistics and Language Technologie*, 22(2):1–17, 2008.

Harald Lüngen, Claudia Kunze, Lothar Lemnitzer, and Angelika Storrer. Towards an integrated OWL model for domain-specific and general language wordnets. In *Proceedings of the Fourth Global WordNet Conference (GWC 2008)*, pages 281–296, Szeged, Hungary, 2008.

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

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

Boris Motik. On the properties of metamodeling in OWL. In *Proceedings of the Fourth International Semantic Web Conference (ISWC2005)*, 2005.

Natasha Noy. Representing classes as property values on the semantic web. W3C Working Group Note, 2005. `http://www.w3.org/TR/swbp-classes-as-values/`, visited 15 January 2010.

Jeff Z. Pan, Ian Horrocks, and Guus Schreiber. OWL FA: A metamodeling extension of OWL DL. In *Proceedings of the Workshop OWL: Experiences and directions*, Galway, Ireland, 2005.

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

Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A practical OWL-DL reasoner. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(2):51–53, June 2007.

Michael K. Smith, Chris Welty, and eds. Deborah L. McGuinness. OWL Web Ontology Language – Guide. Technical report, W3C (World Wide Web) Consortium, 2004. `http://www.w3.org/TR/2004/REC-owl-guide-20040210/`, visited 15 January 2010.

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

Steffen Staab and Rudi Studer, editors. *Handbook on Ontologies*. International Handbooks on Information Systems. Springer, Heidelberg, 2004.

Angelika Storrer. Mark-up driven strategies for text-to-hypertext conversion. In Dieter Metzing and Andreas Witt, editors, *Linguistic Modeling of Information and Markup Languages. Contributions to Language Technology*, Text, Speech and Language Technology. Springer, Dordrecht, 2010.

Mark van Assem, Maarten R. Menken, Guus Schreiber, Jan Wielemaker, and Bob Wielinga. 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, 2004.

Mark van Assem, Aldo Gangemi, and Guus Schreiber. 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, 2006a.

Mark van Assem, Aldo Gangemi, and Guus Schreiber. RDF/OWL Representation of WordNet. W3C Public Working Draft of 19 June 2006 of the Semantic Web Best Practices and Deployment Working Group, 2006b. `http://www.w3.org/TR/wordnet-rdf/`, visited 15 January 2010.

Vangelis Vassiliadis. Thea. A web ontology language - OWL library for [SWI] Prolog. Web-published manual, 2006. `http://www.semanticweb.gr/TheaOWLLib/`, visited 15 January 2010.

Piek Vossen. *EuroWordNet: a mutlilingual database with lexical-semantic networks*. Kluwer Academic Publishers, Dordrecht, 1999.

Jan Wielemaker. SWI-Prolog Semantic Web Library, 2005. `http://www.swi-prolog.org/pldoc/package/semweb.html`, visited 15 January 2010.

Jan Wielemaker, Guus Schreiber, and Bob Wielinga. Prolog-based infrastructure for RDF: Scalability and performance. In D. Fensel, K. Sycara, and J. Mylopoulos, editors, *Proceedings of the 2nd International Semantic Web Conference (ISWC2003)*, volume 2870 of *Lecture Notes in Computer Science*, pages 644–658, Sanibel Island, Florida, 2003. Springer Verlag.

Zhibiao Wu and Martha Palmer. Verb semantics and lexical selection. In *Proceedings of the 32nd annual meeting of the Association for Computational Linguistics - ACL 1994*, pages 133–138, New Mexico, 1994.