Declarations of Relations, Differences and Transformations between Theory-specific Treebanks: A New Methodology

Felix Sasaki, Andreas Witt, Dieter Metzing

Bielefeld University
Department of Computational Linguistics and Text Technology
{felix.sasaki,andreas.witt,dieter.metzing}@uni-bielefeld.de

1 Introduction

This paper deals with the problem of how to interrelate theory-specific treebanks and how to transform one treebank format to another. Currently, two approaches to achieve these goals can be differentiated. The first creates a *mapping algorithm* between treebank formats [18]. Categories of a source format are transformed into a target format via a given set of general or language-specific mapping rules. The second relates treebanks via a transformation to a *general model of linguistic categories*, for example based on the EAGLES recommendations for syntactic annotations of corpora [5], or relying on the HPSG framework [12]. For both approaches, the following desiderata are discussed [3]:

- In the transformation of theory- or language-specific treebanks, a direct *mapping algorithm* of categories from the source format to a target format is not sufficient. Categories should be interrelated in several steps, making use of information on multiple levels of linguistic description, e.g. node types, parts of speech etc., which are part of the source treebank format (desideratum A).

- A *general model of linguistic categories*, though valuable for many processing tasks, leads to a loss of theory-specific information. Hence, in addition, a model is necessary that does not generalize over categories, but concentrates on the *explication of differences* between them (desideratum B).

This paper proposes a new methodology as a solution for these desiderata.

conceptual level

HPSG
Model

General
Model

Depend.
Model

identity   identity

Head   Phrase   Bunsetsu

identity

data

Annotations for HPSG

hd

shucchou   no   keN

Annotations for Dependency Model

bunsetsu

shucchou   no   keN

hd annotation   (x) concept   (M) conceptual model

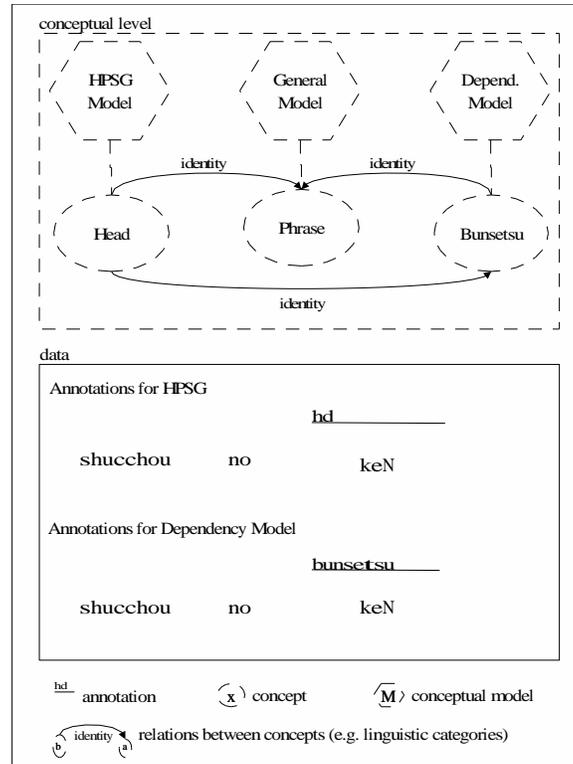(b) identity (a) relations between concepts (e.g. linguistic categories)

Figure 1: Overview of the methodology

2 The methodology

2.1 General outline

The main characteristics of the methodology are *declarative explications of relations between theory-specific categories of treebanks*. As an example, the Japanese phrase "shuchou no keN" 'the business-trip matter'[1] will be used. The methodology, which is visualized in fig. 1, proposes a set of formal properties for treebank annotations, for their conceptual description and for transformation operations:

- For each of the linguistic models, the relevant categories, i.e. concepts of a treebank, are declared at a formalized *conceptual level*. (See upper part of fig. 1). For example **Head** is a relevant concept for an HPSG-model, **Bunsetsu** is part of an Japanese dependency model. (See also section 3). Properties

---

[1]The glossing is like follows: "shucchou" 'business-trip' "no" 'genitive-marker' "keN" 'matter'.

are assigned to categories to express the *relations* between them, for example **Head identity Bunsetsu**, or **Bunsetsu identity Phrase**.

- The relations between the categories at the conceptual level are interpreted as descriptions of *configurations* between annotation units, which are explored within *annotated data*. (See lower part of fig. 1). The configurations can be used to create and validate transformation rules. The rules are created at the conceptual level. For example, the annotations for the concepts **Head** and **Bunsetsu**, namely `hd` and `bunsetsu`, span the same data. This configuration corresponds to the interconceptual relation **Head identity Bunsetsu**. The transformation rule will be straightforward: **Head -> Bunsetsu** or **Bunsetsu -> Head**.

- *Transformation rules* are applied to transform theory-specific annotations into other theory-specific annotations or into annotations which are in accordance with a general, conceptual model. In fig. 1, the concepts **Head** and **Bunsetsu** both can be categorized as a **Phrase**, which is part of a more general model.

There are four advantages of this methodology. First, there is no limitation in extending or recreating a set of categories at the conceptual level; the feasibility of this process can be evaluated in annotated data. This helps to empirically separate notational variants from categories which show substantial differences (desideratum B). Second, several more or less specific or general treebank models can be used to analyze the same annotations (desideratum B). Third, relations between treebank formats are created for each category separately. This allows for the evaluation and reformulation of transformation declarations at the most fine-grained, atomic level (desideratum A). And fourth, the user gains enormous flexibility in creating various treebank formats out of a given set of annotations, because only the differences between the underlying syntactic theories have to be formulated (desideratum B).

The annotation format plays a crucial role in this methodology. The remainder of this paper will describe this format, the formal properties of the conceptual level and the declaration, validation and processing of the transformation rules. An example using Japanese data will demonstrate the applicability of the methodology.

2.2 The annotation format

Common *annotation schemata* for treebanks rely on annotation categories and annotation structures. These can be distinguished in terms of constituent structure,
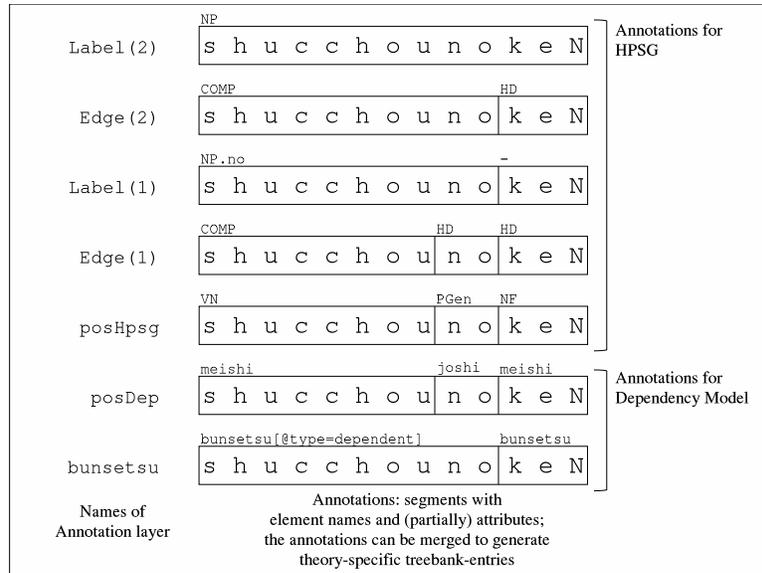
Names of Annotation layer	Annotations: segments with element names and (partially) attributes; the annotations can be merged to generate theory-specific treebank-entries
Label(2)	NP: `s h u c c h o u n o k e N` — Annotations for HPSG
Edge(2)	COMP: `s h u c c h o u n o` | HD: `k e N`
Label(1)	NP.no: `s h u c c h o u n o` | -: `k e N`
Edge(1)	COMP: `s h u c c h o u` | HD: `n o` | HD: `k e N`
posHpsg	VN: `s h u c c h o u` | PGen: `n o` | NF: `k e N`
posDep	meishi: `s h u c c h o u` | joshi: `n o` | meishi: `k e N` — Annotations for Dependency Model
bunsetsu	bunsetsu[@type=dependent]: `s h u c c h o u n o` | bunsetsu: `k e N`

Figure 2: Multiple annotations of the same textual data

functional structure and theory-specific annotations [11]. What is appropiate depends mainly on the language in question. For example, for languages with a greater degree of word-order freedom and a rich morphology functional structure is more suitable than constituent structure. Recently, non-purely syntactical annotational categories have been created as well, e.g. information structure [16].

The great variety leads to the challenge of how to represent the annotation schemata and the annotations in a computational way, i.e. as an *annotation format*. One solution is formulated within the *annotation graph model* [1]. Annotations are formally defined by a start point, an end point and a labeled arc between them. The genericity of this format allows for the representation of various theory- and language-specific annotation schemata.

The annotation format used in this paper uses the methodology of separately annotating the same original, textual data [17], i.e. the same sequence of characters is multiply annotated. An annotation may be used for an HPSG-treebank or another model, but its basis, i.e. the textual data, remains the same. Such an annotation is visualized in fig. 2.

Each annotation layer is identified by its name, e.g. `posHpsg`. An annotation is created by choosing one of the respective annotational categories, e.g. VN. No hierarchical or other constraints on configurations between annotations within one annotation layer, e.g. constituent structures or functional relations, exist. For ex-

ample, it is not necessary to declare a hierarchical configuration between the annotations `bunsetsu`, `meishi`, `hd` etc. of the character string `keN`, since such structures are declared on the conceptual level (cf. section 2.4). Moreover, there are no constraints for the creation of overlapping hierarchies between annotations on different layers. These structures will also be expressed on the conceptual level.

With this representation format, the same annotation layers are easy to reuse for several, theory-specific treebank schemata. For example, the parts of speech level of the HPSG annotations contains mostly notational variants of the parts of speech relevant for dependency annotations. New annotation layers whose role cannot be foreseen when creating a new treebank can be easily added, e.g. 'deep' levels like information structure. Already existing annotation layers which do not fit into the new model can easily be withdrawn. For example, only the parts of speech annotations of HPSG would be reused for the dependency annotations; other levels like `Edge` or `Label` might be withdrawn. The interchange of annotation layers takes place before constructing the tree structure of the result treebank. In this way it can be assured that the information contained in already created treebanks will not be lost during the transformation process.

2.3 Formal representation of annotations

During the annotation process, the data is represented in several XML-documents, sharing the same textual data. For the analysis of the configurations between the annotations on different annotation layers, a special-purpose tool has been developed [2]. First, for each XML-element node and attribute, Prolog facts are generated from the XML-documents:

> **node('posDep',11, 13, [1, 3], element('meishi')).**

This representation is an extension of an existing approaches to generate a Prolog representation out of XML- or SGML-documents [14]. The Prolog facts shown above provide information about the name of the annotation layer **'posDep'**, the range of characters contained **11,13**, the position in the annotation layer **[1,3]** and the name of the XML-element **element('meishi')**.[2] These Prolog facts are called within predicates which describe configurations between annotation units. All possible configurations are visualized in fig. 3.

Predicates like **identity**, **included_B_in_A**, **overlap_B** etc. are used to describe the cofigurations. For example, the annotations of the string `keN` in fig. 2 on the annotation layer `bunsetsu` and `edge1` can be regarded as identical. This would

---

[2]for XML-attributes, the representation is slightly different.

Configurations	Visualization
identity	
independence	
inclusion	
start point identity	
end point identity	
end point is starting point	
overlap	

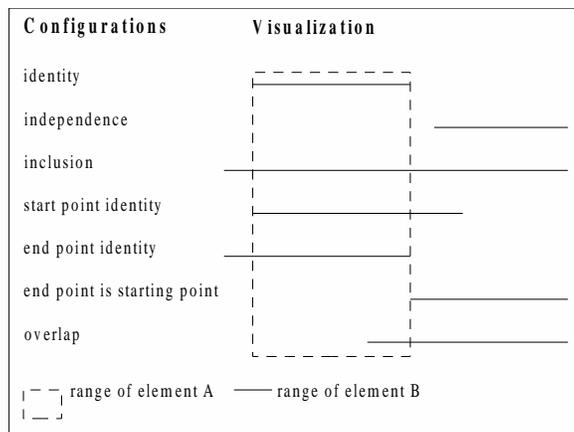range of element A   range of element B

Figure 3: Configurations between annotations on different layers

be analyzed with the predicate **identity**, which is used in the following predicate **chk_relation/6**:

> **chk_relation(identity,hd,'Edge1',bunsetsu,bunsetsu,L).**

chk_relation has 6 arguments: the configuration to be analyzed **identity**, the name of the element in the first annotation layer **hd**, the name of that layer **'Edge1'**, the name of the element in the second annotation layer **bunsetsu**, the name of that layer **bunsetsu**, and the name of the Variable **L**, which contains the output of the analysis.

The finite set of predicates is insufficient to express complex configurations between annotations of linguistic categories, e.g. all principles of an HPSG grammar fragment. For a corpus-based definition of a certain linguistic model, one might represent all information, i.e. textual data, linguistic categories and their properties on a formal, conceptual level [8]. On the other hand, the methodology proposed in this paper does not aim to create one complex model, but to relate models and make their differences explicit. For this purpose, a clear separation of textual data, theory-specific annotation levels and the model descriptions, with a small set of interconceptual properties which allows to refer to both data and conceptual level, seems to be more feasible. The key concept of the description of properties of categories on the conceptual level is that the predicates described above can be formulated at this level and applied to the annotated data.

2.4 Formal characteristics of the conceptual level

The conceptual level is represented by the *Resource Description Framework* (RDF) and its extension *RDF Schema* (RDFS) [4]. Its basic characteristics have been de-

scribed before [13], without a formal description in RDF. RDF allows to make statements about resources, using *triples* of the form **S(ubject) P(redicat) O(bject)**, e. g. **John loves Mary**, and to assign *properties* to them. In addition, RDFS supplies predefined statements to describe sets of resources, so-called *classes*, and to inter-relate them in a relation *subClassOf*, i.e. **John rdfs:subClassOf Human**. The concepts and conceptual models introduced in fig. 1 are examples of RDFS-classes. They are specified by their names, e. g. **Head**, and optionally by properties. For example a user-defined property **Head-identity-Bunsetsu** can be used to describe the relation **identity** between **Head** and **Bunsetsu**.
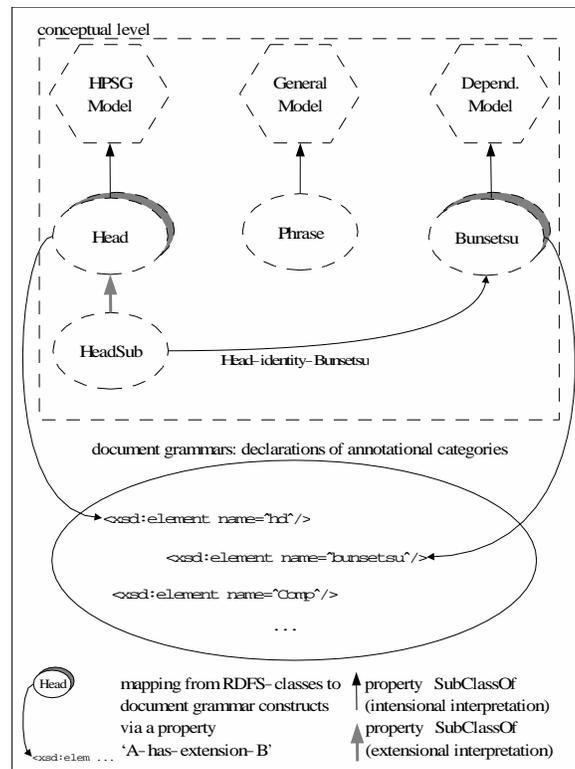

Figure 4: Visualization of the conceptual level

Normally, the formal interpretation of such statements is *intensional*: RDF(S) assures the logical thruthfulness of statements. No mechanisms are provided to examine whether the scope of the statements, i.e. the resources really exist and whether they have the characteristics described, because the characteristics of such a mechanism depend on the resources described. In this paper, such an *extensional interpretation* is being created, without violating the specifications of the RDF(S)

framework. The truthfulness of the extensional interpretation is validated with respect to the annotations and their formal representation as Prolog facts, which have been described in the previous sections. That is, RDF-based descriptions are interpreted as *prescriptions* for annotations. The basic idea is visualized in fig. 4, which contains the conceptual level of fig. 1 with its extensional interpretation. This is defined as follows:

- A RDFS-class **conceptualModel** is created. For each RDFS-class which should be interpreted as a conceptual model of a treebank, the predefined RDFS-property **rdfs:subClassOf** is defined, e. g. **HPSG-Model rdfs:subClassOf conceptualModel**. All RDFS-classes which represent a concept, e.g. a linguistic category of a conceptual Model, are RDFS-subclasses[3] of that model, e.g. **Head rdfs:subClassOf HPSG-Model**.

- The annotational categories are declared as document grammar constructs, formulated for example as XML Schema documents [15]. For each construct, a RDFS-class is created whose name is identical with that of the construct, e.g. the RDFS-class **hd** for the declaration of an XML-element `<xsd:element name="hd"/>`.

- The intensionally defined concepts, i.e. their RDFS-classes, are mapped to the document grammar constructs via a RDFS-property of the kind **a-has-extension-b**, for example **Head-has-extension-hd**. This property is assigned to the respective RDFS-classes by the RDFS-properties **rdfs:domain** and **rdfs:range**, e. g. **Head-has-extension rdfs:domain Head** and **Head-identity-Bunsetsu rdfs:range hd**.

- For all relations between concepts, i.e. linguistic categories which are to be interpreted as configurations between annotations (see section 2.3), a property of the kind **a-identity-b** is defined, for example **Head-identity-Bunsetsu**. The domain and the range of this definition is described again using the **rdfs:domain** and **rdfs:range** properties, e.g. **Head-identity-Bunsetsu rdfs:domain Head** and **Head-identity-Bunsetsu rdfs:range Bunsetsu**.

With these definitions, the declarative constructs of the data level, i. e. the document grammar constructs, are mapped onto the conceptual level, i. e. the RDFS-classes, as a basis for its extensional interpretation. With the RDFS-property **rdfs:subClassOf** it can be assured that this mapping is *prescriptive* for all subclasses. The extensional interpretation of subclasses, i.e. a validation of their properties

---

[3]More precisely, the relation between RDFS-classes which represent a conceptual model and RDFS-classes which represent a concept is **part-of**. For convenience, this difference is generalized.

with respect to the annotations, can be assured because their properties make use of the finite set of predicates described in section 2.3 (**included_B_in_A**, **end_point_A** etc.). By making use of the **rdfs:subClassOf** property, for each subclassified RDFS-class a **superclass** can be inferred which is mapped onto a document grammar construct. In this way, Prolog predicates like the ones illustrated below can be generated automatically from the conceptual level and be applied to the configurations of annotations. The predicates can be used to validate the interconceptual properties for **HeadSub**, namely **Head-identity-Bunsetsu**:

> **chk_relation(identity,hd,'Edge1',bunsetsu,bunsetsu,Resultlist).**
> **chk_relation(end_point_B,hd,'Edge1',bunsetsu,bunsetsu,Resultlist).**

The name of a conceptual model and the name of the subclasses which are the last to be interpreted extensionally serve as an input for the validation process. The name of a source model and a target model serve as an input for the transformation process. During transformation, all categories of the source model which have a relation **identity** with a category in the target model will be subject to the transformation. Nevertheless, not the names of the categories will be transformed, but the instances of the document grammar constructs, e.g. annotations like `hd -> bunsetsu`. Considering the annotations in fig. 2 and the conceptual level in fig. 4, this transformation process will be applied only to a subset of the `hd` annotations.

3 Sample application: An HPSG treebank model and a dependency treebank model for Japanese

A conceptual model of basic Japanese linguistic categories has been developed[4]. It is represented as an RDF Schema and contains about 200 morpho-syntactic and discourse related categories. These are related to general linguistic categories such as **Word** or **Phrase** which are part of the *Suggested Upper Merged Ontology* (SUMO) [10].

250 utterances from the Japanese Verbmobil corpus are used, which have been annotated with an HPSG-based annotation schema [6]. For the same textual data, an annotation schema with dependency categories has been used. (See fig. 5). It is part of a large-scale project for the creation of syntactically annotated Japanese corpora [7]. The dependency units, the so-called *bunsetsu*, are similar to chunks. In addition to SUMO, a conceptual level for HPSG and the dependency categories is being created.

---

[4]The model is accessible at http://coli.lili.uni-bielefeld.de/Texttechnologie/Forschergruppe/ sekimo/internet-praesentation/klassifikation/index.html
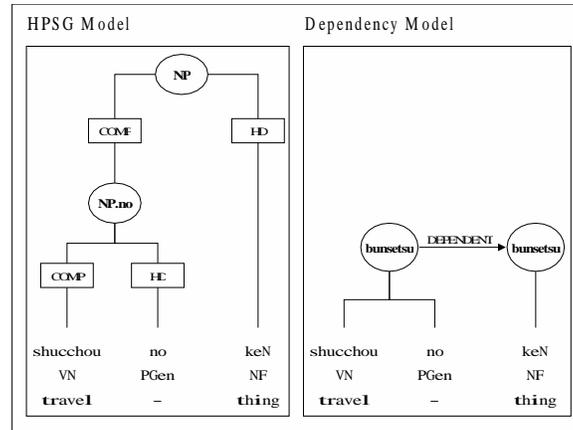
Figure 5: The phrase from fig. 1, visualized with respect to the HPSG model and the dependency model

As for the conceptual model of basic Japanese categories and the SUMO-categories, the theory-specific annotations of the HPSG-model or the dependency-model can be automatically transformed into the generalized annotation. For example, the **NP** node-labels of HPSG are generalized as **Phrase**, and the same holds for the **Bunsetsu** categories from the dependency model. Because the SUMO-categories and the Japanese categories do not supply annotation schemata, but concepts, the annotations categories are automatically created by using the name of a concept in small letters, e.g. **Phrase** would be the element name `phrase`. The relation **Head -> Phrase** would be used to transform annotations `hd -> phrase`.

4   Summary and prospects for future research

In this article, a methodology for the declaration of theory-specific concepts of treebanks and its application (validation, transformation) to annotated data has been introduced. The methodology is fully declarative, and it can be used for the description of a whole treebank or a subset of relevant categories from different models. Relations between the categories are declared as part of a conceptual model, and they are interpreted as configurations to be given in annotated data. The methodology relies on an annotation format with seperate annotation layers using the same textual data, a Prolog representation generated out of the annotations and a conceptual model with specific properties. All of these have been discussed in the article, and an example from Japanese has been given.

As a next step, the methodology will be applied to larger corpora with a higher

degree of complexity within annotations and conceptual models. The formal description of the conceptual level in this article is based upon the RDF-framework. More expressive languages like the *Web Ontology Language* (OWL) [9] rely upon the formal semantics of RDF and use the same syntax, i.e. an XML-serialization of their constructs. Hence, the next step will be the integration of the conceptual model in such a language, to be able to express more complex characteristics of linguistic theories. For the same purposes, the finite set of predicates used for the validation of conceptual properties within annotated data will be extended.

References

[1] S. Cotten and S. Bird. An Integrated Framework for Treebanks and Multi-layer Annotations. In *Proceedings of LREC 2002*, Las Palmas, 2002.

[2] D. Goecke, D. Naber, and A. Witt. Query von multiebenen-annotierten XML-Dokumenten mit Prolog. In U. Seewald-Heeg, editor, *Sprachtechnologie für die multilinguale Kommunikation - Textproduktion, Recherche, Übersetzung, Lokalisierung*, Sankt Augustin, 2003. Gardez!

[3] E. Hajicova and I. Kucerova. Argument / Valency Structure in PropBank, LCS Database and Prague Dependency Treebank: A Comparative Pilot Study. In *Proceedings of LREC 2002*, Las Palmas, Spain, 2002.

[4] P. Hayes and B. McBride. RDF Semantics. Technical report, W3C, 2003. http://www.w3.org/TR/rdf-mt/.

[5] N. Ide and R. Romary. Encoding Syntactic Annotation. In A. Abeillé, editor, *Building and Using Parsed Corpora*. Kluwer, Dordrecht, 2003.

[6] Y. Kawata and J. Bartels. *Stylebook for the Japanese Treebank in VERBMOBIL*, 2000. Verbmobil-Report 240.

[7] S. Kurohashi and M. Nagao. Building a Japanese Parsed Corpus while improving the Parsing System. In A. Abeillé, editor, *Building and Using Parsed Corpora*. Kluwer, Dordrecht, 2003.

[8] T. Lager. *A Logical Approach to Computational Corpus Linguistics*. PhD thesis, University of Goeteborg, Department of Linguistics, 1995.

[9] D. L. McGuinness and F. v. Harmelen. OWL Web Ontology Language Overview. Technical report, W3C, 2003. http://www.w3.org/TR/owl-features/.

[10] I. Niles and A. Pease. Towards a Standard Upper Ontology. In *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS-2001)*, Ogunquit, Maine, 2001.

[11] J. Nivre. What kinds of Trees grow in Swedish Soil? In *First Workshop on treebanks and linguistic theories*, Sozopol, Bulgaria, 2002.

[12] S. Oepen, D. Flickinger, K. Toutanova, and C. D. Manning. LinGO Redwoods. In *First Workshop on treebanks and linguistic theories*, Sozopol, Bulgaria, 2002.

[13] F. Sasaki and J. Pönninghaus. Testing Structural Properties in Textual Data: Beyond Document Grammars. *Literary and Linguistic Computing*, 18(1):89-100, 2003.

[14] C. M. Sperberg-McQueen, C. Huitfeldt, and A. Renear. Meaning and Interpretation of Markup. *Markup Languages*, 2(3):215-234, 2000.

[15] H. S. Thompson, D. Beech, M. Maloney, and N. Mendelsohn. XML Schema Part 1: Structures. Technical report, W3C, 2001. http://www.w3.org/TR/xmlschema-1/.

[16] Y. Tisheva and M. Dzhonova. Information Structure Level in TreeBanks. In *First Workshop on treebanks and linguistic theories*, Sozopol, Bulgaria, 2002.

[17] A. Witt. Meaning and Interpretation of Concurrent Markup. In *Proceedings of ALLC / ACH 2002*, Tübingen, Germany, 2002.

[18] F. Xia and M. Palmer. Converting Dependency Structures To Phrase Structures. In *Proceedings of HLT 2001*, San Francisco, 2001. Morgan Kaufmann.