Linköping Electronic Articles in
Computer and Information Science
Vol. 6(2001): nr ?

From Manual to Semi-automatic Semantic Annotation: About Ontology-based Text Annotation Tools

M. Erdmann
A. Maedche
H.-P. Schnurr
S. Staab

Institute AIFB
Karlsruhe University
76128 Karlsruhe, Germany

{erdmann, maedche, schnurr, staab}@aifb.uni-karlsruhe.de

http://www.aifb.uni-karlsruhe.de/WBS

Linköping University Electronic Press
Linköping, Sweden

http://www.ep.liu.se/ea/cis/2001/???/

Published on 30. December, 2001 by
Linköping University Electronic Press
581 83 Linköping, Sweden

Linköping Electronic Articles in
Computer and Information Science
ISSN 1401-9841
Series editor: Erik Sandewall

Typeset by the authors using LaTeX
Formatted using étendu *style*

Recommended citation:
<Authors>. <Title>. Linköping Electronic Articles in
Computer and Information Science, Vol. 6(2001): nr ?.
http://www.ep.liu.se/ea/cis/2001/???/. *30. December, 2001.*

This URL will also contain a link to the authors' home pages.

Abstract

Semantic Annotation is a basic technology for intelligent content and is beneficial in a wide range of content-oriented intelligent applications, esp. in the area of the Semantic Web. In this paper we present our work in ontology-based semantic annotation, which is embedded in a scenario of a knowledge portal application. Starting with seemingly good and bad manual semantic annotation, we describe our experiences made within the KA^2-initiative. The experiences gave us the starting point for developing an ergonomic and knowledge base-supported annotation tool. Furthermore, the annotation tool described are currently extended with mechanisms for semi-automatic information-extraction based annotation. Supporting the evolving nature of semantic content we additionally describe our idea of evolving ontologies supporting semantic annotation.

This paper has been presented at the COLING-2000 Workshop on *Semantic Annotation and Intelligent Content*, Centre Universitaire, Luxembourg, 5.-6. August, 2000.

1 Introduction

The KA^2-initiative (Knowledge Annotation initiative of the Knowledge Acquisition community) was launched at EKAW in 1997 in order to provide semantic access to information stored in web pages in the WWW. It built on manual semantic annotation for integration and retrieval of facts from semantically annotated web pages, which belonged to members of the knowledge acquisition community (Decker et al., 1999; Benjamins et al., 1999). The initiative recently developed into a more comprehensive concept viz. the KA^2 community portal, which allows for providing, browsing and retrieving information through various means of ontology-based support (Staab et al., 2000). All along the way, the usage of semantic annotation as the underpinning for semantics-based fact retrieval, integration, and presentation has remained one of the major cornerstones of the system.

The content of the paper is organized as follows. In Section 2 we start with a brief introduction to our notion of a community web portal to set up the context of our use of semantic annotations. Then, we present the practical problems we have encountered with manual annotations and the lessons learned from these experiences (cf. Section 3). In Section 4 the development of annotation tools is sketched that facilitate manual semantic annotation by following ergonomic considerations about the process that someone who is annotating information goes through and inferencing support that provides a comprehensive view on what has been annotated, so far. The development of an information extraction-based system for semi-automatic annotation that proposes annotations to the human who is performing annotations is presented in Section 5. We conceive semantic annotation as a cyclic process between the actual task of annotating documents and the development and adaptation of a *domain ontology*. Incoming information that is to be annotated does not only require some more annotating, but also continuous adaptation to new semantic terminology and relationships. This cyclic process of evolving ontologies is shown in Section 6. Our objective here is to give the reader a comprehensive picture of what semantic annotation has meant in our application and where it is heading now.

2 Scenario: Semantic Community Web Portal

Community web portals serve as high quality information repositories for the information needs of particular communities on the web. A prerequisite for fulfilling this role is the accessibility of information. In *community* portals this information is typically provided by the users of the portal, i.e. the portal is driven *by* the community *for* the community. We have been maintaining a web portal for the Knowledge Acquisition community[1] and, thus, have gained some experience with the difficulties of providing information for that portal by semantic annotations.

We here give only a very brief sketch of the KA community web portal. A broader introduction to the methods and tools developed in this context can be found in (Staab et al., 2000). The portal's main component is Ontobroker (Decker et al., 1999), that uses ontologies to provide an integrated view on distributed, heterogenous information sources. The ontology is the means for capturing domain knowledge in a generic way that provides a commonly agreed understanding of a domain, which may be reused and shared within communities or applications. The ontology can be used to semantically annotate web pages that are accessed by Ontobroker

The Ontobroker system consists of (i) a crawling component, (ii) a knowledge base, (iii) an inference engine, and (iv) a query interface. The crawler collects in-

[1] `http://ka2portal.aifb.uni-karlsruhe.de`

formation contained in registered web pages and stores it in the knowledge base. The HTML pages are manually annotated with special semantic tags, a proprietary extension to HTML that is compatible with common web browsers. This annotation language is presented in the next section. Thus, the web crawler establishes the core of the knowledge base, that is enhanced by applying axioms from the ontology to these ground facts. The ontology is represented in Frame Logic (Kifer et al., 1995), an object-oriented and logics-based language. Thus, axioms can be formulated using a subset of first order logic statements including object oriented modelling primitives. Finally, the information stored in the knowledge base or derived by the inference engine can be accessed using Frame Logic queries.

3 Manual Semantic Annotations

3.1 HTML-A

The main source of information for the KA portal stems from distributed web pages maintained by members of the KA community. These web pages have been manually annotated to explicitely represent the semantics of their contents (cf. Figure 1). Since a huge amount of relevant information for most communities is represented in HTML, we chose to enhance HTML with few semantically relevant extensions. The resulting annotation language HTML-A (Fensel et al., 1998) adds to HTML primitives for tagging instances of concepts, for relating these instances, and for setting their properties, i.e. the ontology serves as a schema for semantic statements in these pages. For all these primitives the HTML anchor tag `<A>` has been extended with a special attribute `onto`. This decision implies that the original information sources hardly have to be changed to provide semantically meaningful information. The semantic tags are embedded in the ordinary HTML text in such a way that standard browsers can still process the HTML pages and, at the same time, Ontobroker's crawler can extract the semantic annotations from them. This kind of semantic annotation resembles Knuth's literate programming (Knuth, 1984), where few semantically relevant and formal statements are embedded in unstructured prose text. In Ontobroker, objects (instances of concepts) are uniquely identified by a URI, i.e. resources in the web are interpreted as surrogates for real objects like persons, organizations, and publications. To associate (in HTML) such an object with a concept from the ontology one of the following statements can be made in the HTML source.

```
<A onto="'http://www.aifb.uni-karlsruhe.de/studer':Researcher"></A>
<A onto="'www9':InProceedings"></A>
<A onto="page:Institute"></A>
```

In the schema <A onto="O:C"></A> of these expressions O represents the instance and C represents the concept. O can either be a global URI, a local part of a URI (that is expanded by the crawler to a global one), or one of the special keywords `page`, `body`, `href`, or `tag`. These special keywords represent resources relative to the current tag and the current web page, e.g. the keyword `page` represents the URI of the webpage of this statement. The following statements both define formally the value of the `name` attribute of the object represented by the current `page`:

```
<A onto="page[name='Rudi Studer']"></A>
<A onto="page[name=body]">Rudi Studer</A>
```

The keyword `body` refers to the content of the anchor tag. Thus, the actual information is rendered by a web browser and at the same time interpreted formally by the crawler. Including semantics in this way into HTML pages reduces redundancy and enhances maintainability, since changes in the prose part of the page are immediately reflected in the formal part, as well.

To establish relationships between two objects similar statements can be made, since binary relations can be modelled as attributes:

```
<A onto="page[affiliation='http://www.aifb.uni-karlsruhe.de']"></A>
<A onto="page[affiliation=href]"
   href="http://www.aifb.uni-karlsruhe.de">
        Institut AIFB</A>
<A onto="page[authorOf=href]"
   href="publications.html#www9">
        Semantic Community Web Portals</A>
```

The `href` keyword defines the target of the hypertext link as an object representing the value of the attribute. If this link is relative it is expanded to its global URI before putting the facts into the knowledge base.

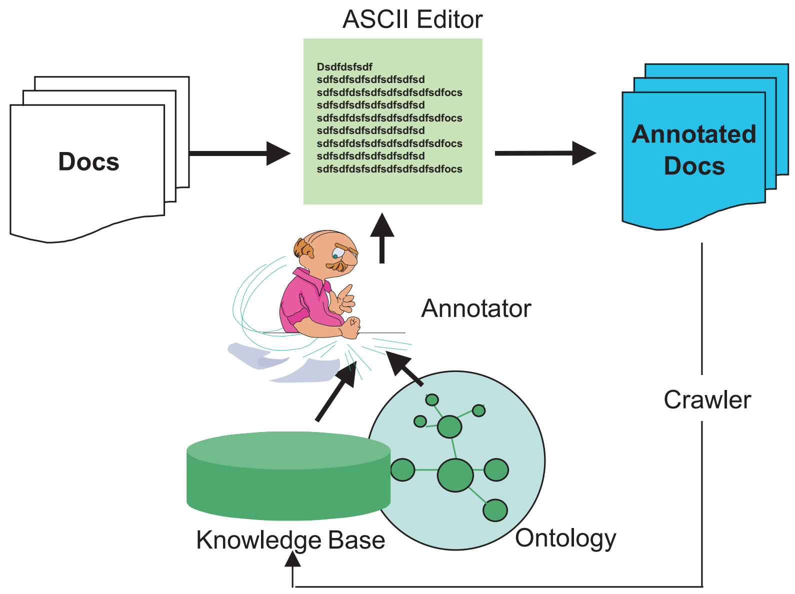


Figure 1: Manually annotating HTML pages with semantic information.

3.2 Experiences

Our experiences with the KA2 initiative were quite disappointing, concerning the information providing process. There were about 30 people willing to provide information from their web pages to Ontobroker. About 15 accepted and were (more or less) able to annotate their pages. The other 15 needed rather extensive support from the Ontobroker team. One of our students prepared annotated versions of their homepages. Since the annotation task was not supported by any tool, severe problems appeared. First of all, a lot of annotations were simply syntactically incorrect, i.e. have been rejected by the parser, for reasons like missing brackets or quotation marks. This problem has been remedied by providing a syntax checker that is available online and tests annotated pages for syntactic correctness.

A second major problem concerned terminology. Since the ontology was fixed from the beginning, the annotations had to strictly conform to the concept and attribute names defined in the ontology. Typing errors, e.g. `online-Version` instead of `onlineVersion`, were the most prominent in this category.

The last group of problems deals with the semantics of the annotations:

- The class of some objects had been defined in a too general manner, e.g. most publications in the KA2 knowledge base have been categorized simply as `Publication` instead of `JournalArticle`, `TechnicalReport` or another more specific concept. The intention of some ontological terms have not been completely understood by some providers. This resulted in things like the classification of a web page containing a list of publications of some researcher to be defined as the value of his `publication` attribute. This set valued attribute was intended to contain a set of `publication` objects and not a single container object. Application of axioms in the ontology yielded the fact that this publication list was categorized as a `Publication` which was not intended. On the other hand, querying the knowledge base for all publications of this researcher resulted in an acceptable answer, namely a link to this list page. Additionally, each object should be identified by a single URI. But we experienced major problems with the use of object identifiers to refer to certain objects in an unambiguous way.

- Often, instead of introducing a URI or referring to an existing object identifier to denote an object, information providers simply used text from the web page, e.g. a co-author of a publication was often identified by a string like “John Doe” instead of the URI for his home page.

- Similarly, even if object identifiers, i.e. URIs, were used to refer to remote objects like co-authors, these URIs often did not match. An implication of these mismatches is the creation of several objects that should have been unified into one, e.g. our colleague Dieter Fensel at some time was represented in the knowledge base by three object identifiers, each denoting an object with some information linked to it. These information could be integrated only after the sources of the mismatch had been identified.

- Finally, the overall quantity of semantic annotations could have been larger, i.e. although some information was textually present on the annotated web pages, this information has not been annotated and, thus, was invisible for Ontobroker and for its users. This problem especially occured on pages annotated by our student, due to her lack of deep domain knowledge.

3.3 Lessons learned

After reviewing the different types of problems, we came up with a set of lessons learned that may be summarized by:

- Keep the ontology simple and explain its meaning!

- Support annotators with interactive, graphical tools!

  – to help avoid syntax errors and typos of ontological entities,

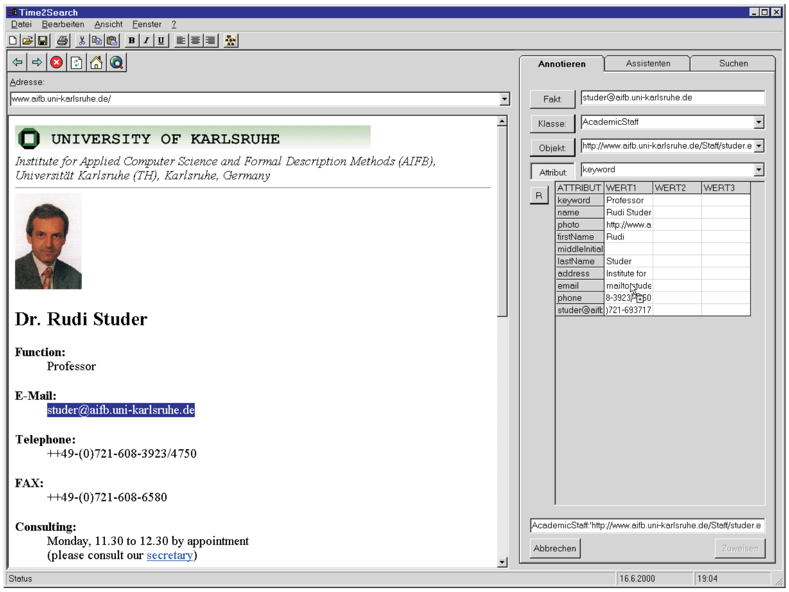
  – to help to correctly choose the most specific concepts for instances, and

  – to provide a list of all known objects of a certain concept to reduce false co-references (with a kind of repository of objects)

- Allow importing information from other sources to avoid annotations where possible, e.g. import BiBTeX-files for the publications of researchers.

4 Ergonomic and knowledge base-supported Annotation

Targeting to an ergonomic and intuitive support of the annotation task within documents, we developed the annotation tool. It allows the quick annotation of facts within any document by tagging parts of the text and semantically defining its meaning via interacting with the dialog shown in the screenshot of Figure 2. To illustrate the annotation process using the annotation tool, we sketch in the following a small annotation scenario using the annotation tool.



www.aifb.uni-karlsruhe.de/

UNIVERSITY OF KARLSRUHE

Institute for Applied Computer Science and Formal Description Methods (AIFB), Universität Karlsruhe (TH), Karlsruhe, Germany

Dr. Rudi Studer

Function:
Professor

E-Mail:
studer@aifb.uni-karlsruhe.de

Telephone:
++49-(0)721-608-3923/4750

FAX:
++49-(0)721-608-6580

Consulting:
Monday, 11.30 to 12.30 by appointment
(please consult our secretary)

Fakt: studer@aifb.uni-karlsruhe.de
Klasse: AcademicStaff
Objekt: http://www.aifb.uni-karlsruhe.de/Staff/studer.e
Attribut: keyword

ATTRIBUT	WERT1	WERT2	WERT3
keyword	Professor		
name	Rudi Studer		
photo	http://www.a		
firstName	Rudi		
middleInitial			
lastName	Studer		
address	Institute for		
email	mailto:stude		
phone	8-3923/[illegible]50		
studer@aifb	)721-693717		

AcademicStaff:'http://www.aifb.uni-karlsruhe.de/Staff/studer.e

16.6.2000 19:04

Figure 2: Annotation Dialog.

Given an ontology, the annotation process usually starts with tagging one or more phrases in the document. This selection is indicated in the fact list in the left column of Figure 2. The user picks the desired word or text part in the fact list and selects the appropriate concept in the ontology, depicted in the second column of Figure 2 as an explorer tree view. In our example, `studer@aifb.uni-karlsruhe.de` is chosen in the selected fact list and the concept Researcher is selected in the ontology. Therefore, the annotation tool supports the intuitive and correct choice of the most specific concept for the selected instance. Now, as described in further detail in (Schnurr & Staab, 2000) the concept choice of the user triggers the F-Logic inference engine to search for all known objects of this certain concept in the knowledge base. The third column in Figure 2 shows these objects, in our example a list of researchers. Thus, the user may insert references to the known objects or add a new one to the knowledge base. In our example, the user selects `Researcher_01`, the primary key of the researcher with the last name `Studer`, to add his EMAIL address to the knowledge base. If there would be no corresponding known object in the knowledge base, the user would have to select `Researcher_Neu` to add a new instance. Thereby, the system automatically creates a primary key for that new object. In the fourth column of Figure 2, the attributes and relations of the highlighted concept are shown. The user chooses the appropriate attribute or relation, in our example the attribute EMAIL, and thereby annotates the selected part of the document, namely `studer@aifb.uni-karlsruhe.de` in our example. Selecting a relation, the right most column in Figure 2 gives a list of possible instances, where the relation may point to. The user picks up the appropriate instance and thus, links both concepts with the selected relation. The dialog shown in Figure 2 offers the whole range of annotation support to the user. As our experience has shown, a simple concept-attribute-value structure underlies the majority of all annotation cases. To support this common and highly standardized task, we developed a quick annotation bar. After highlighting a part of the text in the document, the user may select a concept, the appropriate existing instance or a new one, and an attribute or a relation. In just picking the desired choices from the pull down lists of the annotation bar, the user can quickly annotate the document. With the features of our annotation tool, we support annotators with an interactive, graphical means helping to avoid syntax errors. We support them in choosing the most appropriate concepts for instances and provide an object repository to identify existing instances. As indicated in Figure 3, the annotation tool integrates the ontology and the knowledge base into the editing environment to allow for ergonomic and knowledge base-supported annotating.

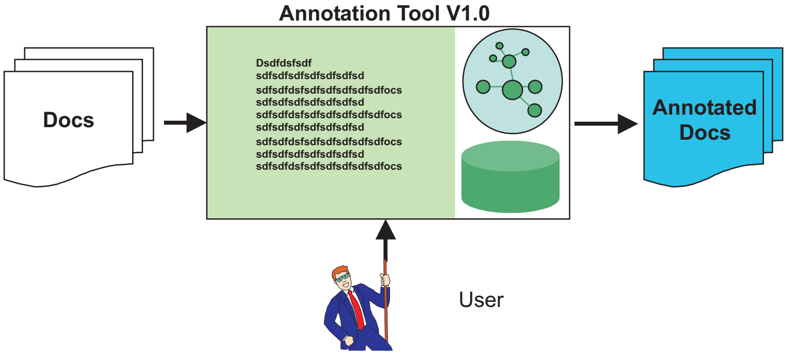


Figure 3: Ergonomic and inference-supported Annotation.

5 Semi-Automatic Annotation

Based on our experiences and the existing annotation tool for supporting ontology-based semantic annotation of texts, we now approach semi-automatic annotation of natural language texts. We conceive an information extraction-based appraoch for semi-automatic annotation, which has been implemented on top of SMES (Saarbrücken Message Extraction System), a shallow text processor for German (cf. (Neumann et al., 1997)). This is a generic component that adheres to several principles that are crucial for our objectives. *(i)*, it is fast and robust, *(ii)*, it realizes a mapping from terms to ontological concepts, *(iii)* it yields dependency relations between terms, and, *(iv)*, it is easily adaptable to new domains.[2]

We here give a short survey on SMES in order to provide the reader with a comprehensive picture of what underlies our system. The architecture of SMES comprises a *tokenizer* based on regular expressions, a *lexical analysis* component including a *word and a domain lexicon*, and a *chunk parser*. The tokenizer scans the text in order to identify boundaries of words and complex expressions like “$20.00” or “Mecklenburg-Vorpommern”[3], and to expand abbreviations. The lexicon contains more than 120,000 stem entries and more than 12,000 subcatego-

[2] The interlinkage between the information extraction system SMES and domain ontologies is described in further detail in (Staab et al., 1999).

[3] Mecklenburg-Vorpommern is a region in the north east of Germany.

rization frames describing information used for lexical analysis and chunk parsing. Furthermore, the domain-specific part of the lexicon associates word stems with concepts that are available in the concept taxonomy. *Lexical Analysis* uses the lexicon to perform, *(1)*, morphological analysis, *i.e.*, the identification of the canonical common stem of a set of related word forms and the analysis of compounds, *(2)*, recognition of name entities, *(3)*, retrieval of domain-specific information, and, *(4)*, part-of-speech tagging. While the steps (1),(2) and (4) can be a viewed as standard for information extraction approaches (cf. (Appelt et al., 1993; Neumann et al., 1997)), the step (3) is of specific interest for our annotation task. This step associates single words or complex expressions with a concept from the ontology if a corresponding entry in the domain-specific part of the lexicon exists. E.g., the expression "Hotel Schwarzer Adler" is associated with the concept Hotel.

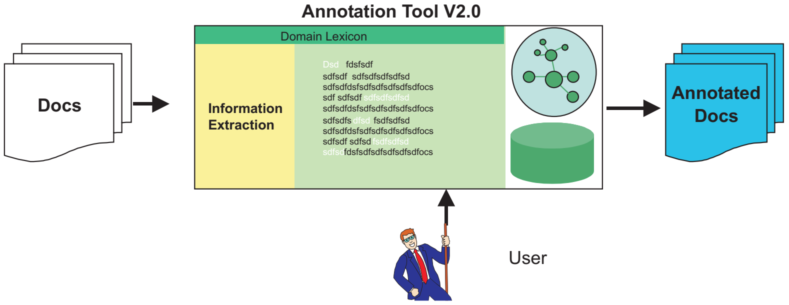


Figure 4: Semi-automatic Annotation.

SMES includes a *chunk parser* based on weighted finite state transducers to efficiently process phrasal and sentential patterns. The parser works on the phrasal level, before it analyzes the overall sentence. Grammatical functions (such as subject, direct-object) are determined for each dependency-based sentential structure on the basis of subcategorizations frames in the lexicon. Our primary output derived from SMES consists of *dependency relations* (Hudson, 1990) found through lexical analysis (compound processing) and through parsing at the phrase and sentential level. Thereby, the grammatical dependency relation need not even hold directly between two conceptually meaningful entities. For instance, in the sentence "The `Hotel Schwarzer Adler` in `Rostock` celebrates Christmas.", "Hotel Schwarzer Adler" and "Rostock", the concepts of which appear in the ontology as Hotel and City, respectively, are not directly connected by a dependency relation. However, the preposition "in" acts as a mediator that incurs the conceptual pairing of Hotel with City.

Figure 4 depicts the architecture of the semi-automatic annotation tool. Incoming documents are processed using the information extraction system SMES. SMES associates single words or complex expressions with a concept from the ontology, connected through the domain lexicon linkage. Recognized concepts and dependency relations between concepts are highlighted as suggested annotations. This mechanism has the advantage that all relevant information in the document with regard to the ontology is recognized and proposed to the annotator. The actual process of annotation is delegated to the annotation tool described in section 4.

6 Evolving Ontologies

In the previous sections 3, 4 and 5 we have abstracted from the interlinkage between evolving ontologies and the different annotation mechanisms. However, in any realistic application scenario, incoming information that is to be annotated does not only require some more annotating, but also continuous adaptation to new semantic terminology and relationships. Terms evolve in their meanings, or take on new meanings as new technologies are developed, and as existing ones evolve.

The abstraction from the interlinkage between annotation and evolving ontologies resulted in problems, *(i)* if the meaning of ontological elements changed, *(ii)* if the elements in the ontology became unnecessary and have been eliminated, or *(iii)* if new elements have been added to the ontology. Our experiences have shown that annotation and ontology development and maintenance must be considered as a cyclic process. Thus, in a realistic annotation scenario a feedback loop and tight integration is required, so that new conceptual structures can be added to the ontology for supporting the actual task of annotating documents towards evolving ontologies.

Manual Ontology Engineering. Starting with manual semantic annotation as described in Section 3 the ontology was represented as an ASCII file in FLogic. There was only few documentation, no browsing was possible, and it was fixed from the beginning. The process of manual semantic annotation didn't incorporate the ontology, so that typing errors were not unusual. One of the more fundamental problems were incorrect coreferences, because no interlinkage between new annotated facts and existing facts was supported.

As described in Section 4 our experiences showed us the necessity for ergonomic and knowledge base-supported annotation. We developed a tool which includes the domain ontology directly in its interface, defines automatically identifiers and references to existing facts contained in the knowledge base. We also developed an ontology engineering environment OntoEdit[4] supporting the ontology engineer in modeling conceptual structures.

Semi-Automatic Ontology Engineering. Currently we are working on the tight integration between semantic annotation and ontology engineering. Lexical resources are directly mapped onto concepts and relations contained in the ontology. The coding nature of ontologies makes it necessary to account for changes. Hence, we have been developing methods that propose new conceptual structures to the maintainer of the ontology (cf. (Maedche & Staab, 2000a)). In parallel, linguistic resources are gathered, which connect the conceptual structures with the information extraction system. The information extraction system supports the engineering of evolving ontologies as well as the process of extracting annotation-relevant information. The underlying idea is that acquired domain specific knowledge and linguistic resources are connected to natural language using a tight interplay between ontology and domain lexicon.

In (Maedche & Staab, 2000b) we describe our work in semi-automatic engineering and learning of domain ontologies from text. A comprehensive architecture lays the foundation for acquiring domain ontologies and linguistic resources. The main components of the architecture are *(i)* the Text & Processing Management, *(ii)* the Information Extraction Server (SMES), *(iii)* a Lexical Database and Domain Lexicon, *(iv)* a Learning Module, and *(v)* the Ontology Engineering Environment OntoEdit. The architecture has been fully implemented in the "Ontology Learning"-Environment Text-To-Onto and lays the foundation for supporting the development of evolving ontologies from text.

[4] A comprehensive description of the ontology engineering environment OntoEdit and the underlying methodology is given in (Staab & Maedche, 2000).

7 Related Work

An approach similar to our first tries of annotating HTML using ontologies has been developed at the University of Maryland. The SHOE system (Luke et al., 1997) defines additional tags that can be embedded in the body of HTML pages. In SHOE there is no direct relationship between the new tags and the original text of the page, i.e. SHOE tags are not annotations in a strict sense. In (Heflin et al., 1999), the authors report of similar observations of the "annotation" process as we present here.[5]

When talking about semantic annotations, terms like XML (Bray et al., 1998) and RDF (Lassila & Swick, 1999) must not be absent. Especially XML (Extensible Markup Language) earned a lot of attention in the last two years since its standardisation. XML allows the definition of individual tags that can be interpreted according to the user's will. E.g. XHTML represents an HTML-like vocabulary to describe the layout of web pages for browsers, SMIL defines tags that describe complete multimedia documents, or with XMLNews-tags the text of news can be annotated with rich semantic meaning such as the location and date of an event. Pure XML vocabularies like these are not sufficient as means for representing deep semantics, but they can be complemented by ontologies to achieve a flexible and well understood way to represent and transfer content (via XML) and at the same time to embed the represented facts in a formal and machine interpretable model of discourse (via the ontology). In (Erdmann & Studer, 1999) we show how to establish such a close coupling automatically.

We expect the relationship of semantic annotations or semantic metadata with ontologies to be central for the success of semantic information processing in the future. The Resource Description Framework (RDF), an (XML-based) representation format for meta data defined by the W3C could take a central part in this development, since an ontology representation mechanism has been defined on top of the basic RDF primitives. A core language introducing notions of classes and relationships has been proposed to the W3C as RDFS (Brickley & Guha, 1999). Even richer languages for more elaborate modeling primitives like symmetric relationships, part-of relations, or Description-Logic-like subsumption hierarchies were proposed in (Erdmann et al., 2000) or (Horrocks et.al., 2000). Thus, RDF could become *the means* to represent metadata *and* ontologies in an open, widely "spoken" representation and interchange format.

Concerning our mechanisms for semi-automatic semantic annotation described in Section 5 there has been done only little research. Pustejovsky et al. (Pustejovsky et al., 1997) describe their approach for semantic indexing and typed hyperlinking. As in our approach finite state technologies support lexical acquisition as well as semantic tagging. The goal of the overall process is the generation of so called *lexical webs* that can be utilized to enable automatic and semi-automatic construction of web-based texts.

In (Bod et al., 1997) approaches for learning syntactic strctures from syntactically tagged corpus has been transferred to the semantic level, too. In order to tag a text corpus with type-logical formulae, they created tool environment called SEMTAGS for semi-automatically enriching trees with semantic annotations. SEMTAGS incrementally creates a first order markov model based on existing annotations and proposes a semantic annotation of new syntactic trees. The authors report promising results: After the first 100 sentences of the corpus had been annotated, SEMTAGS already produced the correct annotations for 80% of the nodes for the immediately subsequent sentences.

[5] For a further comparison of several ways to represent knowledge in the WWW (often by means like semantic annotations) refer to (van Harmelen & Fensel, 1999).

8 Discussion

Based on the KA^2 community portal scenario we have shown in Section 3 how information has been provided in the beginning. Our lessons learned from this experience gave us a starting point for developing more advanced and more user friendly methods for semantically annotating documents. The methods are combined with an information extraction system that semi-automatically proposes new annotations to the user. Our experiences have shown that semantic annotation and ontology engineering must be considered a cyclic process.

In the future much work remains to be done. First,we will have to build an integrated system of annotation and ontology construction. This system will combine knowledge base-supported, ergonomic annotation, with an environment and methods for ontology engineering and learning from text supporting evolving ontologies. Second, we have to evaluate our annotation mechanisms. Evaluation in our annotation architecture can be splitted into several sub-evaluation phases: ergonomic evaluation, evaluation of the ontology, evaluation of the semi-automatic suggestions, evaluation of the user's annotations. Third, we will support the RDF standard for representing metadata on the web, representing both ontologies and generated annotated facts in RDF(S). This standard will make annotated facts reusable and machine-processable on the web (Decker et al., 2000).

Acknowledgements. We thank Stefan Decker for initiating the first version of an annotation tool. We thank our students Mika Maier-Collin and Jochen Klotzbuecher for the Annotation Tool; Dirk Wenke for the Ontology Engineering Environment OntoEdit; Raphael Volz for the Ontology Learning Environment Text-To-Onto; DFKI language technology group, in particular Guenter Neumann, for their dedicated support in using SMES. This work was partially supported by grant GETESS and by Ontoprise GmbH.

References

Appelt, D., Hobbs, J., Bear, J., Israel, D., & Tyson, M. (1993). FASTUS: A finite state processor for information extraction from real world text. In *Proceedings of IJCAI-93*, Chambery, France.

Benjamins, R., Fensel, D., & Decker, S. (1999). KA2: Building Ontologies for the Internet: A Midterm Report. *International Journal of Human Computer Studies*, 51(3):687.

Bod, R., Bonnema, R., & Scha, R. (1997). Data-oriented semantic interpretation. In *In Proceedings of the Second International Workshop on Computational Semantics (IWCS), Tilburg, 1997.*

Bray, T., Paoli, J., & Sperberg-McQueen, C. (1998). Extensible markup language (XML) 1.0. Technical report, W3C. http://www.w3.org/TR/1998/REC-xml-19980210.

Brickley, D. & Guha, R. (1999). Resource description framework (RDF) schema specification. Technical report, W3C. W3C Proposed Recommendation. http://www.w3.org/TR/PR-rdf-schema/.

Decker, S., Erdmann, M., Fensel, D., & Studer, R. (1999). Ontobroker: Ontology Based Access to Distributed and Semi-Structured Information. In Meersman, R. et al. (Eds.), *Database Semantics: Semantic Issues in Multimedia Systems*, pages 351–369. Kluwer Academic Publisher.

Decker, S., Jannink, J., Mitra, P., Staab, S., Studer, R., & Wiederhold, G. (2000). An information food chain for advanced applications on the www. In *Proceedings of the Fourth European Conference on Research and Advanced Technology for Digital Libraries*.

Erdmann, M., Maedche, M., Staab, S., & Decker, S. (2000). Ontologies in RDF(S). Technical Report 401, Institute AIFB, Karlsruhe University.

Erdmann, M. & Studer, R. (1999). Ontologies as Conceptual Models for XML Documents. In *Proceedings of the 12th International Workshop on Knowledge Acquisition, Modelling and Mangement (KAW'99), Banff, Canada, October*.

Fensel, D., Decker, S., Erdmann, M., & Studer, R. (1998). Ontobroker: The Very High Idea. In *Proceedings of the 11th International Flairs Conference (FLAIRS-98), Sanibel Island, Florida, May*.

Heflin, J., Hendler, J., & Luke, S. (1999). Applying Ontology to the Web: A Case Study. In *Proceedings of the International Work-Conference on Artificial and Natural Neural Networks, IWANN'99*.

Horrocks et.al., I. (2000). The ontology interchange language oil: The grease between ontologies. Technical report, Dep. of Computer Science, Univ. of Manchester, UK/ Vrije Universiteit Amsterdam, NL/ AIdministrator, Nederland B.V./ AIFB, Univ. of Karlsruhe, DE. http://www.cs.vu.nl/˜dieter/oil/.

Hudson, R. (1990). *English Word Grammar*. Basil Blackwell, Oxford.

Kifer, M., Lausen, G., & Wu, J. (1995). Logical foundations of object-oriented and frame-based languages. *Journal of the ACM*, 42.

Knuth, D. E. (1984). Literate programming. *The Computer Journal*, 27:97–111.

Lassila, O. & Swick, R. (1999). Resource description framework (RDF) model and syntax specification. Technical report, W3C. W3C Recommendation. http://www.w3.org/TR/REC-rdf-syntax.

Luke, S., Spector, L., Rager, D., & Hendler, J. (1997). Ontology-based Web Agents. In *Proceedings of First International Conference on Autonomous Agents, LNCS*.

Maedche, A. & Staab, S. (2000a). Discovering conceptual relations from text. In *Proceedings of ECAI-2000*. IOS Press, Amsterdam.

Maedche, A. & Staab, S. (2000b). Semi-automatic engineering of ontologies from text. In *Proceedings of the 12th Internal Conference on Software and Knowledge Engineering. Chicago, USA, July, 5-7, 2000*. KSI.

Neumann, G., Backofen, R., Baur, J., Becker, M., & Braun, C. (1997). An information extraction core system for real world german text processing. In *In Proceedings of ANLP-97*, pages 208–215, Washington, USA.

Pustejovsky, J., Boguraev, B., Verhagen, M., Buitelaar, P., & Johnston, M. (1997). Semantic indexing and typed hyperlinking. In *Proceedings of AAAI Spring Symposium, NLP for WWW*.

Schnurr, H.-P. & Staab, S. (2000). A proactive inferencing agent for desk support. In *Proceedings of the AAAI Symposium on Bringing Knowledge to Business Processes*, Stanford, CA, USA. AAAI Technical Report, Menlo Park.

Staab, S., Angele, J., Decker, S., Erdmann, M., Hotho, A., Maedche, A., Studer, R., & Sure, Y. (2000). Semantic Community Web Portals. In *Proceedings of the 9th World Wide Web Conference (WWW-9), Amsterdam, Netherlands*.

Staab, S., Braun, C., Düsterhöft, A., Heuer, A., Klettke, M., Melzig, S., Neumann, G., Prager, B., Pretzel, J., Schnurr, H.-P., Studer, R., Uszkoreit, H., & Wrenger, B. (1999). GETESS — searching the web exploiting german texts. In *Proceedings of the 3rd Workshop on Cooperative Information Agents*, LNCS, Berlin. Springer.

Staab, S. & Maedche, A. (2000). Ontology engineering beyond the modeling of concepts and relations. In *Proceedings of the ECAI'2000 Workshop on Application of Ontologies and Problem-Solving Methods*.

van Harmelen, F. & Fensel, D. (1999). Practical Knowledge Representation for the Web. In *Proceedings of the IJCAI Workshop on Intelligent Information Integration*.