Proceedings of the IASTED International Conference
SOFTWARE ENGINEERING
February 17-19, 2004, Innsbruck, Austria

E-Learning Infrastructure for Software Engineering Education: Steps in Ontology Modeling for SWEBOK

Cornelius Wille, Reiner R. Dumke
Otto-von-Guericke-University of Magdeburg,
Faculty of Computer Science,
Postfach 4120, 39016 Magdeburg, Germany,
(dumke,wille)@ivs.cs.uni-magdeburg.de

Alain Abran, Jean Marc Desharnais,
École de Technologie Supérieure - ETS
1100 Notre-Dame Ouest, H3C 1K3 Montréal
Québec , Canada,
(aabran,jmdeshar)@ele.etsmtl.ca

Abstract: The Guide to the Software Engineering Body of Knowledge (SWEBOK) has been developed to represent an international consensus formed through broad public participation in the review process and is now close to final approval as ISO/IEC TR 19759. This guide constitutes an integrated structuring of a large set of software engineering concepts developed individually over the past forty years from a large number of distinct viewpoints. The absence of a recognized consensus on software engineering terminology has been a challenging task in building the SWEBOK Guide and in achieving this international consensus. This paper presents a first ontological approach to building domain-specific ontologies as a part of the Semantic Web, and shows how it can be used to build the SWEBOK ontology and to increase its internal consistency and clarity. Finally, new ideas on how a SWEBOK ontology can help in developing an e-learning system on software engineering are presented.

Key Words: Software Engineering Body of Knowledge, SWEBOK, ISO/IEC TR 19759, Ontology, E-Learning

1 SWEBOK

Gaining the widest possible consensus on the content of a Software Engineering Body of Knowledge (SWEBOK) is an essential step toward developing the software engineering profession. Without such a consensus, no licensing examination can be validated, no curriculum can prepare an individual for an examination and no criteria can be formulated for accrediting a curriculum. The IEEE Computer Society has championed the development of such an international consensus on a compendium and guide to the body of knowledge that has been developing and evolving over the past four decades: the Guide to the Software Engineering Body of Knowledge (SWEBOK) project [1].

SWEBOK knowledge is subdivided into ten Knowledge Areas (KAs) – see Figure 1. To provide a topical access to the knowledge, each KA is further broken down into topics and sub-topics, and also identifies the related seminal reference material and a matrix linking the reference material to the topics listed. In the OO paradigm, the 10 KAs could be considered as subclasses of the SWEBOK super class. Every software engineering concept would be a subclass of one or more of the KAs. This means that a concept should be a subclass of the super class and have relations to different KAs. But super classes and subclasses, as well as the definitions of the concepts, represent only a first step. A SWEBOK user is not only interested in the definitions of the concepts, but also in much more detailed information about the topics that are important to him. In SWEBOK, this detailed level of information is not in the Guide itself, but in its reference material.

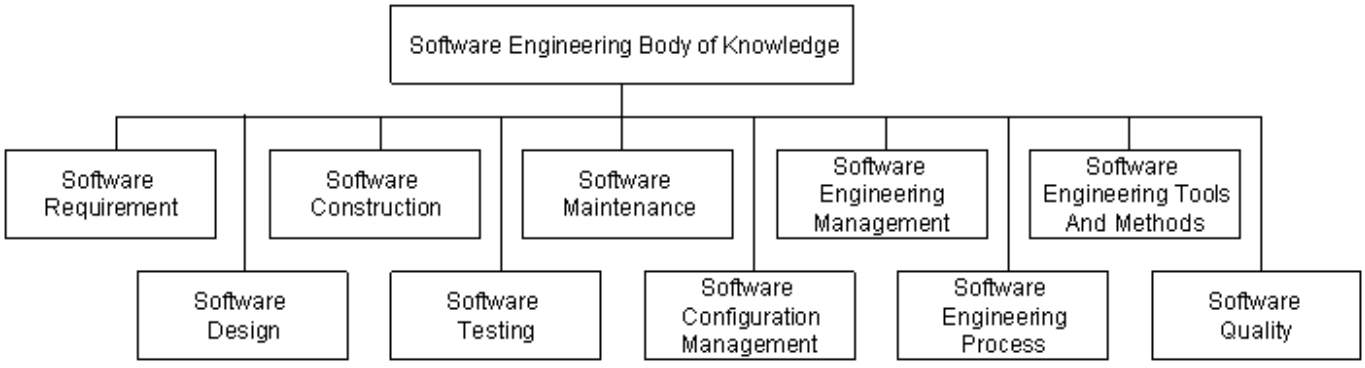

Figure 1: Knowledge Areas of the Software Engineering Body of Knowledge

The authors and hundreds of reviewers from 42 countries have contributed to SWEBOK and, in parallel, the document was reviewed by national software engineering standardization committees and approved in 2003 for publication and an ISO technical report, ISO/IEC TR 19759.

Because many authors have contributed to the initial versions of the SWEBOK Guide, it is necessary to verify the coherency and clarity of the terminology used within each chapter and across all chapters. For instance, in the SWEBOK Guide (Trial Version 1.00), the term *quality* is used 340 times and the word *software quality* 104 times.

Terms such as 'quality', 'measurements' and 'process' are used extensively in the SWEBOK Guide, but each of these terms might refer to many concepts used in different contexts and at different conceptual levels. This makes it challenging for beginner users of the Guide to recognize whether or not different subconcepts are being discussed when they are not identified as such by the use of distinct terms or expressions. It is therefore necessary to verify the precise interpretation of each of these terms throughout the text and to ensure that they are adequately identified, in order to improve the understandability of the SWEBOK Guide at a detailed level.

Detailed analysis of the SWEBOK text reveals that terms and expressions (concepts and related subconcepts) are often used in chapters with both similar and dissimilar meanings.

In the inventory and analysis of the SWEBOK Guide, it was observed that sometimes expressions used in a particular sense were being replaced by generic expressions, leaving the reader to figure out from the context that the expression was being used in the particular sense. One example is the use of the subconcepts *quality attributes* and *software quality attributes*; both appear in SWEBOK. 'Quality attributes' is used 15 times and 'software quality attributes' twice.

For users (humans or machines), different interpretations in distinct contexts sometimes make the meanings of terms confusing and ambiguous, while a coherent terminology adds clarity and facilitates understanding. "People can't share knowledge if they don't speak a common language" [17]. Explicit specifications of domain conceptualizations, called *ontologies*, are essential for the development and use of intelligent systems as well as for the interoperation of heterogeneous systems.

2 Ontology as a part of the Semantic Web

In recent years, the development of ontologies has moved from the realm of Artificial Intelligence laboratories to the desktops of domain experts and finally to the Web, taking advantage of the possibilities of this new communication tool. Many disciplines now develop ontologies or standardized "vocabularies" which domain experts can use to share and annotate information in their fields. Medicine, for example, has produced large, standardized and structured vocabularies, and there is also evidence of emerging "ontologies" in the field of software engineering [16]. There is also a need for ontologies in computer science [7].

What is an ontology then? For Gruber, "an ontology is a specification of a conceptualization" [19]. An ontology is also a specification of some topic. It is a formal and declarative representation which includes the vocabulary required for referring to the concepts in that subject area and the logical statements that describe what the concepts are, how they are related, and can be related, to one another. Ontologies therefore provide a vocabulary for representing and communicating knowledge about some topic and a set of relationships which hold among the concepts in that vocabulary.

Some of the reported benefits of ontologies are that they [10]:

- Enable a new and effective way to reuse knowledge;
- Help us use, and understand, some area of knowledge better;
- Help us analyze the structure of knowledge;
- Help us reach a consensus on our understanding of some area of knowledge;
- Help us share a common understanding of the structure of information, among people or software agents;
- Enable a machine to use the knowledge in some application.

In addition, a SWEBOK ontology could help to separate software engineering knowledge from other operational knowledge. In this way, general statements could be consciously delimited. For example, every product has quality attributes; however, the ontology shows that quality attributes in the context of software (software quality attributes) are different from quality attributes for other products.

An internationally recognized software engineering ontology, when and if one becomes available, would make it easier to carry out changes to the knowledge and to teach this new knowledge to software engineers. In addition, explicit specifications of software engineering knowledge are useful for new researchers who will learn the meaning of concepts in the domain. A software engineering ontology can play an important role for people who want to learn more about software engineering.

Knowledge based on an ontology is also machine-readable and so useful for an e-learning structure. For researchers, an ontology will also include machine-interpretable definitions of basic concepts in the domain

and the relations among them. The ontology approach seems a promising path to follow to tackle terminology issues at lower levels of detail, since an ontology provides a standard terminology for a specific context.

Ontology development is necessarily an iterative process. Concepts in the software engineering ontology should be close to objects of interest (physical or logical) and to the relationships between them.

Ontologies are a part of the Semantic Web. The Semantic Web is the representation of data on the World Wide Web, and it will by developed under the leadership of the W3C consortium. "The Semantic Web is also a extension of the current Web, in which information is given well-defined meaning, enabling computers and people to work more cooperatively."[18] The Semantic Web should be able to support automated services based on formal descriptions of semantics, and is seen as a key factor in finding a way out of the growing problems of traversing the expanding Web space [7].

Two important and fundamental technologies for developing the Semantic Web are already in place: eXtensible Markup Language (XML) and languages based on the Resource Description Framework (RDF).
Figure 2 gives an overview of the development and structure of languages for the Semantic Web.

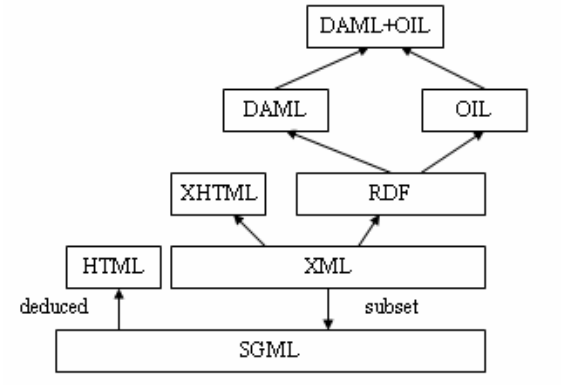

Figure 2: Developing languages for the Semantic Web

DAML (DARPA Agent Markup Language) and OIL (Ontology Inference Layer) have been combined into an important ontology language [13] [16]. The following table gives a first overview of the criteria of the various ontology languages.

Table 1: Relations between ontology languages

characteristics	XML DTD	XML Schema	RDF (S)2002	OIL	DAML+OIL
ordered list			X		X
cardinality restrictions	X	X		X	X
class expressions				X	X
data types		X	X	X	X
class definition				X	X
listing	X	X			X
equivalence				X	X
extensibility			X	X	X
formal semantic			X	X	X
inheritance			X	X	X
inference				X	X
local restrictions				X	X
quantitative restrictions					X

Based on XML and RDF, the DAML+OIL language has been specially developed to create ontologies for the Semantic Web [16] [9].

3 Design for a SWEBOK ontology

The first challenge in developing an ontology for SWEBOK is to define what the ontology should contain and the purpose for which it should be used.

Of course, the SWEBOK Ontology should include all the important concepts in software engineering. These concepts should be supported by widely accepted definitions, facilitating a common understanding by all users in this knowledge domain. Concurrently, an ontology should provide a necessary delimitation with respect to other domains of knowledge. In practice, developing an ontology also includes defining classes within the ontology and arranging the classes in a taxonomic (subclass–super class) hierarchy. The structure of knowledge provided in the SWEBOK Guide provides a starting point for the design of a software engineering ontology.

SWEBOK is the super class of the ontology. The ten KAs are the subclasses of the super class and represent specialized views of parts of the software engineering knowledge. Each KA is represented by a structured set of concepts and corresponding definitions. All concepts are subclasses of the super class and they can also be subclasses of one or more KAs.

A second important aspect in the design of an ontology is that much of the software engineering knowledge in the SWEBOK Guide is represented by links to internal and external references. To model such links, we need more than only the unidirectional HTML links. In the current book format of the SWEBOK Guide, it is not possible for the user to access a single (and unique) reference for a concept. In future versions of the Guide, the user should ideally be provided with a quick way to find either a reference or a concept by means of the SWEBOK ontology, as well as additional information related to his search.

Bidirectional and multidirectional links allow for information-sharing in both directions, which means that every concept can by referenced in one or more ways.

Also, the path from the reference to the concept is available, as illustrated in Figure 2, for the testing concept and some of the relevant references.

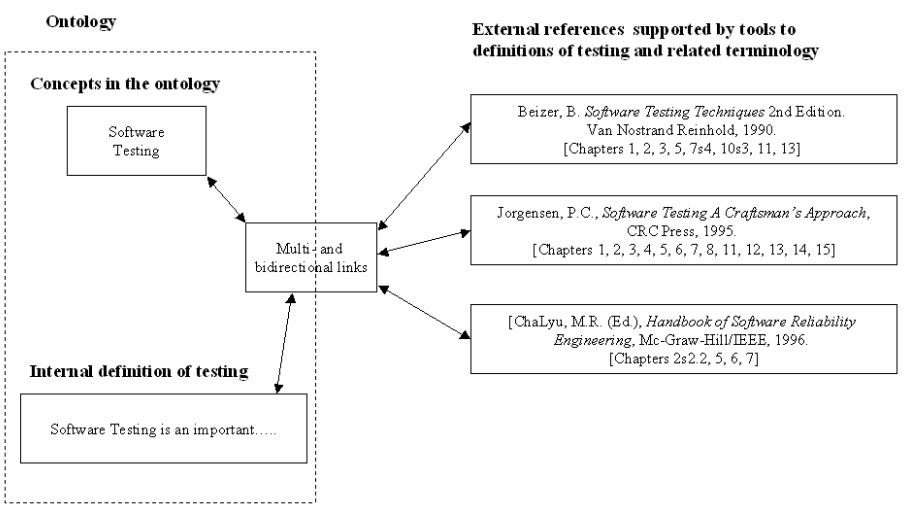

Figure 3: Example of a link structure with internal and tool-supported external sources

An ontology with bidirectional and multidirectional links would make it possible for every user (as well as applications and agents) to have very fast access to the corresponding details of high-level knowledge.

By contrast to other ontologies, such as in the medical field, the structure of a software engineering ontology will be relatively flat. Under the root element, there will be different KAs and different concepts. In other domains, an electronic marketplace, for example, the structure would be much deeper.

The SWEBOK structure will contain many different links to help the user find knowledge quickly, and most of these links will point to external references (see Figure 3).

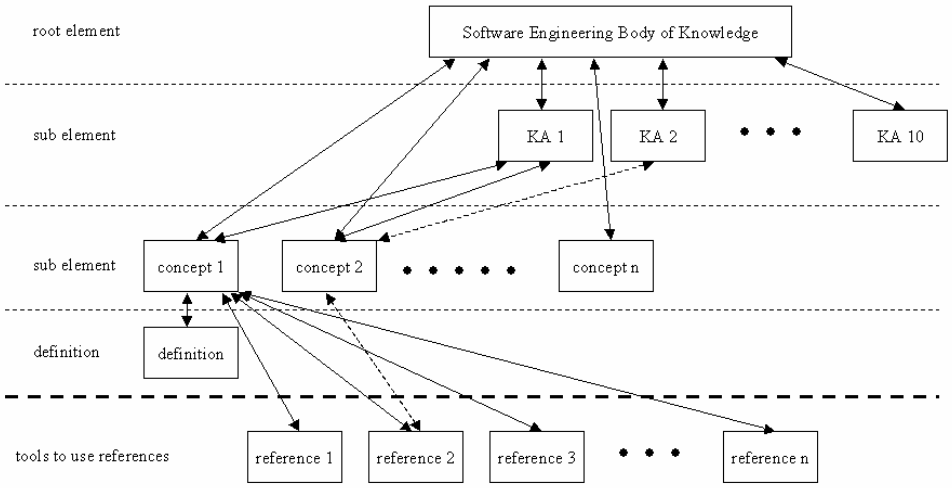

Figure 4: Design of a software engineering ontology with different levels of knowledge

Only a few links are illustrated in Figure 4, out of a much larger number of available references. All concepts are subclasses of SWEBOK and also subclasses of one or more KAs. Every concept has a definition and one or more internal or external references. To find the knowledge he is looking for, the user can use various views. For example, he can navigate from KAs through to concepts with their definitions and references. He can also search from the perspective of all the concepts in SWEBOK. In the future, if bidirectional links are available, users will also be able to navigate from a reference to a KA or to other references.

An initial example of a data structure for a software engineering ontology is presented in Figure 5. Under the root element (SWEBOK) are the KAs with their names and a list of all the concepts used in each of the corresponding KAs. Also under the root element are all the concepts used in all the KAs. These concepts have the following attributes: 'name', 'is_defined_as', 'is_used' and 'uses'. The definition of the concept gives information

about the source of the definition. The expression 'is_used' represents a list of all the KAs that use the concept, and 'uses' represents a list of references outside SWEBOK which are supported by tools.

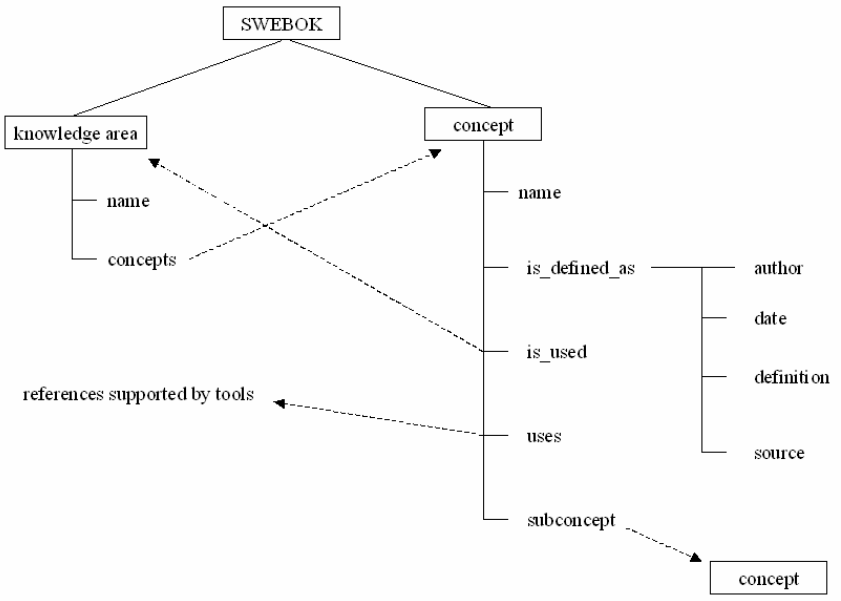

Figure 5: Structure for a software engineering ontology

Concepts can have associated subconcepts. These subconcepts are also concepts, and have the same structure. For example: software quality is a subconcept of quality, and design quality is also a subconcept of quality.

In section 2, it was noted that different XML- and RDF-based languages are available to develop an ontology. Currently, DAML+Oil would appear to be the best technology choice; however, since technology in this area is still evolving rapidly, an evolutive strategy must be designed to ensure stability of the ease of technological evolution for the development platform to be selected.

4 SWEBOK ontology and E-Learning

Web technologies, dynamic Websites and e-learning technology make it possible to develop e-learning systems on software engineering. With the use of ontologies, it is possible to give data and knowledge a structure which allows machines and people to use, and share, this knowledge.

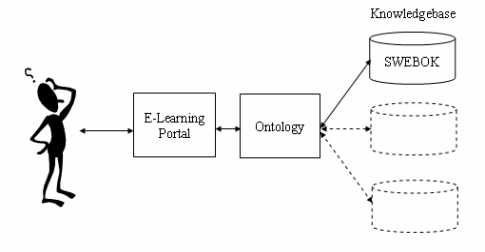

Figure 6: Ontology-based structure for an e-learning system

The system types of e-learning are [3]:

- open telelearning
- advice telelearning
- teleteaching

In future, documents could be logically linked with the help of Semantic Web. Thus, e-learning systems can improve knowledge engineering from data mining through text mining to Web mining. The Semantic Web provides descriptions of Web resources and e-learning portals present them to the user in a contextually clear way (Figure 6).

The first examples of e-learning in the area of software measurement are supported by the SML@b of the University of Magdeburg [15].

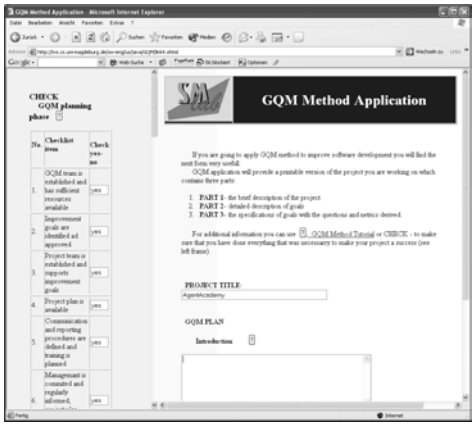

Figure 7: Example of measurement e-learning in the SML@b

The SWEBOK ontology will annotate unstructured information and knowledge with semantic information to integrate information and to generate user-specific views which make access to software engineering knowledge easier.
As the next-generation Web (does this make sense?), the Semantic Web will enable automatic knowledge processing over the Internet, using intelligent services such as search agents, information brokers, and information filters.

5 Conclusions and Future Work

Under the leadership of the IEEE Computer Society, a compendium and Guide to the Software Engineering Body of Knowledge has been developed, and approved in 2003 for publication in 2004 as ISO/IEC TR 19759. A very detailed inventory of terms and expressions used in the SWEBOK Guide points, however, to a need to improve the consistency of the terminology. The current Trial version of the Guide represents a large number of viewpoints in a domain where there had not yet been a consensus on a single set of software engineering terms.
The design of a software engineering ontology could help improve the consistency of the terminology in the SWEBOK Guide. An ontology is a flexible and useful way to define terms, their concepts and subconcepts, and show how they are related to one another in the context of domain knowledge. An ontology is also a conceptualization, and presents domain knowledge and its structure in a general manner.
In this paper, we presented a candidate approach for the design of an ontology for SWEBOK. Various languages are available to create an ontology, and all are based on XML and RDF. To decide which is the best one to use will be challenging, since ontologies and their languages are only in the beginning stages of development, as are their technologies. A critical step in creating a SWEBOK ontology will be to define the strategy required for selecting the technology that will best support it.
The Semantic Web and ontologies will bring structure to the meaningful content of Web pages and allow building of e-learning systems on software engineering knowledge based on SWEBOK.

6 References

[1] A. Abran, J. Moore, P. Bourque, R.L. Dupuis, L. Tripp, *Guide to the Software Engineering Body of Knowledge – SWEBOK*, Trial Version 1.0, IEEE-Computer Society Press, May 2003, URL: http://www.swebok.org

[2] A. Abran, R. R. Dumke, Investigations in Software Measurement-Proceedings of the 13th International Workshop on Software Measurement, Shaker Verlag Aachen, 2003.

[3] R. R. Dumke, M. Lother, C. Wille, F. Zbrog, Web Engineering, Pearson Studium München, 2003.

[4] ISO, ISO/IEC 9126-1:2001 *Software engineering – Product quality – Part 1: Quality model*, International Organization for Standardization/International Electrotechnical Commission., Geneva, 2001.

[5] ISO, ISO/IEC 15939:2002 *Software Engineering: Software Measurement Process*, International Organization for Standardization/International Electrotechnical Commission., Geneva, 2002.

[6] J. Hasebrook, L. Erasmus, G. Doeben-Henisch, *Knowledge Robots for Knowledge Workers: Self-Learning Agents Connecting Information and Skills*, In: Jain/Chen/Ichalkaranje: Intelligent Agents and Their Applications, Physica-Verlag Heidelberg New York, Heidelberg, 2002, pp.59-81.

[7] A. Maedche, *Ontology Learning for the Semantic Web*. Boston: Kluwer Academic, 2002, 272 p.

[8] A. Maedche, S. STAAB, *Ontology learning for the Semantic Web*, Intelligent Systems, IEEE [see also IEEE Expert], vol. 16 (2), pp. 72-79, 2001.

[9] M. Missikoff, R. Navigli, S. P. Velardi, *Integrated Approach to Web Ontology Learning and Engineering*, IEEE Computer, November 2002.

[10] N. F. Noy, D. L. McGuinness, *A Guide to Creating Your First Ontology*, Stanford University, Mai 2003, URL: http://protege.stanford.edu/ publications/ontology_development/ontology101.pdf

[11] N. Guarino, P. Giaretta, *Ontologies and Knowledge Bases: Towards a Terminological Clarification*, In N.J.I. Mars, editor, *Proc*. of the 2nd Intern. Conf on Building and Sharing Very Large Knowledge Bases. IOS Press, Enschede, The Netherlands, 1995.

[12] N. Zhong, *Ontologies in Web intelligence*, In: Jain/Chen/Ichalkaranje: Intelligent Agents and Their Applications, Physica-Verlag Heidelberg New York, Heidelberg, 2002, pp.83-97.

[13] *OIL Homepage*, July 2003, URL: http://www.ontoknowledge.org/oil/

[14] O. Andruschak, Ein Softwaremess- und –bewer-tungsansatz für das Semantic Web, Diploma Thesis, Otto-von-Guericke University Magdeburg, 2003

[15] SML@b Website, October 2003 URL: http://ivs.cs. uni-magdeburg.de/sw-eng/us/index.shtml

[16] *The DARPA Agent Markup Language Homepage*, July 2003, URL: http://www.daml.org/

[17] T. H. Davenport, L. Prusak, *Working Knowledge: How organizations manage what they know*, Harvard Business School Press, 1997.

[18] T. Berners-Lee, J. Hendler O. Lassila, The Semantic Web, Scientific American, May 2001.

[19] T. R. Gruber. *A Translation Approach to Portable Ontology Specifications*, Technical Report KSL 92-71, Stanford University, April 1993.