Issues in the development of an ontology for an emerging engineering discipline

Olavo Mendes
DECOM/CCHLA/UFPB
Federal University at Paraiba – Brazil

PhD Student Cognitive Informatics
Université du Québec à Montréal - UQAM
olavomendes@hotmail.com

Alain Abran
École de Technologie Supérieure – ETS
Université du Québec
1100 Notre-Dame Ouest,
H3C 1K3 Montréal Québec , Canada,
aabran@ele.etsmtl.ca

Abstract

The Guide to the software engineering body of knowledge (SWEBOK - ISO TR 19759) provides a consensually validated characterization of the bounds of the software engineering discipline as well as a topical access to the Body of Knowledge supporting that discipline. This Body of Knowledge is currently organized as a taxonomy subdivided into ten Knowledge Areas designed to discriminate among the various important concepts, but only at the top level. Of course, the software engineering knowledge is much richer that this high level taxonomy and currently resides in the textual descriptions of each knowledge area. Such textual descriptions widely vary in style and content. The ontology approach is therefore used to analyze the richness of this body of knowledge and to improve its structuring. This paper presents the proto-ontology developed in the first phase of the construction of a domain ontology for this new engineering discipline. Overall, some six thousands (6000) software engineering concepts and about 400 relationship types between concepts have been identified. Some of the major results obtained to this point are detailed and discussed.

Keywords: *SWEBOK, Software Engineering Body of Knowledge, Ontology, Domain ontology, Ontology development, Ontology construction, SWEBOK Ontology, Software Engineering ontology*

1 Introduction

Ontologies have been known in philosophy since Aristotle and Porphyry [N1 3b]. In the computer domain the emergence of ontologies is much more recent: in the early 90s, the DARPA project «Knowledge Sharing Initiative» [2] that involved many research centers across the USA, had as a goal to reduce the time and effort (and, therefore, the costs) required to develop knowledge data bases, through sharing and reuse [3]. Since we cannot share and reuse knowledge if we do not speak the same language and have somehow a consensus concerning the meanings of the concepts used to communicate, the researchers introduced the ontologies to describe the semantics/meanings and to make explicit the domain assumptions associated to the knowledge to be shared and reused **[4] [5].**

So, in the computer domain, an ontology represents a consensual and shared description of the pertinent objects and of their interrelationships considered as existing in a certain domain of knowledge **[6]**, described in a formal and explicit way as well as the terms we use to refer to them and their agreed meanings and properties [6] [8]. This description takes the form of: concepts, properties and attributes, constraints on properties and attributes and, often but not always, individuals (instances of the concepts) **[7].**

Ontologies make thus possible communication among people/organizations, systems/software agents, and people and systems by agreeing and sharing a common understanding about a conceptualisation, recognizing the existence of a set of objects and their relationships, as well as the terms used to refer to them and their agreed meanings (ontological commitment) [7] [5].

Ontologies could play an important role in Software Engineering as they do in other disciplines where they: 1) provide a source of precisely defined terms that can be communicated across people, organisations and applications (information systems or intelligent agents); 2) offer a consensual shared understanding concerning the domain of discourse; 3) render explicit all hidden assumptions concerning the objects pertaining to a certain domain of knowledge [6] [8] [17].

Despite some initial effort to develop partial (sub domain) ontologies (software maintenance [14] [15], software measurement [16], software quality [9] [10], OO Design [17]) as a field of knowledge, Software Engineering still does not have a comprehensive detailed ontology which describes the concepts that domain experts agree upon, as well as their terms, definitions and meanings. Such an ontology would also need to look at the more pertinent relationships where concepts participate in the creation of the semantic network in which they are inserted [11].

The development of a “software engineering domain ontology” would allow us to: 1) share and reuse knowledge accumulated until now in the Software Engineering field; 2) open new avenues to automatic *interpretation* of this knowledge using information systems or *intelligent* software agents.

The rest of this text is structured as follows. Section 2 presents the SWEBOK guide that provides a consensually validated characterization of the bounds of the software engineering discipline as well as a topical access to the Body of Knowledge supporting that discipline. Then, Section 3 presents the construction methodology used to produce the SWEBOK ontology. Section 4 presents next some preliminary results for the SWEBOK proto-ontology currently under development and, Section 5, a summary and some directions for further work.

2. The SWEBOK Guide

The SWEBOK project - Software Engineering Body of Knowledge [11] [12], is the result of a collaborative effort between the IEEE Computer Society and Université du Québec (École de Technologie Supérieure and UQAM). Over the years, close to 500 reviewers from very diverse domains including the industrial and academic fields, government agencies, professional societies, international standards organisations, as well as research centers, have been involved in the project, which has thus earned an international credibility in the software engineering field.

The resulting SWEBOK Guide is the result of great effort of declarative and procedural knowledge mining, acquisition and structuring that was, until then, scattered in a myriad of very diverse documents (scientific papers, conference proceedings, books, chapters, technical reports, technical standards), and of empirical knowledge from field experts and researchers.

The SWEBOK project team established the project with five objectives [12]: 1) To characterize the content of the software engineering discipline; 2) To provide topical access to the software engineering body of knowledge; 3) To promote a consistent view of software engineering worldwide; 4) To clarify the place – and set the boundaries – of software engineering with respect to other disciplines such as computer science, project management, computer engineering, and mathematics; 5) To provide a foundation for curriculum development and individual certification material.

The SWEBOK project allowed, through multiple review cycles, to build a consensus on: 1) the knowledge areas consensually agreed to integrate the software engineering field; 2) the knowledge content associated to each knowledge area, as well as the related major references; 3) the scientific disciplines participating in each knowledge area.

The resulting product of the SWEBOK project it is not the body of knowledge itself, but rather a guide to it, permitting to gain consensus on the core subset of knowledge characterizing the software engineering discipline [12] [13]. As a result, ten knowledge areas have been identified as integrating the Software engineering field: KA.01 Software requirements, KA.02 Software design, KA.03 Software construction, KA.04 Software testing, KA.05 Software maintenance, KA.06 Software configuration management, KA.07 Software engineering management, KA.08 Software engineering process, KA.09 Software engineering tools and methods, KA.10 Software quality.

This Body of Knowledge is currently organized as a taxonomy subdivided into ten Knowledge Areas designed to discriminate among the various important concepts, but only at the top level. Of course, the software engineering knowledge is much richer than this high level taxonomy and currently resides in the textual descriptions of each knowledge area. Such textual descriptions widely vary in style and content. The ontology approach is therefore used in the research presented here to analyze the richness of this body of knowledge, to improve its structuring and to develop a consensus on its detailed terminology.

3. Ontology Development Methodology

The process adopted by the SWEBOK project has permitted a progressive building of consensus among the experts participating to the Delphi panels concerning the knowledge and structure of the Software Engineering discipline: the SWEBOK Guide represents therefore an important and privileged information source for the construction of a Software Engineering domain ontology.

The ontology building process integrates a number of major activities: 1) Specification; 2) Conceptualization; 3) Ontologization; 4) Integration (with other sub-ontologies which might be available); 5) Operationalization; 6) Evaluation [18].

Our process to develop the software engineering domain ontology requires three phases: 1) Proto-ontology construction; 2) Internal validation cycle; 3) External validation and possibly extension - V&E cycle.

Proto-ontology construction: We started the ontology construction process with the development of a **proto-ontology** using the information contained in the SWEBOK Guide. The descriptions contained in the SWEBOK Guide were analysed and the concepts, relationships between concepts, terms and definitions were extracted, one SWEBOK knowledge area at a time. Some definitions for the concepts extracted were complemented using the 610.12-1190 IEEE Standard Glossary of the Software Engineering Terminology that contains 1500+ entries.

This phase corresponds to the conceptualization and ontologization phases traditionally existing in ontology development methodologies.

This concept extraction by detailed inspection of the SWEBOK Guide content was complemented by the use of automatic terms extraction tools having as input the SWEBOK corpus of text in natural language. The outputs of the term extraction tools were used to cross-validate and complete the list of concepts and relationships identified through the analysis of the documents.

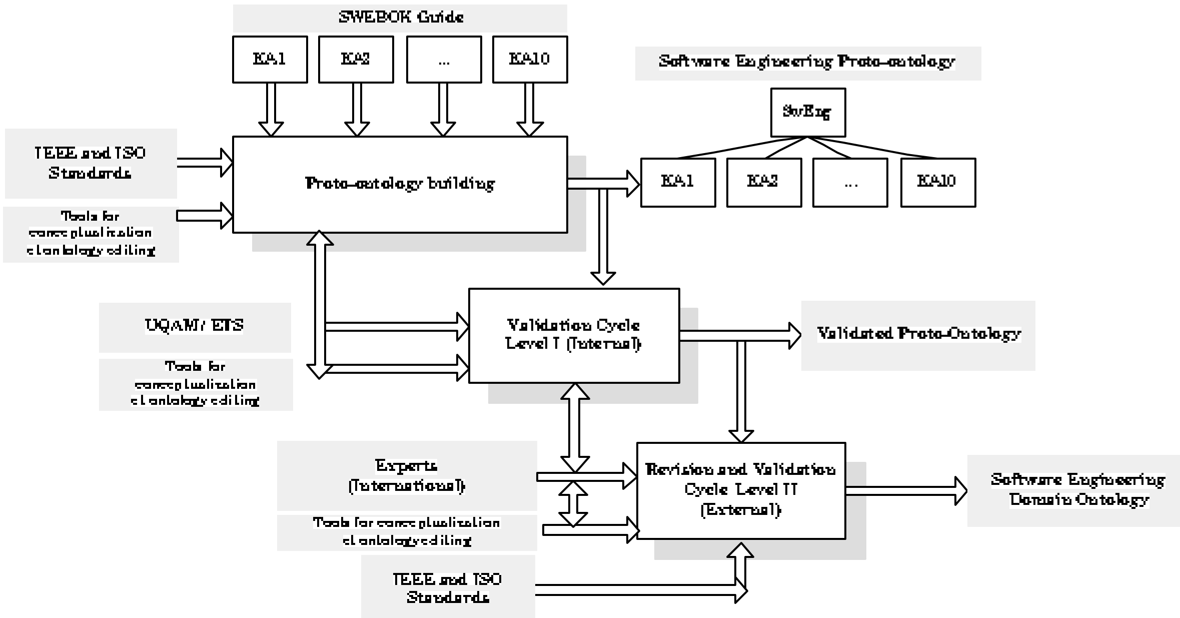

Figure 1 – The SWEBOK ontology project phases ontology

Internal validation cycle: We are presently starting the internal validation cycle at various instances levels (internal: ETS – UQAM – SPIN, etc.), aiming to build a progressively larger consensus about the elements in this software engineering proto-ontology.

External validation cycle: Finally, a series of external validation and possibly proto-ontology extension - V&E - cycles will be performed (beginning in July, 2005), aided by international software engineering domain experts, to build progressively a consensus about the concepts, attributes and relationships between class/concepts that should be present in the final ontology.

This proto-ontology represents the starting point for the development of a Software Engineering domain ontology: it is based on an already consensual domain knowledge (e.g. the SWEBOK Guide) and will serve as an initial focus to the domain experts starting up the ontology construction process. The V&E phase will be performed on the conceptual level of the SWEBOK proto-ontology. Once the V&E completed, the SWEBOK ontology will be translated to the operational level using ontology editors and the OWL language.

4. The SWEBOK Proto-ontology

The proto-ontology development phase has identified in the SWEBOK Guide over 6,000 concepts, linked by normalized relationships, as well as 1,200 facts (examples/instances of concepts). Table 1 presents a breakdown by knowledge areas: the column 'Relationships' shows the total number of relationships linking the concepts in the ontology. These relationships have been normalized in order to limit and standardize the great variety of terms having the same meaning that the natural language allows. The column 'Index' represents the concepts related to the structure of the SWEBOK guide (KA, section, sub-section, etc.) and will permit to trace back where a concept is used in the SWEBOK guide. In Table 1, Software Engineering Management, Software Testing and Software Maintenance knowledge areas have the greatest number of concepts; on the other hand, Software Engineering Tools and Methods, Software Requirements and Software Design knowledge areas have the smallest number of concepts.

Figure 2 presents the concepts in the main level of the SWEBOK ontology (in its conceptual format). A set of concepts mainly related to the *structural* organisation of the SWEBOK guide are depicted (shown in grey). Other concepts in the example relate to the *contextual* aspects: the guide version, the editors, the reviewer team, the industrial advisory board, and the experts who participated in the SWEBOK review cycles to build the consensus about the knowledge areas, KAs knowledge content and related domain areas.

Each knowledge area is then progressively exploded to reveal the concepts (and relationships linking these concepts) embedded in their sections and subsections. The grey boxes represent concepts associated to the SWEBOK structure and, the oval boxes, an index that allows to subsequently tracing back a concept pertaining to a section of the SWEBOK Guide.

Table 1 – Overview of the number of elements currently in the SWEBOK proto-ontology

	Relationships	Index	Concepts	Facts
SWEBOK (Main structure)	6	0	**39**	57
KA 01 Introduction	25	0	**673**	14
KA 02 Software Requirements	41	44	**205**	72
KA 03 Software Design	46	45	**267**	200
KA 04 Software Construction	23	20	**200**	62
KA 05 Software Testing	97	101	**1048**	165
KA 06 Software Maintenance	47	45	**725**	141
KA 07 Software Configuration Management	51	56	**960**	102
KA 08 Software Engineering Management	40	38	**1059**	109
KA 09 Software Engineering Process	45	37	**562**	134
KA 10 Software Engineering Tools and Methods	19	51	**198**	58
KA 11 Software Quality	37	34	**412**	82
CH 12 Related Disciplines of Software Engineering	12	0	**164**	32
TOTAL		**471**	**6512**	**1228**

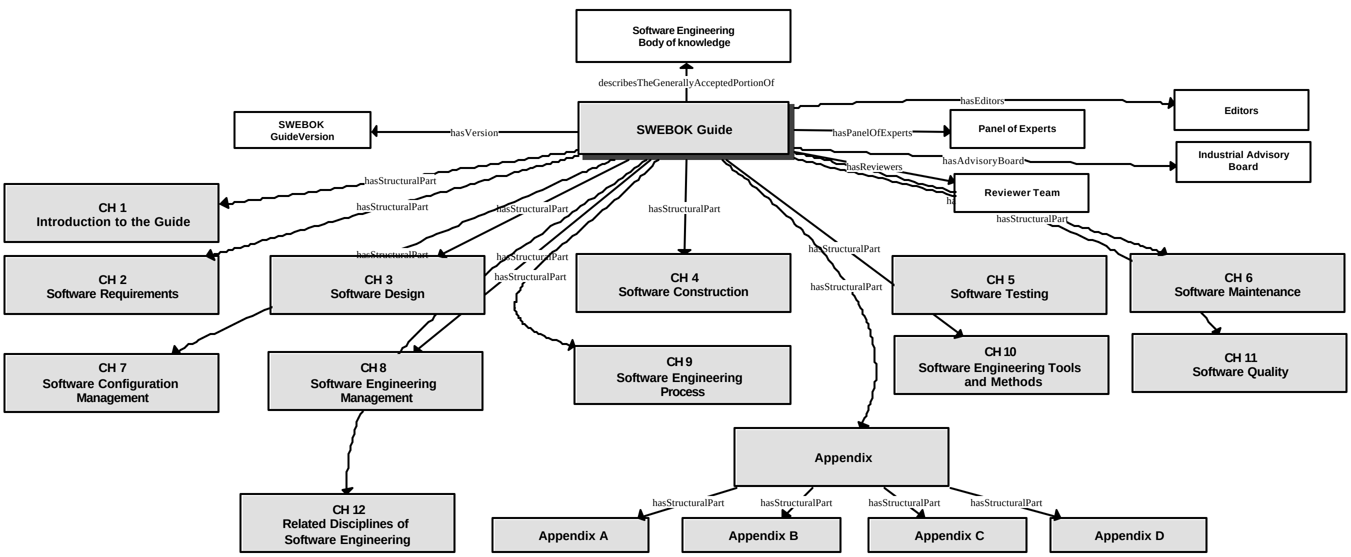

Figure 2 – Overview of the SWEBOK Proto-ontology (Main Level)

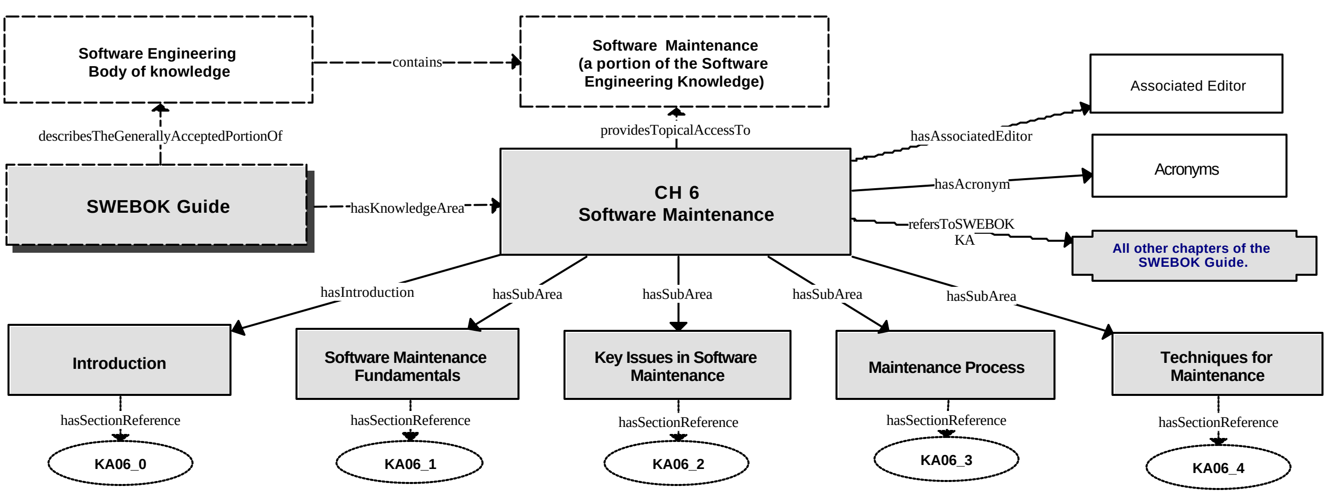

Figure 3 – The Software Maintenance Knowledge Area (Main Level)

Internal references between Knowledge Areas are represented by an instance of the structural concept KA (in the example depicted in Figure 2, the Software Maintenance KA is related to all other SWEBOK Knowledge Areas).

A more detailed view of the proto-ontology is presented in Figure 3 that shows the main level of the Software Maintenance knowledge area. The descriptions associated to this KA are presented first in an introduction presenting the sub area, followed by four sections where the main concepts are presented. Indexes representing the sections references are also shown as ovals. Some contextual information concerning the knowledge area associated editor and the acronyms used are also shown. This background information is provided only for the specific purpose to provide to the domain experts traceability to the SWEBOK structure in the proto-ontology to be validated and extended. Therefore, the concepts related to the structure of the SWEBOK Guide will not appear in the final Software Engineering ontology.

In Figure 4, three instances of concepts are also shown (two bibliographic references and Spiral, as an instance of the software life cycle model). Two generalization-specialisation hierarchies are also shown (Actor and Maintenance actions), represented by the «S» links.

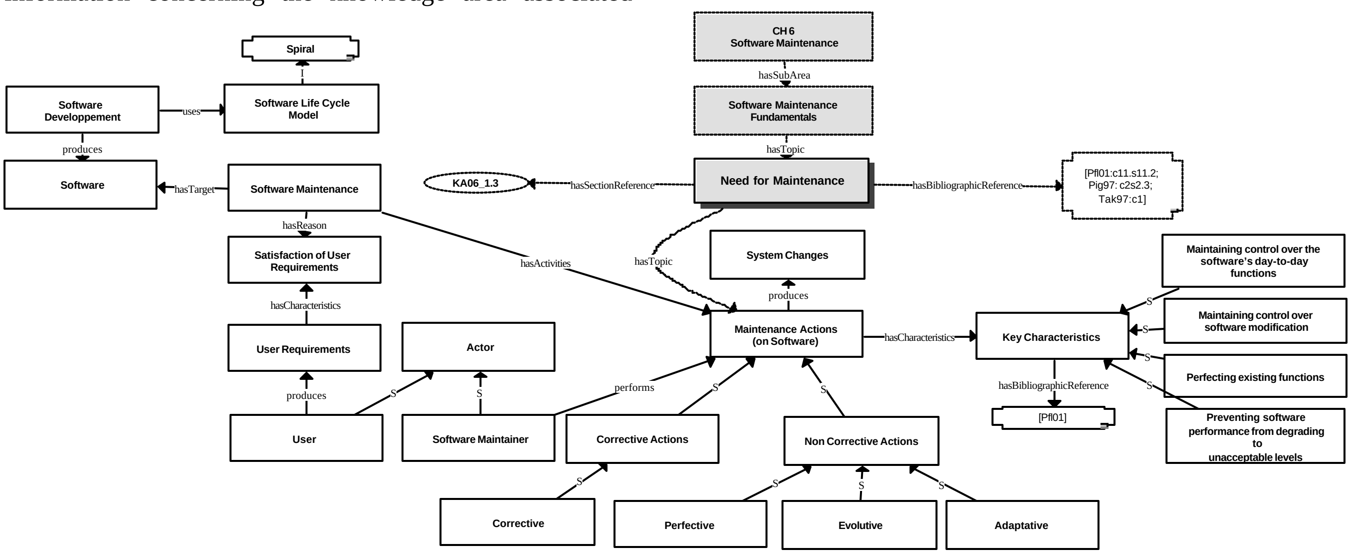

Figure 4 – The Software Maintenance Knowledge Area (Detailed Level)

5. Summary and next steps

Our project goal is to build and validate an ontology for the Software Engineering discipline. To reach this goal, an initial domain ontology (e.g. a proto-ontology) was developed for the software engineering area, taking as starting point the consensual knowledge already acquired, structured, validated and made available by the SWEBOK project (SWEBOK Guide - Iron Man version, 18.05.2004). Technical standards (IEEE and ISO) will also be used to complete the SWEBOK ontology, providing for definitions of the currently accepted terminology as well as alternate accepted terms. The resulting domain ontology will integrate a set of artefacts corresponding to the conceptual, ontological and operational levels of the software engineering validated ontology.

This paper has presented samples of the proto-ontology developed in the development phase for a comprehensive ontology for the software engineering discipline.

The major contributions expected fromthis study are:

1) Identification of the main inputs, outputs and activities to be performed in order to develop the target ontology;

2) Identification of the main software engineering concepts, terms, definitions, relations between classes/concepts (IsA, Part-Whole, and other specifics relationships) and axioms describing the concepts;

3) Validation (and possibly extension) of the software engineering ontology;

4) Progressive building of a consensus concerning the concepts in the ontology with the support of international software engineering domain experts.

Besides the benefits already mentioned in section 1, the use of this “software engineering ontology” may also contribute later to the development of additional content validation by carrying out *automatic* cross-correlation validation across the ten areas of knowledge integrated in the SWEBOK Guide. This next step would ensure that all concepts and definitions are used in a consistent fashion throughout all ten SWEBOK knowledge areas as well as to harmonize the level of description of the SWEBOK Guide content.

An automatic validation would also be useful in the ISO/IEC JTC1/SC-7 SWG5 development towards the harmonisation of all vocabulary used by the various working groups involved in software engineering technical standards.

Further work in this project includes: 1) ontology V&E and 2) cognition-communication analysis. In the former, we are starting the validation and extension (V&E) cycle with panels of domain experts. This phase will produce a series of sub-ontologies (one for each validated knowledge area) that, once integrated, will form the SWEBOK ontology. These sub-ontologies will be subsequently operationalized using the OWL language.

In the second one – cognitive-communication analysis – we will observe and analyse the interactions that take place among the group of domain experts when they are working collaboratively to validate and extend the SWEBOK proto-ontology. The identification and modelling of the communication interactions and of the cognitive activities that emerge within the distributed cognitive system formed by the experts working in the V&E of the SWEBOK ontology, will contribute to identify major key issues and challenges in the ontology V&E process, as well as to formulate some recommendations aiming at improving the global efficiency of the ontology construction process.

References

[1] Porphyry "Isagoge", Vrin, 1998, ISBN: 2711613445

[2] Patil et al. 1992, "The DARPA Knowledge Sharing Effort: Progress Report", Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference KR'92, San Mateo, California, p: 777-788.

[2] Neeches R., F. R. E., Finin T., Gruber T. R., Senator T., and Swartout W. R., 1991. "Enabling technology for knowledge sharing", AI Magazine, 12, p: 35-56.

[4] Davenport, Thomas H., 1993, "Process Innovation: Reengineering Work Through Information Technology", Boston, MA, Harvard Business School Press.

[5] Guarino, N., Schneider , L., 2002, "Ontology-Driven Conceptual Modelling", Lecture Notes In Computer Science; Vol. 2503 Proceedings of the 21st International Conference on Conceptual Modeling, ISBN:3-540-44277

[6] Gruber, T.R.,. 1993, "Towards Principles for the Design of Ontologies Used for Knowledge Sharing", in Roberto Poli Nicola Guarino, editor, International Workshop on Formal Ontology, Padova, Italy, 1993, Technical report KSL-93-04, Knowledge Systems Laboratory, Stanford University.

[7] Rector, A., Schreiber, G., Noy, N. F., Knublauch H. and Musen, M., 2004, "Ontology Design Patterns and Problems ", Tutorial at the Third International Semantic Web Conference (ISWC 2004), November 7th, 2004.

[8] Gruninger, M., Lee, Jintae, 2002, "Ontology Design and Applications". Communications of the ACM, February 2002, 45 (2), p: 1-2.

[9] Wille, C., Abran, A., Desharnais, J.M.., Dumke, R., 2003, "The Quality concepts and sub concepts in SWEBOK: An ontology challenge", in International Workshop on Software Measurement (IWSM), Montreal, 2003 , p. 18.

[10] Wille, C., Dumke, R., Abran, A., Desharnais, JM. ,2004. E-Learning Infrastructure for Software Engineering Education: Steps in Ontology Modeling for SWEBOK, Software Measurement European Forum, Rome, Italy.

[11] Mendes, O., Abran, A. 2004. "Software Engineering Ontology: A Development Methodology", Position Paper, Metrics News 9:1,August, p: 68-76

[12] Bourque, P., Dupuis, R., Abran, A., 1999, "The Guide to the Software Engineering Body of Knowledge", IEEE Software, November/December.

[13] Abran, A., Moore, J., Bourque, P., Dupuis, R., Tripp, L., Guide to the Software Engineering Body of Knowledge – SWEBOK, Iron Man Version 1.0, IEEE-Computer Society Press, to be published 2005, URL: http://www.swebok.org

[14] Kitchenham, B., et al. 1999, "Towards a software maintenance ontology", Journal of Software Maintenance: Research and Practice, Vol. 11, p: 365-389.

[15] Ruiz, F., Vizcaíno, A., Piattini, M. y García, F., 2004, "An Ontology for the Management of Software Maintenance Projects", International Journal of Software Engineering and Knowledge Engineering, Vol. 14, No. 3, p: 323-349.

[16] Martin, M de A., Olsina, L., 2003, "Towards an Ontology for Software Metrics and Indicators as the Foundation for a Cataloging Web System ", First Latin American Web Congress (LA-WEB'03). 10 - 11, 2003. Santiago, Chile.

[17] Garzás J., Piattini M. 2005, "An Ontology for Microarchitectural Design Knowledge", IEEE Software Vol. 29, p: 28 -33.

[18] Mendes, O., 2004, "Méthodologies de construction d'ontologies", Congrès de l'ACFAS, Montreal May 12