SOFTWARE ENGINEERING ONTOLOGY: A DEVELOPMENT METHODOLOGY

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

PhD Student Cognitive Informatics
Quebec University at Montreal - UQAM
olavomendes@hotmail.com

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

Keywords: *SWEBOK, Software Engineering Body of Knowledge, Ontology, Ontology development, Ontology methodologies, SWEBOK Ontology*

1 Introduction

According to Gruber's definition an ontology [1] is "a formal specification of a conceptualization". A conceptualisation being a simplified, abstract way of perceiving a segment of the world (a piece of reality), for which we agree to recognize the existence of a set of objects and their interrelations, as well as the terms we use to refer to them and their agreed meanings and properties.

Thus, ontologies represent a consensual, shared description of the pertinent objects considered as existing in a certain domain of knowledge (the domain of discourse). They constitute a special kind of software artefact conveying a certain view of the world (conceptualization), specifically designed with the purpose of explicitly expressing the intended meaning of a set of agreed existing objects.

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) to render explicit all hidden assumptions concerning the objects pertaining to a certain domain of knowledge [2].

Despite some initial efforts to develop partial (sub domain) ontologies [3] [4] [5] [6], 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 ontology would also need to look at the more pertinent interrelations where concepts participate in the creation of the semantic network in which they are inserted.

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

2. The SWEBOK Project

The SWEBOK project - Software Engineering Body of Knowledge [7] [8], 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 standard organisation, as well as research centers, have been involved in the project, which has thus earned an international reputation 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 an myriad of very diverse documents (scientific papers, congress proceedings, books, chapters, technical reports, technical standards), and of background knowledge from field experts, consultants and researchers.

The SWEBOK project team established the project with five objectives [7]:

1) To characterize the contents 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 to build a consensus (using the Delphi technique) on: 1) the knowledge areas consensually agreed to integrate the software engineering field; 2) the knowledge content associated to each domain, as well as the related major references; 3) the scientific disciplines participating in each area of knowledge.

The resulting product of the SWEBOK project it is not the body of knowledge itself, but rather a

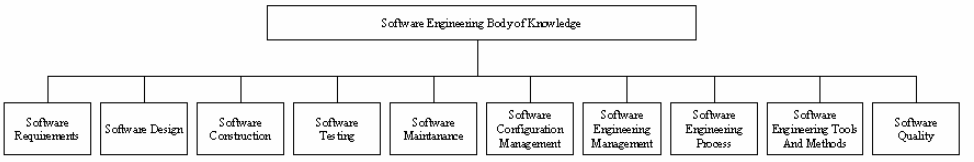

Figure 1: Knowledge Areas of the Software Engineering Body of Knowledge [7] [8]

3. Project Goal

Our ultimate project goal is to build and validate an ontology for the Software engineering field, using the knowledge already acquired, structured, validated and made available, by the SWEBOK project in the form of the SWEBOK Guide (last version Iron Man, 18.05.2004), as well as other scientific knowledge sources such as technical standards (ISO and IEEE).

Besides the benefits already mentioned in section 1, the use of the "software engineering ontology" which is a result of this project may also contribute to the development of additional content validation by *automatic* cross-correlation validation (besides that which is already done already done continuously by the SWEBOK review team) across the ten areas of knowledge integrated in the SWEBOK Guide. This would ensure that all concepts and definitions are used in a consistent fashion throughout all SWEBOK's areas of knowledge. An automatic validation would also be useful in the ISO/IEC JTC1/SC-7 SWG5 development toward's the harmonisation of all vocabulary used by the various working groups involved in software engineering technical standards.

4. The Problem

The ontology development process involves many activities that can present a high level of complexity, depending on the intended scope, size and level of detail of the ontology under construction [9] [10] [11].

As a consequence, the construction of an ontology cannot be conducted in an improvised or *ad hoc* fashion. The complexity of activities like conceptualisation, knowledge structuring/ontologisation, ontology evaluation, etc., require the use of management processes, in order to control cost, risks, schedules and to ensure that the artefacts produced are of the intended quality..

An important number of methodologies are presently described in the literature. The problem however is that 1) there is presently no consensus about the best practices to adopt concerning the construction of an ontology; 2) these ontology development methodologies make use of different construction methods, and frequently offer guidance to different portions of the ontology development cycle; 3) finally, until now, the ontology development process did not have any technical standard (official or *de facto*) to guide the development process, despite major efforts in this direction.

Deleted:

Thus a number of questions remain open:

- Which ontology development methodology provides the best guidance to attaint our established goal (the development of comprehensive software engineering ontology)?
- Which life-cycle model (cascade, incremental prototyping, evolutionary prototyping, etc.) is best suited to the planned ontology development?
- Which are the inputs, outputs and activities to be performed in order to develop the aimed ontology?
- Which are the key activities in the ontology development process?

This paper presents some preliminary results aimed at answering the above stated questions.

5. Methodology

In order to attain the stated goal, the following activities have been developed in this study:

- A detailed literature review of the ontology development methodologies;
- Preliminary classification of construction methodologies according to the mode of construction. Special emphasis was given to methodologies permitting ontology construction from scratch (Figure 2);
- Analysis of the ISO/IEC 12207-95 software life-cycle standard;
- Analysis of the surveyed methodologies from the ISO/IEC 12207-95 perspective;
- Preliminary identification of the differences and commonalities between the stated ontology development activities and the ISO/IEC standard;
- Proposal of a conceptual framework to compare and analyse the ontology development activities considered in the different methodologies;
- Additional literature reviews and refinement of the proposed conceptual framework;
- Identification and comparison of the ontology development activities proposed in the methodologies surveyed;
- Identification of the most and least frequently mentioned activities;
- Identification of key activities,

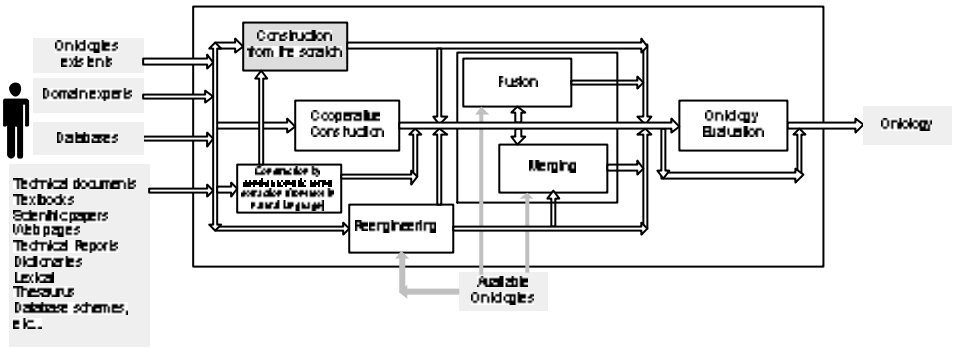

Figure 2: Framework for ontology development methodologies

6. Results

Some of the results produced by this study include:

A total of forty-eight ontology development methodologies have been identified in the literature review. Among these, fourteen corresponding to methodologies for the construction from scratch (the most recent in 2003) and seven for ontology evaluation. These figures indicate the dynamics in this area of research and the lack of an international standard or even of a *de facto* standard.

Leading ontology development methodologies authors [9] [11] [12] [13], agree that the process must be managed like any other software development project, in order to ensure that cost, schedule, risk and quality of the produced artefacts always remain under control. Nevertheless some project phases like Feasibility study, Project Planning, Tracking and Control are mostly absent from the methodologies surveyed. Configuration management, and quality assurance are also activities which are somehow absent. Despite being considered as primary life-cycle processes in the software development life-cycle, activities such as Deployment, Utilisation and Maintenance, are still very absent from the surveyed methodologies.

The activities mostly frequently mentioned in the literature are: Ontology specification, Conceptualisation, Ontologisation and Implementation. Authors consider these to be the three key ontology development activities.

It must be noted however, that there is a wide variation between methodologies concerning the terms used to name these activities, and the boundaries which define them. Sometimes, certain activities are absent or amalgamated with others.

Ontology evaluation and integration are examples of activities which share a large consensus between the surveyed methodologies that must be present in the process of ontology development.

Finally, among the fourteen ontology development methodologies surveyed, only two have a sufficient degree of coverage and detailed guidelines for users (domain experts and knowledge engineers/ontologists). We will adopt the guidance principles and activities prescribed by these methodologies in our project to develop a comprehensive Software Engineering ontology.

7. Towards a Software Engineering Ontology

We have chosen to implement the SWEBOK ontology using the OWL formalism due to its knowledge representation capabilities (by defining classes, individuals, properties, relationships in which these classes participates and axioms), and the possibility to reason about these classes and individuals. Other major web ontology languages are: SHOE (1996), XML (1996, 97), RDF (1997), OIL (late nineties), DAML – DARPA (2000), DAML+OIL (2001). OWL, the Web Ontology Language is the more recently ontology language (2001, Feb 2004).

At the root class of the ontology we find a concept, which corresponds to the SWEBOK Guide. Under this class (subclass of owl: Thing, a class that contains all classes), we find the main classes corresponding to the ten areas of knowledge that integrate the Guide, linked to the root class by the hasParts property. Each area of knowledge represents the agreed knowledge about the domain class, and can be successively exploded, revealing new classes with growing levels of detail. An example of the SWEBOK ontology (presented in the OWL formalism) is depicted at figure 3 (corresponding to the SWEBOK main level presented in figure 1).

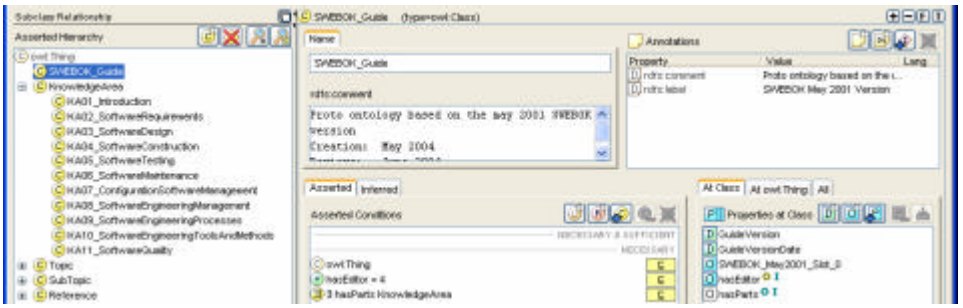

Figure 3: The SWEBOK Ontology main level

The classes (superclasses and subclasses) are organized in a structured hierarchy, using generalization/specialisation links to produce a taxonomy. Other types of links are also present (ex: contains, hasTopic, defines, and the inverse relations pertainsTo, isTopicOf, isDefinitionOf, etc.), capturing the existing semantics conveyed by multiples concept associations.

A zoom on the concept representing chapter 11 of the SWEBOK guide, reveals additional concepts, representing the knowledge associated with this topic. Figure 4 presents the four subtopics which exist under the Software Quality topic. The C* links represents the hasParts link with a *many* cardinality.

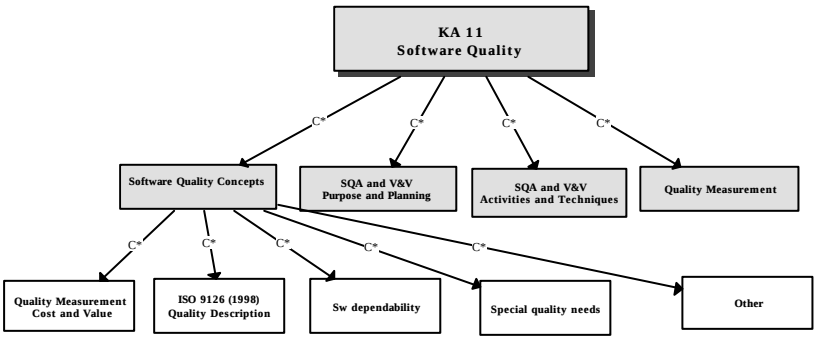

Figure 4: The Software Quality Ontology (a partial view, levels 1 and 2)

Formatted

The above topic of the SWEBOK ontology represented in the OWL formalism is depicted in figure 5. Topics contained in the area of Software Quality knowledge are shown in the left side panel. The subtopics integrating the first element (Software quality concepts) are also partially shown. The central widget (asserted conditions) are used to compose the axioms (logical expressions) that describes (using a set of necessary conditions) and define (using sets of necessary and sufficient conditions) the concepts that integrate the SWEBOK ontology.

With concept *KA11 Software Quality* as an example, some axioms are shown: the KA11 Software Quality is an area of Knowledge, part of the SWEBOK guide that has other areas (mutually disjointed). The axioms describe also that Software Quality is composed of four topics (Quality concepts, SQA and V&V Purpose and planning and SQA and V&V Activities and techniques)

The OWL widget (at the right side) contains the properties (attributes and relationships, describing the concept and linking this one to other concepts). As an example, three inherited and modified (locally overriden) properties are shown: KA11 has authors, is part of the SWEBOK guide and has topics (four already mentioned).

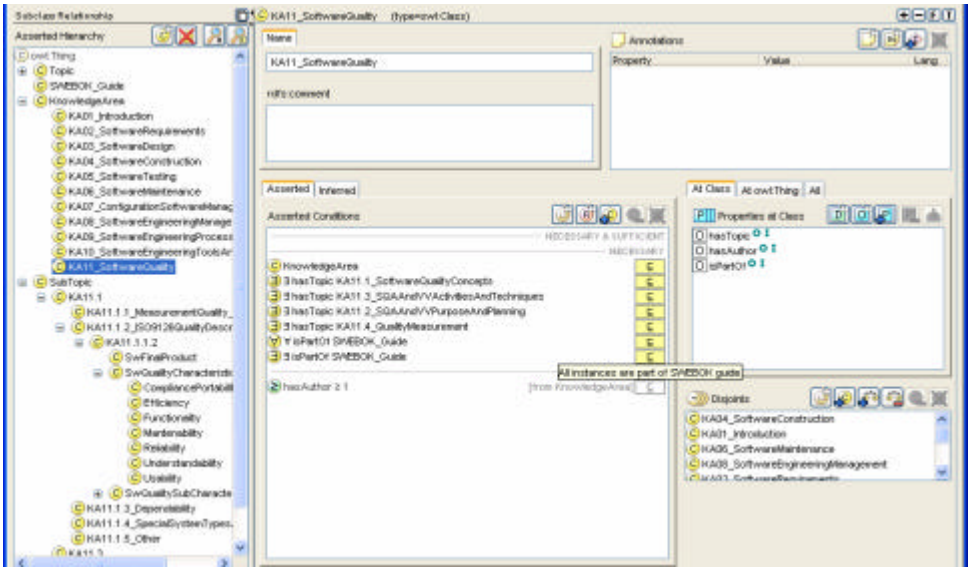

Figure 5: The Software Quality Ontology in OWL(a partial view), levels 1 and 2

8. Conclusion

This paper has presented the results of the first phase of a project aimed at developing a comprehensive ontology of the Software Engineering field. The major contributions provided by this study are: 1) Identification of the ontology development methodologies providing the best guidance to attaint our established goal; 2) Identification of a life-cycle model best suited to the planned ontology development; 3) Identification of main inputs, outputs and activities to be performed in order to develop the aimed ontology; 4) Identification of key activities in the ontology development process. Some preliminary results of the software quality ontology are also presented and developed using the May 2001 version of the SWEBOK Guide.

References

[1] T.R. Gruber. *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.

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

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

[4] C. Wille, R. Dumke, A. Abran, J-M, Desharnais, *E-learning Infrastructure for Software Engineering Education: Steps in Ontology Modeling for SWEBOK*, in Ontology Modeling for SWEBOK , in Software Measurement European Forum , Rome, Italy , 2004

[5] A Qasem, *A prototype DAML+OIL Ontology IDE*, International Semantic Web Working Symposium, Stanford, 2001. http://www.semanticweb.org/SWWS/program/position/s oi-qasem.pdf

[6] A Qasem, *The WOSE Portal* URL: http://java-emporium.com/projects/wose/index.html

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

[8] 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 2001, URL: http://www.swebok.org

[9] M. Uschold and Michael Gruninger *Ontologies; principles, Methods and Applications*, Knowledge Engineering Review, Vol 11, No 2, Jun 1996

[10] D. Jones, T. Bench-Capon and P. Visser, *Methodologies for Ontology Development*, Proceedings of the IJCAI-99 workshop on Ontologies and Problem-Solving Methods, 1999

[11] R. Mizoguchi, *Fundamental Aspects of Ontology Engineering*, to appear in Proceedings of the ACFAS Congress, Colloque d'Informatique Cognitive (C622), Montréal, May 2004

[12] M. Uschold, and M. King. *Towards a Methodology for Building Ontologies*. Proceedings of IJCAI95's Workshop on Basic Ontological Issues in Knowledge Sharing. 1995.

[13] M. Fernandez, A. Gomez-Perez, and N. Juristo. *METHONTOLOGY: From Ontological Art to Ontological Engineering*. In Workshop on Knowledge Engineering: Spring Symposium Series (AAAI'97), pages 33-40, Mellow Park, Ca, 1997. AAAI Press.