Experiences in Using a Method for Building Domain Ontologies

Ricardo de Almeida Falbo
Computer Science Department, Federal University of Espírito Santo
Fernando Ferrari Avenue, CEP 29060-900, Vitória - ES - Brazil
falbo@inf.ufes.br

Abstract. Since 1997 we are working in building domain ontologies. During this period of time, we have developed several ontologies using a systematic approach for building ontologies, first published in 1998, and now called SABiO. In this paper we discuss strong points and weakness of this method for building ontologies, presenting some lessons learned and improvement opportunities.

1. Introduction

Building domain ontologies is not a simple task. Like any complex software modeling activity, to build quality ontologies we need methods and tools to support their development. In 1997, we defined a systematic approach for building ontologies (SABiO), first published in 1998 [1]. SABiO was proposed based on Uschold and King skeletal methodology [2], adding some features to improve it, such as a graphical languages for expressing ontologies, an axiom classification, and the use of competency questions, as proposed by Gruninger and Fox [6]. Since then, we have been using this approach to build several domain ontologies, such as an ontology of software process [1], an ontology of software metrics [3], an ontology of the port domain [4], and an ontology of steel metallurgy, among others.

In this paper we discuss our experience in building domain ontologies using SABiO, focusing on lessons learned and improvement opportunities. Section 2 briefly presents SABiO, and some improvements made along these years of use. Section 3 discusses its strengths, weaknesses and some lessons learned. Section 4 presents some improvement opportunities to evolve SABiO. Finally, section 5 reports our conclusion.

2. A Systematic Approach for Building Domain Ontologies

According to Guarino [5], an ontology is an engineering artifact, constituted by a vocabulary used to describe a certain reality, plus a set of explicit assumptions (formal axioms) regarding the intended meaning of the vocabulary words. This set of assumptions has usually the form of a first-order logical theory, where vocabulary words appear as unary or binary predicate names, respectively called concepts and relations.

Like any other conceptual modeling activity, ontology construction must be supported by software engineering practices. Thus, we need methods and tools to support ontology engineering. In 1997, we proposed SABiO, a Systematic Approach for Building Ontologies [1], that encompasses the following activities:

- Purpose identification and requirement specification: concerns to clearly identify the ontology purpose and its intended uses, i.e. the competence of the ontology. To do that, competency questions are used.
- Ontology capture: the goal is to capture the domain conceptualization based on the ontology competence. Relevant concepts and relations should be identified and organized. A model using a graphical language and a dictionary of terms should be used to aid communication with domain experts.
- Ontology formalization: aims to explicitly represent the conceptualization captured in a formal language.
- Integration of existing ontologies: during ontology capture or formalization, it could be necessary to integrate the current ontology with existing ones, in order to use previously established conceptualizations.
- Ontology evaluation: the ontology must be evaluated to check whether it satisfies the specification requirements. It should be evaluated in relation to the ontology competence and some design quality criteria, such those proposed by Gruber [7].
- Documentation: all the ontology development must be documented.

During ontology capture, the use of a graphical representation is essential in order to facilitate the communication between ontology engineers and experts. Such representation is basically a language representing a meta-ontology, and thus this language must own basic primitives to represent a domain conceptualization [1].

SABiO proposed the use of LINGO [1], a graphical language for expressing ontologies. In its first version, LINGO had notations for representing concepts, relations, and properties, and some types of relations that have a strong semantics, such as subsumption and whole-part relations. For each one of these types of relations, a specialized notation was proposed. In fact, this was the striking feature of LINGO and what made it different from other graphical representations: any notation, beyond the basic notations for concepts, relations and properties, aims to incorporate an axiomatization. During its use, some new notations were incorporated to LINGO to address other types of relations, always defining explicitly the axiomatization imposed by them.

More recently, we decided to allow ontology capturing in UML too [4], since UML has also been used as an ontology modelling language [8], and we cannot ignore that UML is a standard and its use is widely diffused. Based on that, we defined a subset of UML's elements that plays the same role of LINGO's notation, i.e., these UML's model elements are applied using the same semantics imposed by the corresponding elements in LINGO. For instance, the epistemological axioms imposed by the whole-part relation are assumed to be incorporated to the ontology when the aggregation notation of UML is used. A lightweight extension of UML was proposed, using stereotypes [4].

A graphical model is useful, but it is not enough to completely capture an ontology. Axioms should be provided in order to fix the semantics of the terms, and to establish domain constraints. To guide axiom definition, SABiO uses an axiom classification that considers two classes of axioms: *derivation axioms*, which allow new information to be derived from the previously existing knowledge, and *consolidation axioms* that define constraints for establishing a relation or for defining an object as an instance of a concept.

Derivation axioms can concern the meaning of the concepts and relations in the ontology, or the way these concepts and relations are structured. When axioms are defined to show constraints imposed by the way concepts are structured, we call them *epistemological axioms*. When they describe domain signification constraints, we call them *ontological axioms*. This distinction is important to guide the ontology engineering defining axioms. Epistemological axioms can be assumed to be captured by the graphical notation, and should not be explicitly written. Ontological axioms, in turn, are not captured by the graphical notation, and need to be explicitly defined. In Figure 1, we show part of the software process ontology defined in [1], written in UML. In this figure, the aggregation notation imposes some axioms, such as:

$\forall a \ \neg subActivity(a,a)$

$\forall a1,a2 \ \ subActivity(a1,a2) \rightarrow \neg\, subActivity(a2,a1)$

$\forall a1,a2,a3 \ subActivity(a1,a2) \wedge subActivity(a2,a3) \rightarrow subActivity(a1,a3)$

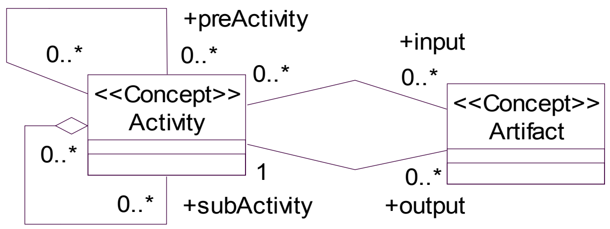

Figure 1 - Part of the software process ontology.

These axioms are part of the mereological theory, which says that whole-part relations are irreflexive, anti-symmetric and transitive, respectively, and do not need to be written by the ontology engineer, since they are epistemological axioms.

In the same ontology, however, there is the following ontological axiom:

$\forall a1,a2,s \ input(s,a2) \wedge output(s,a1) \rightarrow preActivity(a1,a2)$

This axiom does not refer to the way concepts are structured, and thus, cannot be captured by the graphical notation. It is an ontological axiom and must be written down by the ontological engineer. This way, the distinction between epistemological and ontological axioms indicates which axioms must be written by the ontology engineer.

Going back to the activities of the ontology development process shown in Figure 1, to formalize ontologies, SABiO suggests the use of first order logics, and gives some guidelines to perform this step [1].

In ontology evaluation, SABiO suggests checking the ontology against its competency questions, and to verify some quality criteria, as pointed early.

Finally, for documentation purposes, SABiO advocates the use of hypertexts. Using a hypertext, concepts can be easily linked to relations, properties, ontology diagrams, dictionaries of terms, axioms, and competency questions. This way, people can browse the ontology to learn about the domain.

3. Strong Points, Weakness and Lessons Learned

After we had used SABiO in several ontology developments, we can point out some benefits and some weakness of the method.

Concerning strong points of SABiO, we can highlight:

- The set of activities, artifacts and guidelines proposed by the ontology development process of SABiO showed to be good. It can be considered part of a standard software process for building ontologies, but we need more.
- The use of competency questions showed to be very useful to guide ontology capturing, formalization and evaluation. Concepts, relations, properties and axioms in an ontology should be those necessary and

sufficient to address the competency questions, as pointed by Gruninger and Fox [6].

- The use of a graphical language for expressing ontologies proved to be essential for ontology capture. It is very hard to communicate with domain experts without it. More over, the epistemological axioms incorporated to the graphical notation free ontology engineers to concentrate in some classes of axioms, in spite of having to consider all of them.
- The axiom classification also proved to be a good guideline to drive the axiom definition. Based on it, ontology engineers can inspect the world looking for axioms that consider the structuring of the concepts and relations (the epistemological axioms), their meanings and constraints (the ontological axioms) and the integrity laws that govern them (the consolidation axioms). But the first class of axioms do not need to be written down.
- Hypertext proved to be an adequate format for documenting ontologies. Using hypertexts, ontologies can be easily browsed, and people can use them to learn about the domain.

But SABiO has also weaknesses, such as:

- SABiO does not address important activities of a software process, as recommended in Software Engineering, such as planning and configuration management. Regarding the last, in fact, SABiO says nothing about ontology evolution.
- Concerning competency questions, SABiO says nothing about formal competency questions. We think they are very important. But we need tools for verifying ontologies in the light of them.
- Although LINGO has a strong semantics, it is "another modeling language". This is a recurrent claim. Many ontology engineers do not know it, and sometimes use it in an inappropriate way. Several times, we notice that notations were not being correctly used, and the models were not well interpreted by ontology engineers. When we started to use UML as modeling language, some of these problems attenuated. On the other hand, sometimes, ontology engineers with background in software engineering used some UML constructions that are not expected in ontology building, and consequently without precise semantics.
- As to axiom classification, sometimes ontology engineers have doubts about how to classify an axiom. The most common problems are about some epistemological axioms, like those imposed by cardinalities. Cardinalities, for instance, express domain constraints, and thus ontology engineers tend to classify them as ontological axioms, in spite of they are related to structural concerns.
- A first order predicate logic language for formalizing ontologies is good due to its expression power. But it is difficult to evaluate an ontology formalized using it, since we do not have inference engines capable to do that. Other languages, such DAML+OIL [9] and KIF [10], could be better choices, since we can use an inference engine to verify the ontology. Competency questions could be formalized and submitted to the inference engine to check if the ontology satisfies them. But some of them, like DAML+OIL, are less expressive languages.
- Ontology integration in SABiO is extremely superficial. Nothing is said about consistency and coherence of the model elements imported to a new ontology.

Finally, we can enumerate some lessons learned. First, like any other software product, ontology building must be conducted as a quality software process. As a software process, we need tools to support ontology building. Ideally, such tools have to allow competency question definition and formalization, ontology capture using a graphical language, axiom definition and formalization, ontology integration, ontology verification and validation, ontology documentation, and ontology evolution.

Second, especially in ontology capturing we need to achieve consensus from experts. Books, papers, manuals, web pages and other literature sources are very important for capturing an ontology, but they are not enough. We need experts, and need to achieve consensus between their positions. In this process, Gruber's minimum ontological commitment criterion [7] is very useful. In all work we have been done, we needed to apply this criterion in order to achieve consensus.

Third, in ontology building, evaluation regards the set of activities that ensure that the ontology concepts, relations, properties and axioms answer appropriately the competency questions. Two questions have to be answer: "Are we building the ontology right?" and "Are we building the right ontology?" The first one regards ontology verification, the second ontology validation. In both cases, evaluation implies to check each competency question, looking if it is being correctly answered. For this purpose, we need tools to support those activities, since they are hard tasks to be done manually. Particularly in ontology validation, experts are essential. In ontology validation not only we are checking if the competency questions are being correctly answered, but we are also checking if the competency questions actually pose the right questions for the ontology purpose.

Finally, although hypertexts proved to be an excellent way to document ontologies, we need tools to automate, at least partially, their construction. Ontology engineers spend a substantial amount of time developing the ontology documentation. Documentation functionalities integrated into an ontology editor is an important opportunity to improve productivity.

4. Improvement Opportunities

Based on the weaknesses of SABiO, we can devise some improvements to evolve it to a better approach for building ontologies:

- It is worthwhile to define a standard software process for building ontologies, in the sense of Software Engineering. Planning activities and methods to do that should be investigated. There are few works addressing this important issue. Metrics for evaluating ontology development should also be provided. Software Engineering experience can serve as basis, but we need to adapt it to better fit ontology development.
- Regarding a modeling language for expressing ontology, we think that the use of a lightweight extension of UML, such that proposed in [11] is a promising way. We are now studying how to incorporate it to SABiO.
- We should refine the guidelines for classifying axioms in order to clarify the categories. Also, we are studying how relation meta-properties, such as transitivity and symmetry, can be integrated to our axiom classification. These are very frequent axioms, and so it is worthwhile to better support their capture.
- During ontology formalization, competency questions should be formalized. In ontology evaluation, they should be submitted to an inference engine to check if the ontology satisfies them.
- SABiO needs to better address ontology integration. In its current version, all important decisions are left to the ontology engineers. We need to better study this activity to improve the guidelines offered to it.
- SABiO does not consider ontology maintenance or evolution. Since we are now working in some ontology evolutions (this is the case of the software process ontology [1]), we intend to improve SABiO with practical guidelines to address ontology evolution.

5. Conclusions

Building domain ontologies is not a simple task. We need methods, tools and guidelines to drive ontology engineers in performing their activities. Software engineering practices should be incorporated to ontology development, and SABiO goes a step ahead towards a defined ontology development process.

In this paper we presented some reflections regarding the strengths and weaknesses of SABiO, and discussed some lessons learned and improvement opportunities.

Our experience in ontology development highlights an important issue concerning tool support. We would not be able to scale up ontology building without good ontology editors. Fortunately, now there are some of them available, such as OILEd [12]. We are also working on ODEd [4], an ontology editor that minimizes some of the reported problems, such as formalization and evaluation.

Acknowledgments

This work was accomplished with the support of CNPq, an entity of the Brazilian Government reverted to scientific and technological development.

References

[1] R.A. Falbo, C.S. Menezes, A.R.C. Rocha. “A Systematic Approach for Building Ontologies”. Proc. of the 6th Ibero-American Conference on Artificial Intelligence, Portugal, Lecture Notes in Computer Science, vol. 1484, 1998.

[2] M. Uschold, M. King. “Towards a Methodology for Building Ontologies”, Workshop on Basic Ontological Issues in Knowledge Sharing, IJCAI’1995.

[3] R.A. Falbo, G. Guizzardi, K.C. Duarte. “An Ontological Approach to Domain Engineering”. Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering, SEKE'2002, pp. 351- 358, Ischia, Italy, 2002.

[4] P.G. Mian, R.A. Falbo. “ Building Ontologies in a Domain Oriented Software Development Environment”. *Proceedings of the IX Argentine Congress on Computer Science,* pp. 930 – 941, La Plata, Argentina, 2003.

[5] N. Guarino. Formal Ontology and Information Systems. In N. Guarino (Ed.), *Formal Ontologies in Information Systems*, IOS Press, 1998.

[6] M. Grüninger, M.S., Fox. Methodology for the Design and Evaluation of Ontologies. *Technical Report*, University of Toronto, 1995.

[7] T.R. Gruber. Toward principles for the design of ontologies used for knowledge sharing. *Int. Journal Human-Computer Studies*, 43(5/6), p. 907-928, 1995.

[8] S. Cranefield, M. Purvis. UML as an Ontology Modelling Language, In *Proceedings of the IJCAI-99, Workshop on Intelligent Information, 16th International Joint Conference on AI*, Stockholm, Sweden, July 1999.

[9] D. Connolly, F. van Harmelen, I. Horrocks, D.L. McGuinness, P.F. Patel-Schneider, L.A. Stein. "DAML+OIL Reference Description", December 2001.

[10] M.E. Genesreth, R.E. Fikes. “Knowledge Interchange Format, Version 3.0 Reference Manual”. Tech. Rep. Logic-921, Computer Science Dept., Stanford University, 1992.

[11] G. Guizzardi, H. Herre, G. Wagner, “Towards Ontological Foundations for UML Conceptual Models”, 1st International Conference on Ontologies, Databases and Application of Semantics (ODBASE'2002), Irvine, California, USA, 2002.

[12] S. Bechhofer, I. Horrocks, C. Goble, R. Stevens. “OilEd: a Reason-able Ontology Editor for the Semantic Web”. Working Notes of the 14th International Workshop on Description Logics (DL-2001), pp.1-9, USA, August 2001.