Integrating Domain Ontologies into Knowledge-Based Systems

Frédéric Fürst, Francky Trichet
LINA, CNRS-FRE 2729, University of Nantes
2 rue de la Houssinière - BP 92208
44322 Nantes CEDEX 03, France
{furst,trichet}@lina.univ-nantes.fr

Abstract

The work presented in this paper deals with the integration of heavyweight ontologies into Knowledge-Based Systems (KBS). We claim that such ontologies have to be built at the conceptual level, and that their use in a KBS requires an operationalization step, that consists in transcribing the ontology in an operational knowledge representation language according to a given scenario of use. For this purpose, we propose TooCoM, a tool based on the Conceptual Graphs model and dedicated to the edition and the operationalization of heavyweight ontologies.

Introduction

An ontology aims at representing knowledge of a domain at the conceptual level, independently of any operational goal, in order to allows users to reuse the same ontology in different Knowledge-Based Systems (KBS). Moreover, using ontologies to improve the efficiency of reasoning requires the representation of the whole semantics of the considered domain, including well-known properties such as the subsomption, but also any kind of *axioms*. These axiom-based ontologies are called *heavyweight ontologies*, in contrast to *lightweight* ontologies that only include a limited set of properties (Gomez-Perez, Fernandez-Lopez, & Corcho 2003). Consequently, we propose a language, called OCGL (Ontology Conceptual Graphs Language), dedicated to the representation of heavyweight ontologies at the conceptual level, and based on a graphical syntax inspired from those of the Conceptual Graphs model (CGs) (Sowa 1984).

In order to effectively use a heavyweight ontology in an operational KBS, an operational semantics must be added to its axioms, via an *operationalization* process which consists in first, specifying the way the axioms will be used in the KBS, and, secondly, transcribing the ontology in an operational knowledge representation language, according to the specifications of the contexts of use of each axiom. We propose a general operationalization method for ontology, applied to the operationalization of heavyweight ontologies in CGs. This method is implemented in a tool, called TooCoM, *a Tool to Operationalize an Ontology with the Conceptual Graph Model*, dedicated to the edition of ontologies in OCGL and to their operationalization in CGs.

Editing heavyweight ontologies with TooCoM

Building an ontology in OCGL consists in (1) specifying the conceptual vocabulary through concepts and relations and (2) specifying their semantics through *axioms schemata* and *domain axioms*. Axiom schemata are well-known properties which can correspond to *is-a* links between two concepts or two relations, abstractions of concepts, disjunctions between concepts, signatures of relations, algebraic properties of relations, exclusivity or incompatibility between two relations or maximum and minimum cardinalities of a relation.

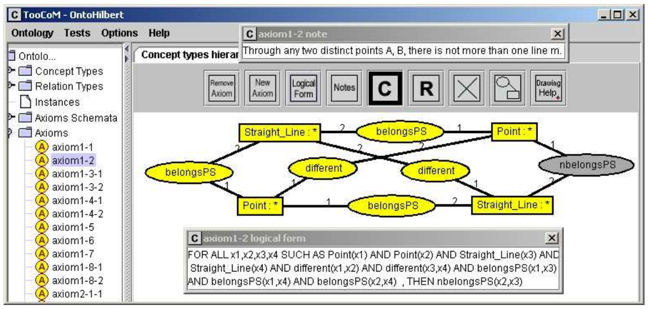

Figure 1: Representation of an axiom in TooCoM. The antecedent part is in bright and the consequent part in dark. Semantics of this axiom is as follows: *« Through any two distinct points A, B, there is not more than one line m »*.

Domain axioms are properties that do not correspond to axiom schemata. An axiom is composed of an antecedent part and a consequent part, with the following intuitive formal semantics: *if the antecedent part is true, then the consequent part is true*. Figure 1 presents an axiom edited in TooCoM[1]. A concept node is labeled with the concept label and a marker that identifies the considered instance. The marker $*$ denotes an undefined instance. A relation node is labeled with the relation label. An edge between a concept and a relation is labeled with the position of the concept in the signature.

[1] TooCoM is available under GNU GPL license at http://sourceforge.net/projects/toocom/.

Operationalizing heavyweight ontologies

The axioms only constrain, through their *formal semantics*, the way the terminological primitives can be manipulated, but, they do not precisely specify, through an *operational semantics*, in what way these primitives are precisely used. This operational semantics is determinated by the considered application, whereas the formal semantics depends on the considered domain. The operationalization method we propose relies on the specification of the *scenario of use* of the ontology in a given KBS, by describing how the ontology, and in particular the axioms, will be used in the KBS. Generally speaking, knowledge is used in KBS to produce new knowledge or to validate existing knowledge. The knowledge manipulation can either be done automatically by the system or be driven by the user. The scenario of use of an ontology is then composed of all the *contexts of use* of the axioms (domain axioms and axiom schemata) of the ontology (*cf.* figure 2). The contexts of use we propose are those which correspond to the combination of the criteria previously given, that are (1) *inferential and explicit* context of use where the user applies the axiom by himself on a fact base to produce new facts, (2) *inferential and implicit* context of use where the axiom is automatically applied by the system on a fact base to produce new facts, and (3) *validation and implicit* context of use where the axiom is applied by the system to verify that a fact base is in accordance with the semantics of a domain. The *validation and explicit* context of use is not taken into account, because if the user has the choice to control or not the accordance of a fact base with an axiom, this accordance can not be certain.

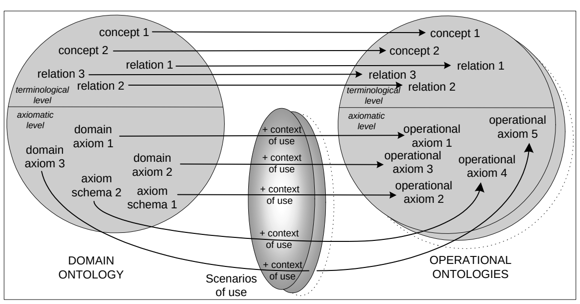

Figure 2: The detailed process of operationalization of an ontology. Each scenario of use, *i.e.* combination of contexts of use, produces a new operational ontology.

For the axiom schemata, the same context of use can be specified for all the axioms that correspond to a given schema. For example, the user can choose an inferential and implicit context of use for all the axioms that express a symmetry relationship, in order to automatically produce symmetric relations. This is why our work is particularly relevant for heavyweight ontologies: the operationalization of lightweight ontologies is immediate since the classical properties can be straight expressed at the operational level. This general methodology for operationalizing an ontology has been applied to automate the operationalization of ontology in the context of the CGs (Fürst, Leclère, & Trichet 2004).

Operationalizing ontologies in CGs

In the context of CGs, knowledge is expressed by specifying concepts and relations, subsomption links between concepts and between relations, signature of the relations, instances of concepts, and facts represented by graphs built with the concepts and relations. The SG-family adds rules and constraints, which are both composed of an hypothesis graph and a conclusion graph (Baget & Mugnier 2002). The operational semantics of a rule is *« if the hypothesis part is present in a graph G, then the conclusion part can be added to G »*, negative constraint semantics is *« if the hypothesis part is present in a graph G, then the conclusion part must be absent in G »* (otherwise the constraint is broken by G) and positive constraint semantics is *« if the hypothesis part is present in a graph G, then the conclusion part must be present in G »* (otherwise the constraint is broken by G). So, in TooCoM, operationalizing an axiom in a particular context of use consists in transcribing this axiom in a set of rules and/or constraints, implicitly or explicitly used, that implement the formal semantics of the axiom and the operational semantics given by the choosen context of use (*cf.* figure 3).

	Inferential and Implicit Context of use	Inferential and Explicit Context of use	Validation and Implicit Context of use
axiom schema that concerns concept	∅	∅	∅
incompatibility and exclusivity	1 implicit negative constraint + n implicit rules (for a n-ary relation)		
axiom schema (that concerns relation) and axiom with only relations in the consequent	1 implicit rule	1 explicit rule + a set of negative and implicit constraints	a set of negative and implicit constraints
axiom with concepts in the consequent	1 implicit rule	1 explicit rule	∅

Figure 3: Operationalization rules for the GCs model.

Conclusion

In this paper, we present OCGL, an ontology representation language, based on the CGs model, which allows to represent heavyweight ontologies at the conceptual level. To bridge the gap between domain ontologies, built at the conceptual level, and KBS, that rely on an operational semantics, we propose an operationalization method, applied to the particular case of the CGs and implemented in TooCoM.

References

Baget, J., and Mugnier, M. 2002. Extensions of Simple Conceptual Graphs: the Complexity of Rules and Constraints. *Journal of Artificial Intelligence Research* 16:425–465.

Fürst, F.; Leclère, M.; and Trichet, F. 2004. Operationalizing domain ontologies: a method and a tool. In *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI'2004)*, volume 110, 318–322. IOS Press.

Gomez-Perez, A.; Fernandez-Lopez, M.; and Corcho, O. 2003. *Ontological Engineering*. Springer-Verlag.

Sowa, J. 1984. *Conceptual Structures: Information Processing in Mind and Machine*. Addison-Wesley.