Do We Need Metamodels AND Ontologies for Engineering Platforms?

Marie-Noëlle Terrasse[1], Marinette Savonnet[1], Eric Leclercq[1], George Becker[2], Thierry Grison[1]
1) Laboratoire LE2I, Université de Bourgogne,
firstname.lastname@u-bourgogne.fr
2) *gbecker@nerim.net*

Abstract

In this paper we show how the joint use of metamodeling and ontologies allows to describe domain knowledge for a complex domain. Ontologies are used as stabilized descriptions of the business domain while metamodels allow a fine description of the domain (to be constructed in the initial phases of modeling). We propose to use an ontology for anticipated categorization, i.e., as a "natural" complement of the formal system which is induced by the metamodel.

1 Introduction

Building an application of a considerable size poses –whatever be its application domain– a series of recurrent problems. Three of these problems (which we discuss below) border on fundamental axes of information system engineering: the know-how (which is expressed by the business process and which is often treated at the computer science level by means of business patterns), the structuring of knowledge (which takes form of thesaurus, nomenclatures or speciality vocabularies, and which is often treated at the computer science level by means of ontologies), the technical basis (which is often treated at the computer science level by means of architecture patterns).
First, applications of a substantial size are based not only on one business domain but on several ones. Each of these domains has its vocabulary and its own nomenclatures which it is advisable to respect. An information system of large scope shows all the characteristics of a cooperative information system. The extent of the actual domain makes it often difficult to use classical tools (ontology, thesaurus), because scaling up of these tools is not guaranteed.
Second, most of the time end-users expect from a new information system new services (in comparison to those proposed by the existing system). Yet these same end-users certainly do not intend to modify their practice and their methods of work. This paradox takes sometimes an extreme form when the wish not to change the practice and the demanded changes of functionalities lead to contradictions. Goepp & al. [12] present examples of such cases and propose a "dialectic" approach[1].
Third, in order to offer high-quality services to end-users, it is tempting to turn to leading-edge technologies (e.g., mechanisms of authentication in e-administration) and to very specialized tools whose handling by novices is difficult (e.g., setting up of image processing tools in order to retrieve information from cadastral maps). Since end-users of large applications come from various company departments, they have at their disposal unequal computer resources and can possess skill levels very different from each other. Certain users run a risk of being eliminated de facto, which is unacceptable and should be taken into account as early as possible in the engineering process.

In this paper we show how the joint use of metamodeling architectures and ontologies allows to describe knowledge linked with a complex domain. For this we start from the following statements:

- The description of a complex domain is built on various types of knowledge whose expression is not generally possible by using a single language.

[1] Their main example is that of an installation of a system of component followup on a complex assembly line. They weigh up the need of component localization with the cost (in the term of modifi cations of business process) of a precise localization.

- The description of a complex domain is often constructed from several partial descriptions. These partial descriptions are combined by more or less sophisticated operations (e.g., alignement, fusion). The difficulty of combining partial descriptions depends considerably on their respective positioning: on different domains, at different semantic levels, on the same domain seen at the same abstraction level but with different objectives [10].
- A formal description is immediately operational but, for a complex domain, it is rarely possible to directly produce a formal description. The teams of knowledge modeling pass rather through a sequence of descriptions: conceptual, semi-formal, formalized [10].

Our paper is organized as follows. Section 2 introduces the notions of metamodels and ontologies, as well as their connections. Section 3 explores stakes tied to information systems of complex domains and sets the outline of requirements to be taken into account. Section 4 outlines our proposal for a domain platform architecture. Section 5 sums up the questions opened by our platform proposal.

2 Metamodels, Ontologies, and Categorization

Many authors have discussed the proper role of metamodels and ontologies in modeling and engineering [4, 11, 15, 25]. We believe that the hard core of knowledge of a complex domain can be reduced to a pair of metamodel and ontology[2]. The ontology (or, failing that, the thesaurus) is a stabilized description of the domain inasmuch as speciality vocabularies are most often an integral part of the core business. The metamodel allows a fine description of the domain but this knowledge may not be stabilized (in the initial phases of work). The combination of two description languages allows to work with reasonable reliability by integrating necessary metamodels "on the fly". This allows to obtain at low cost (in terms of time and money) a beta version of the domain description. The objective of this section is to justify our approach. The first part of this section consists in a comparative study of ontologies and metamodels. The second part introduces the notion of anticipated categorization.

2.1 Comparative Study

Ontologies have been defined in very different ways [14, 24]. We use the definition proposed by Sowa [27, 29]:

"*The subject of ontology is the study of the categories of things that exist or may exist in some domain. The product of such a study, called an ontology, is a catalog of the types of things that are assumed to exist in a domain of interest D from the perspective of a person who uses a language L for the purpose of talking about D*"

This definition allows to consider an ontology –being categorization tool– as a "natural" complement of a formal system. Just as metamodels, ontologies have a scope (a domain of validity) as, for example, general ontologies [20, 21, 41], or domain ontologies [9, 35, 37]. However, ontologies are most often considered as extensible [16], alignable or mergeable [18, 22, 30]. In order to facilitate these operations on ontologies, a certain number of top level categories have been defined. These categories present the outline of ontology description languages (concepts and relations between concepts). Sowa [28] offers a set of ten top level categories, Welty & al. [34] propose a classification of relations, Frst [10] proposes a synthesis of recent classification work.

We propose to establish the following parallels between the work on ontologies and those on metamodeling (as conducted in the sphere of influence of the OMG):

Statement of minimum characteristics of description languages by top level categories for ontologies and by MOF in metamodeling. In both cases this description is proposed as a reference of alignement of languages.

The main difference is between the integration of top-level categories with the ontologies, and the separation in metamodeling architectures of meta-metamodeling, metamodeling, and modeling concepts and constructs. Another difference is the relative recognition of MOF by the metamodeling community, while the set of top level categories is still in discussion in the community of ontologies.

[2] According to Elvesaeter [6]: "*The models at the various levels may be semantically annotated using ontologies which help to achieve mutual understanding on all levels*"

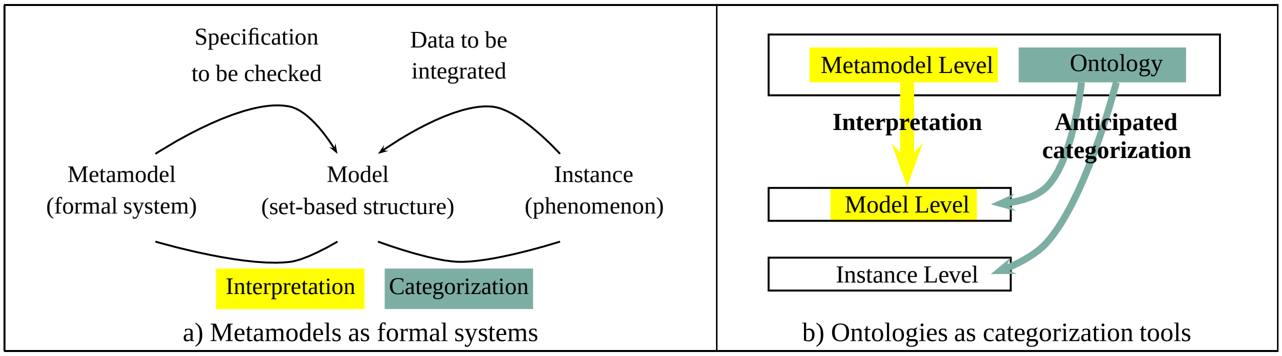

Figure 1: Anticipated categorization

Combination of descriptions in order to treat complex domains: by operations on ontologies (e.g., alignment, fusion) or by operations on metamodels (integration, weaving) and models.

In both cases, operations to be used depend on semantic and structural proximity of the described knowledge, and as well as on the relative positioning of descriptions to be combined: whether or not at the same abstraction level, with or without covering of described domains, etc.

Existence of semi-formalized descriptions with conceptual ontologies and metamodel-model pairs. In both cases, these descritions should evolve towards an operational form. This transition to the operational level can in both cases pass through a translation into a formal language. As far as the metamodel-model pair is concerned, a translation into a programming language is however more current.

As a conclusion of our comparative study, we can state that metamodels and ontologies can be used jointly for domain description since they present enough common "deap" characteristics. In the following section, we show which advantages we can expect from such a joint use of metamodels and ontologies.

2.2 Anticipated Categorization

Metamodeling levels are defined in accordance with the model theory [31], see Figure 1.a. A metamodel is a formal system; models are interpretations of their metamodels; the instance level makes the model's categorization explicit. If we stick to that point of view, categorization does not exist before implementation of an information system: it cannot help in designing or instantiating the system.
As depicted in Figure 1.b, a semantical coupling of a metamodel with an ontology allows describing –at high abstraction levels– complex application domains.

- The metamodel expresses an abstract yet well-established basis of knowledge as much as we can guarantee that metamodels are aligned to world-wide recognized modeling concepts.
- The ontology provides an anticipated categorization since real-life objects are actually classified in ontologies. Three levels of coupling can be used in order to obtain full benefits from such an anticipated categorization. We have experimented with the first two ones in the IkoSem project[3]. We are now working on the third level. At the model level, we use sub-domains of ontologies for definition of attribute domains. At the instance level, we use the ontology for evaluating data quality (e.g., detection of non-fitting attribute values which can be classified, by using the ontology, into actual counter-senses or imprecise values).

  At the metamodel level, we are now coupling ontology top-level concepts and relations with UML constructs (e.g., is-part-of relations with compositions).

[3] IkoSem is a framework for image database prototyping [1, 2, 3]

3 Stakes Under Consideration

Since the principe of coupling of an ontology with a metamodel is set out, it is advisable to decide –in function of the pursued objective– which operational mechanisms will be used together with the ontology-metamodel pair. It seems that executable models are in a position to validate and test modeling. Numerous propositions for code generation come from the MDE community whose objectives are: to clearly define methodologies, to develop systems at various abstraction levels, to organize and automate testing and validation activities [8]. This implies intensive work on models, extended to the metamodel level.
It is however possible to note that metamodeling approaches go considerably beyond the scope of OMG proposals, e.g., the Megamodel proposal [7]: ModelWare, GrammarWare, GraphWare; EAI (Enterprise Application Integration) and MIC (Model Integrated Computing) [19]. For Khn & al. [19] a comprehensive metamodeling plateform should contain (in addition to standard components: meta-metamodel, base of metamodels and models, mechanisms of manipulation of models and metamodels) components for persistance and access, as well as version control, multi-language support, analysis and simulation of models.

Thus, there are several series of stakes in metamodeling environments:

Abstraction levels of metamodeling architectures need to be discussed since components of metamodeling environments tend to cover several levels. Other propositions of levels have been made[4].

Tools of model transformations are based on rewriting rules, whose left and right sides refer to source and target models, respectively; either within the scope of the same metamodel (which serves as a semantical reference), or within the scope of two different metamodels (but having a shared semantic reference, in general at the meta-metamodel level). Several authors [5, 13, 23] emphasize the multiplication of languages of model transformations and set forth the need of convergence towards languages which would be widely used, accessible at the same time by a graphical notation[5] and by a lexical notation (for scaling up), declarative, bidirectional.

Reference to a meta-metamodel is a "sensitive spot" of metamodeling environments. There exists a trend to alignment (in particular in the sphere of influence of OMG) which takes various forms going from incitement of users to tools of metamodel conversion [17]. Yet, platforms such as Adonis [36] or MetaEdit+ [39] have defined their their own meta-metamodels.

With the view of Megamodel [7], an opening up alternative is proposed, in which several reference meta-metamodels are recognized and accepted.

Interoperability is certainly one of crucial stakes of Model Driven approaches. Interoperability seems however deciding at two levels in metamodeling environments. On the one hand, there are risks of semantic heterogeneity which is not controled at the theoretical level with the existence of non-aligned meta-metamodels, of model transformations which do not clearly reference a semantical basis (shared metamodel or meta-metamodel), even of operations on metamodels whose semantics is not clearly defined yet. On the other hand, there is a risk at the level of proposed tools which do not systematically comprise all the necessary components and whose numerous heterogeneities (use of proper metamodels, specific technology of representation of models, non-standandized access mechanisms [40]).

The positioning of our proposal in comparison to these stakes is as follows:

- Preservation of abstraction levels of OMG while emphasizing of the role of the metamodel level for the description of application domains, yet putting a boundary between reuse and contextualization at the model level.
- Work on models which is based on pairs of a generic model with an instantiated model. These pairs are taken into account in the scope of a unique metamodel. During operations on models, our main preoccupation is the guarantee of preservation of contraints in the instantiated model. These contraints have been expressed at the metamodel and model levels.

[4] In the European project ATHENA [38] the proposed levels are *conceptual* (concepts, metamodels, languages, models), *technical* (software development and execution environments), *applicative* (methodologies, domain standards and models).
[5] See, e.g., the graphic notations proposed by Adonis [36] or MetaEdit+ [39].

At the non-functional level, we propose a platform schema consisting of technical specifications based on Open Formats and Open Source tools.

- Restriction to the MOF meta-metamodel (at the moment, we even limit our metamodeling architecture to extensions of the UML metamodel).
- Treatment of interoperability at the metamodel and model levels: the metamodel level is used to determine spots of semantic variations (identified as "high-risk zones" for interoperable applications).

The following section presents an outline of our platform proposal which is based on an experiment, the IkoSem project, that was carried out on a platform for image database engineering.

4 Positionning of our Platform Proposal

We make the following proposal for construction of platforms dedicated to information system engineering in the given domain:

- Study of functionalities necessary to users and study of available knowledge. As far as the knowledge is concerned, we assume that this preliminary study should result in the definition of an organization of terms of the domain in an ontology, or failing that, in a thesaurus[6].

  We assume that the metamodel chosen for the platform is consistent with the links between high-level categories of the ontology.
- Definition of a generic architecture, constructed around the domain-specific metamodel and of a model which is also generic. The generic architecture should allow instantiation in order to constitute the architecture of information systems to be built. This implies three types of control:
  - During the instantiation step, a coupling is introduced between the instantiated model and certain categories of the domain ontology (e.g., using categories for defining ranges of certain attributes). The metamodel-ontology coupling should be used to guarantee the semantics of couplings at the model level.
  - The platform should offer a certain number of software tools that can be combined among them in order to make the instantiated system operational. It is necessary to guarantee that these tools conform to the same semantics as the generic elements to whom they correspond. The metamodel and ontology allow to guarantee such a semantical consistency.
  - Instantiation should not cause the loss of constraints that have been defined at each level: the metamodel level, the generic model level. This last issue is one of our subjects of study in the short term.

The outline that we propose for domain platforms is thus made up of a double integrator (a metamodel integrator based on the MetaSem [32, 33] architecture, and an ontology integrator). These two integrators should allow to determine the knowledge basis associated with the domain platform.

5 Discussion

Our proposal offers the following characteristics:

- The initial systems should be described in the context of a metamodel constructed from a UML metamodel. These metamodels should be introduced into our MetaSem architecture. It is necessary to have available an ontology (or at least a thesaurus) describing the vocabulary which forms the core business of each of the domains concerned by the initial systems.

[6] Elvesaeter [6] : *"In each of these business domains (supply chain management, collaborative product development, e-procurement, portfolio management) we find domain-specific dictionaries, thesauri, nomenclatures, . . . "*. We view such a thesaurus as a simplifi ed form of a terminological ontology.

- The MetaSem architecture allows to build (by integrating necessary metamodels) the reference formal system of the cooperation. The integration of ontologies describes the corresponding categorization. It is thus essential that the integration of ontologies with metamodels are conducted in a consistent way. Several strategies are in general conceivable (with control of choices given either to the metamodel or to the ontology): this is one of advantages of the domain platform to allow testing such strategies.

  A secondary result of our metamodel integration is the knowledge of concepts which are not made to correspond and are thus *spots of semantical variations* between the initial systems and the cooperative system.

- A generic model which can describe the majority of complex domain applications should be created and accompanied by an instantiation guide (explaining the stakes of different choices to make and the couplings to anticipate between the instantiated model and the integrated ontology). The domain platform should allow to test different instantiations of the generic model.

- A certain number of tests can be conducted directly at the metamodel and model levels (if constraints have been introduced in an explicit way into models). Complementary tests can be conducted on the couplings between the formal system proposed (in terms of the integrated metamodel) and the categorization obtained by integration.

  The main stake is to be able to include in the rules of model transformations (and in their execution) verifications such as non-application of transformation rules on the spots of semantical variations without validation by a domain expert, the conformity of transformation rules with metamodels, the consistency of rules with the coupling metamodel-ontology.

Our objective in the short term is to develop several domain platforms in order to be able –if possible– to define several strategies in using the platform (according to the desired semantical consistency). One of the important stakes appears to be able to provide criteria of choice between a global integration strategy (of all the metamodels and all the ontologies), and an integration strategy on demand [26].

References

[1] L. Besson, A. Da Costa, E. Leclercq, and M.N. Terrasse. D'un corpus d'images à une base d'images : une plateforme combinant syntaxe et sémantique et une méthodologie de prototypage. In *CIDE7, Semaine du Document Numérique*, 2004.

[2] L. Besson, B. Leroux, E. Leclercq, and M.N. Terrasse. A Framework and a Methodology for Building an Image Database from a Collection of Images. In *Proceedings of EUROFUSE Workshop on Data and Knowledge Engineering, Poland*, 2004.

[3] L. Besson, J. Savelli, E. Leclercq, and M.N. Terrasse. Ikosem's Generic Model for Syntactical and Semantical Image Description: the Core Component of a Framework for Image Database Engineering. In *Proceedings of the 10th International Workshop on Multimedia Information Systems, Maryland, USA*, 2004.

[4] J. Bzivin, V. Devedžić, D. Djurić, J.M. Favreau, D. Gašević, and F. Jouau. An M3-Neutral Infrastructure for Bridging Model Engineering and Ontology Engineering. In *Proceedings of the INTEROP-ESA*, 2005.

[5] K. Czarnecki and S. Helsen. Classification of Model Transformation Approaches. In *Proceedings of the OOPSLA Workshop on Generative Techniques in the Context of Model-Driven Architecture*, 2003.

[6] B. Elvesæter, A. Hahn, A.J. Berre, and T. Neple. Towards an Interoperability Framework for Model-Driven Development of Software Systems. In *Proceedings of the 1st Int. Conf. on Interoperability of Enterprise Software and Applications, Switzerland*, 2005.

[7] J.M. Favre and T. NGuyen. Towards a Megamodel to Model Software Evolution Through Transformations. *Electronic Notes in Theoretical Computer Science, Proceedings of the Workshop on Software Evolution through Transformations: Model-based vs. Implementation-level Solutions (SETra 2004)*, 127(3), April 2004.

[8] F. Fondement and R. Silaghi. Defining Model Driven Engineering Processes. In *Proceedings of the 3rd Workshop in Software Model Engineering, WiSME 2004*, 2004.

[9] M. Fox and M. Gruninger. Ontologies for Enterprise Integration. In *Proceedings of the 2nd Conference on Cooperative Information Systems*, Toronto, May 1994.

[10] F. Frst. L'ingénierie ontologique. Technical report, Institut de recherche en Informatique de Nantes, France, 2002.

[11] D. Gašević, D. Djurić, and V. Devedžić. MDA Standards for Ontology Development. In *Proceedings of the 4th International Semantic Web Conference, ISWC2005*.

[12] V. Goepp and F. Kieffer. Outils d'analyse dialectique pour la conception d'architecture de systèmes d'information. In *Proceedings du Congrés informatique des organisations et systèmes d'information et de décision, Inforsid'05*, 2005.

[13] R. Gronmo and J. Oldevik. An Empirical Study of the UML Model Transformation Tool (UMT). In *Proceedings of the 1st Int. Conf. on Interoperability of Enterprise Software and Applications, Switzerland*, 2005.

[14] N. Guarino and R. Poli, editors. *Formal Ontology in the Information Technology*, volume 43 of *Special issue of International Journal of Human-Computer Studies*. 1995.

[15] N. Guarino and C. Welty. Towards a Methodology for Ontology-based MDE. In *Proceedings of the First International Workshop on MDE*, 2000.

[16] J. Heflin, J. Hendler, and S. Luke. SHOE: A Knowledge Representation Language for Internet Applications. Technical Report CS-TR-4078, Dept. of computer science, University of Maryland, USA, 1999.

[17] U. Kaufman and W.C. Russell. Interoperability of Knowledge Based Engineering Software. In *Proceedings of the 1st Int. Conf. on Interoperability of Enterprise Software and Applications, Switzerland*, 2005.

[18] K. Kotis and G.A. Vouros. The HCONE Approach to Ontology Merging. In *Proceedings of the European Semantic Web Symposium, ESWS'04*, 2004.

[19] H. Khn and M. Murzek. Interoperability Issues in Metamodelling Platforms. In *Proceedings of the 1st Int. Conf. on Interoperability of Enterprise Software and Applications, Switzerland*, 2005.

[20] D. Lenat. CYC: A Large-Scale Investment in Knowledge Infrastructure. *Communications of the ACM*, 38(11):33–38, November 1995.

[21] G. Miller. WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11):39–41, 1995.

[22] N. Noy and M.A. Musen. PROMPT: Algorithm and Tool for Automated Ontology Merging. In *Proceedings of the National Conference on Artifi cial Intelligence (AAAI), USA*, 2000.

[23] J. Oldevik, T. Neple, and J. Aagedal. Model Abstraction Versus Model to Text Transformation. In *Proceedings of the 2d European Workshop on Model Driven Architecture, EWMDA-2*, 2004.

[24] Letter to B.E. Smith, in collected papers of Charles Sanders Peirce – vols. i-viii (notebook 824). 1931-1958, Harvard University Press.

[25] What are the Differences Between a Vocabulary, a Taxonomy, a Thesaurus, an Ontology, and a Meta-model? Woody Pidcock, URL `www.metamodel.com`.

[26] M.C. Rousset. Small Can Be Beautiful in the Semantic Web. In F. Van Harmelen S.A. McIlraith, D. Plexousakis, editor, *The Semantic Web, LNCS 3298, Proceedings of the Third International Semantic Web Conference, ISWC 2004, Japan*. Springer Verlag, 2003. ISBN: 3-540-23798-4.

[27] J.F. Sowa. Ontology. URL `http://www.jfsowa.com/ontology/`.

[28] J.F. Sowa. *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. 1999.

[29] John F. Sowa. Signs, Processes, and Language Games — Foundations for Ontology. In *Proceedings of the 9th International Conference On Conceptual Structures, ICCS'01*, 2001.

[30] G. Stumme. Ontology Merging with Formal Concept Analysis. In Y. Kalfoglou, M. Schorlemmer, A. Sheth, S. Staab, and M. Uschold, editors, *Semantic Interoperability and Integration*, number 04391 in Dagstuhl Seminar Proceedings, 2005.

[31] A. Tarski. Contributions to the Theory of Models, i. In *Indagationes Mathematicae 16*, pages 572–581, 1954.

[32] M.N. Terrasse, M. Savonnet, G. Becker, and E. Leclercq. A UML-Based Meta-Modeling Architecture with Example Frameworks. WISME 2002, Workshop on Software Model Engineering, Dresden, Germany, Available at URL `http://www.metamodel.com/wisme-2002/terrasse.pdf`, 2002.

[33] M.N. Terrasse, M. Savonnet, E. Leclercq, and G. Becker. Building Platforms for Information System Interoperability. In *Proceedings of the International Workshop on Engineering Federated Information Systems (EFIS), UK*, 2003.

[34] C. Welty and N. Guarino. Supporting Ontological Analysis of Taxonomic Relationships. *Data & Knowledge Engineering*, 39:51–74, 2001.

[35] P. Zweigenbaum. Encoder l'information médicale: des terminologies aux systèmes de représentation des connaissances. Innovation stratégique en information de santé. *ISIS*, 2-3:27–47, 1999.

[36] Adonis - Universal Data Access for Delphi. URL `http://www.cybermagic.co.nz/adonis/`.

[37] Agmes: the Agricultural Metadata Element Set Project. URL `http://www.fao.org/agris/agmes/default.htm`.

[38] Athena European Integrated Project. URL `http://www.athena-ip.org/`.

[39] MetaEdit+ metaCASE tool. URL `www.metacase.com/papers/MetaEditPlus.pdf`.

[40] ModelBus, a Modelware White Paper. URL `http://www.eclipse.org/proposals/eclipse-mddi/main_data/ModelBusWhitePa%per_MDDI.pdf`.

[41] XML Topic Maps (XTM) 1.0 Home Page. URL `http://www.topicmaps.org/xtm/index.html`.