Do We Need Metamodels AND Ontologies for Engineering Platforms?

Marie-Noëlle Terrasse, Marinette Savonnet,
Eric Leclercq, and Thierry Grison
LE2I (UMR CNRS 5158)
UFR Sciences et Techniques
Université de Bourgogne
France
firstname.lastname@u-bourgogne.fr

George Becker
France
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 a 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 early categorization, i.e., as a “natural” complement of the formal system that is induced by the metamodel.

Categories and Subject Descriptors
D.2.2 [Software Engineering]: Design Tools and Techniques

General Terms
Design

Keywords
Metamodel, Ontology, UML, Domain Platform

1. INTRODUCTION
Building an application of a considerable size leads to a series of recurrent problems. Three of these problems (discussed below) touch on fundamental axes of information system engineering: the know-how, the structuring of knowledge, and the technical basis. The know-how is expressed by the business process and is often treated at the computer science level by means of business patterns. The structuring of knowledge takes the form of a thesaurus, and is often treated at the computer science level by means of ontologies. The technical basis is often treated at the computer science level by means of architecture patterns.

• Applications of a substantial size are based on multiple business domains, rather than on a single domain. Each of these domains has its vocabulary which it is desirable to respect. An information system of a large scope shows all the characteristics of a cooperative information system. The extent of complex domains makes it often difficult to use classical tools (ontology, thesaurus) because scaling up of these tools is not guaranteed.

• Most of the time end-users expect new services from a new information system (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. Their main example is that of an installation of a system of component follow up on a complex assembly line. They weigh up the need of component localization against the cost (in the term of modifications of business process) of a precise localization.

• 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 highly specialized tools whose handling by novices is difficult (e.g., setting up 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 tend to possess different skill levels. 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 related to a complex domain:

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

• 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., alignment, 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 generate a formal description. Therefore, teams of knowledge modeling produce 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 relations. Section 3 explores challenges related 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 presents our ongoing work.

2. METAMODELS, ONTOLOGIES, AND CATEGORIZATION

We believe that the hard core of knowledge of a complex domain can be reduced to a pair of a metamodel and an ontology. The ontology 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 these two description languages enables to work with reasonable reliability by integrating necessary metamodels “on the fly”. This allows to obtain a beta version of the domain description at low cost (in terms of time and money). The objective of this section is to justify our approach. The first part of this section consists in a comparative study of related work on ontologies and metamodels. The second part introduces the notion of early categorization. The third part presents an example application.

2.1 Related work

In this section we recall work on metamodels and ontologies.

Metamodels and their relations can be classified according to Mellor & al.’s definitions of model mapping [20] and to Frantz’s classification of model abstraction mechanisms [9]. Model refinement is used to produce detailed models while model abstraction is used to synthesize abstract descriptions; representation or migration change the form of a model in order to emphasize specific properties; combination (or weaving) makes it possible to conjointly use several independent descriptions.

The joint use of two metamodels can be classified into three cases. In the first case, both metamodels describe the same domain, each metamodel representing a different point of view. This is the case of metamodel representation or migration (e.g., [6]). In the second case, both metamodels describe the same domain with each metamodel defining a different level of abstraction. This is obviously the case of metamodel refinement and abstraction. Metamodel combination and weaving which carry out abstraction on the scope of models (e.g., PIM and PSM of MDA) also fall under this case. In the third case, metamodels describe two different domains at the same level of abstraction. Metamodel integration as the joint use of such metamodels is closely related to combination of domain ontologies [10, 17, 22].

Ontologies have been defined in various ways [24, 28]. Just as metamodels, ontologies have a scope (i.e., domain of validity), e.g., general ontologies [19, 21] or domain ontologies [8]. Ontologies are most often considered as extensible [15], alignable or mergeable [17, 29]. In order to facilitate corresponding operations on ontologies, a certain number of top level categories have been defined. These categories define the outline of ontology description languages (concepts and relations between concepts). Sowa [28] offers a set of ten top level categories, Welty & al. [33] propose a classification of relations, Fürst [10] proposes a synthesis of recent classification work. In the following, we use the Sowa's definition of ontologies which emphasizes ontology categories, domains of interest, an user perspectives:

“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”.

Comparative study of metamodels and ontologies Many authors have discussed the proper role of metamodels and ontologies in modeling and engineering [3, 11, 14, 25]. Three major issues arise when comparing the work on ontologies and that 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 such characteristics are proposed as a reference for alignment of languages. The main difference lies between the integration of top-level categories with the ontologies and the separation of levels in metamodeling architectures (meta-metamodeling, meta-modeling, and modeling concepts and constructs). Another difference is the relative recognition of MOF by the metamodeling community, while the community of ontologies still discusses top level categories.

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

In both cases, applicable operations depend on semantic and structural proximity of the described knowledge, as well as on the relative positioning of descriptions to be combined: whether at the same or different 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, such descriptions 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 common.

We finish our comparative study with conclusion that metamodels and ontologies present common “deep” characteristics. In the following section, we show that a joint use of metamodels and ontologies for description of complex domain can be based on such common characteristics.
2.2 Early Categorization

Metamodeling levels are defined in accordance with the model theory [30], see Figure 1. A metamodel is a formal system; models are interpretations of their metamodels. Then, at the instance level the model's categorization is made explicit. However, categorization is not accomplished. As stated by Bessiere & al. [1], it is difficult to argue about formal systems and their instantiation. Yet, categorization should be discussed since it is closely related to semantical issues. In object oriented modeling, class instances express categorization. Yet, during the modeling process instantiation of classes is not known. It is thus highly desirable to introduce categorization into the modeling process so that it can help in designing or instantiating the system. Furthermore, when we build a metamodel by one of the operations allowing joint use of metamodels (as described above) for complex domain description, an explicit correspondence between the metamodel and the described domain is not necessarily guaranteed. In such a case it is even more desirable to use categorization as a guarantee of semantical quality.

We propose to use ontologies for early categorization since real-life objects are in fact classified in ontologies. Such an early categorization is carried out by a semantical coupling of a metamodel with an ontology. As depicted in Figure 2, this allows to describe complex application domains at high abstraction levels:

- The metamodel expresses an abstract yet well-established basis of knowledge provided that we can guarantee that metamodels are aligned to world-wide recognized modeling concepts.
- The ontology provides an early categorization which can assist during the modeling process. As stated by Elvesæter [5]:
  - an ontology provides a kind of categorization: “The models at the various levels may be semantically annotated using ontologies which help to achieve mutual understanding on all levels”;
  - for most application domains, ontologies pre-exist: “In each of these business domains (supply chain management, collaborative eproduct development, e-procurement, portfolio management) we find domain-specific dictionaries, thesauri, nomenclatures,...”.

Three levels of coupling can be used in order to obtain full benefits from such early categorization, namely at the instance, model and metamodel levels. We have experimented with the first two ones in the IkoSem project (briefly presented in the following section). At the instance level, we use an ontology to evaluate data quality (e.g., detection of non-fitting attribute values which can be classified into genuine contradictions or mere imprecise values). At the model level, we use sub-domains of ontologies to define attribute domains. At the metamodel level, we now relate ontology top-level concepts with UML constructs.

2.3 An example: the IkoSem project

IkoSem aims to provide modelers with a framework for prototyping a database from a collection of images: classification or indexing based on syntactical or semantical features of images [2]. IkoSem's core component is a model of images which allows integrated descriptions of syntax and semantics of images. An image is decomposed into image parts which can be either zones or objects. A zone is syntactically defined (e.g., by a uniform RGB color <234,25,25>) while an object is semantically defined (e.g., an archaeological remain of a Roman camp). Since objects that compose an image can be very complex, image descriptions rely upon hierarchies of objects and zones. We call them composition relations. IkoSem's model of images encompasses spatial and semantical relations between parts of an image.

We have defined a metamodel for IkoSem (see Figure 3). The stereotyped class <<Part>> is a construct for modeling of image parts. Three different image parts are available; they can be distinguished from each other by an attribute called nature (with possible values image for the whole image, zone for a syntactically defined part, and object for a semantically defined part). The stereotyped class <<Attr>> is a construct for modeling of attributes which contain information (either syntactical or semantical) related to an image part. Image parts can be associated to attributes by the stereotyped association <<RelAttr>>. The stereotyped association <<Rel>> associates two image parts (with an attribute called type, having possible values semantical, spatial, and composition).
A generic model of an image database based on Ikosem metamodel is depicted in Figure 4. Image descriptions and image parts are described by <<Part>> stereotypes. Attributes (described as <<Attr>> stereotypes) are associated to image descriptions and image parts. An image description is related to image parts by a relation (named decomposed) which is a <<Rel>> stereotype whose attribute type has only one possible value, namely that of composition.

In order to build a model for an image database, we instantiate Ikosem's generic model. Such an instantiation consists in defining attributes associated to an image description and images parts. Attribute domains can be defined as parts of the ontology associated with the image database. Our example is an image database of archaeological sites in Burgundy. Each picture has been annotated: description sheets contain meta-data (e.g., precise locations and dates), as well as archaeological information. We have chosen to decompose an image into geometrical objects (since we are interested in components of buildings). As illustrated in Figure 5, our model has been instantiated by using ontology sub-categories as attribute ranges.

3. CHALLENGES INVOLVED

In order to describe domain knowledge with ontologies and metamodels, it is necessary to decide 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 information system 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, and to organize and automate testing and validation activities. This implies intensive work on models, extending to the metamodel level.

However, metamodeling approaches go considerably beyond the scope of OMG proposals, e.g., the Megamodel proposal [6]; ModelWare, GrammarWare, GraphWare; EAI (Enterprise Application Integration) and MIC (Integrated Computing) [18]. For Kühn & al. [18], a comprehensive metamodeling platform should contain components for persistence and access, as well as version control, multi-language support, analysis and simulation of models (in addition to standard components, i.e., meta-metamodel, base of metamodels and models, mechanisms of manipulation of models and metamodels). Thus, metamodeling environments pose multiple challenges:

- Abstraction levels of metamodeling architectures need to be discussed since components of metamodeling environments tend to cover several levels. Other proposals of abstraction levels have been made.
- 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 [4, 13, 23] emphasize the multiplication of languages in model transformations and set forth the need of convergence towards languages which would be widely used, accessible both by a graphical notation (e.g., [34,36]) and by a lexical notation (for scaling up), declarative, bidirectional.

1 In the European project ATHENA [35] the proposed levels are conceptual (concepts, metamodels, languages, models), technical (software development and execution environments), applicative (methodologies, domain standards and models).
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 inciting users to metamodel conversion [16]. Yet, platforms such as Adonis [34] or MetaEdit+ [36] have defined their own meta-metamodels. The Megamodel [6] proposal offers a new alternative, in which several reference meta-metamodels are recognized and accepted.

Interoperability is certainly one of crucial challenges of Model Driven approaches. Interoperability seems however deciding at two levels in metamodeling environments. On the one hand, there are risks of semantic heterogeneity when non-aligned meta-metamodels are used, or when model transformations do not clearly reference a semantical basis (shared metamodel or meta-metamodel), or when operations on metamodels do not have a clearly defined semantics. 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 own metamodels, specific technologies of representation of models, non-standardized access mechanisms.

4. POSITIONING OF OUR PLATFORM PROPOSAL

The positioning of our proposal in comparison to the above challenges is as follows. Our first objective is to preserve abstraction levels of OMG while emphasizing the role of the metamodel level in the description of application domains, yet introducing a boundary between reuse and contextualization at the model level. An application domain is described by a specific metamodel. A generic model referring to this metamodel describes common features of a family of applications (e.g., image databases, e-learning platforms). Each model of a specific application in one family is instantiated from its generic model.

Our second objective is to guarantee preservation of constraints in the instantiated model. Most constraints are expressed at the metamodel level (e.g., “Zones are <<Part>> with no semantical attribute” in IkoSem metamodel). Some constraints are expressed at the model level (e.g., “the decomposed relation (between an image description and image parts) must have the composition tag value” in IkoSem generic model).

Our third objective is to treat interoperability at the metamodel and model levels: the metamodel level is used to determine spots
of semantic variations (identified as “high-risk zone” for interoperable applications).

Finally, at the non-functional level, we propose a platform schema consisting of technical specifications based on Open Formats and Open Source tools [27]. Based on an experiment, carried out within the IkoSem project, we made the following proposal for construction of platforms dedicated to information system engineering in the given domain [31]:

- Review of functionalities necessary to users and study of available knowledge. As far as knowledge is concerned, we assume that this preliminary study should result in the definition of an organization of terms of its domain in an ontology. If no domain ontology is available, a thesaurus can be used as a simplified form of a terminological ontology. We assume that the metamodel chosen for the platform is consistent with links between high-level categories of the ontology.

- Definition of a generic architecture, based on the domain-specific metamodel and ontology, and of a generic model. Such a generic architecture should allow instantiation in order to constitute an architecture of information systems to be built. This implies three types of control mechanisms:
  - 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 themselves in order to render the instantiated system operational. It is necessary to guarantee that such tools conform to the domain semantics. The metamodel and ontology allow to guarantee such a semantical consistency.
  - Instantiation should not cause loss of constraints that have been defined at the metamodel level and the generic model level. This last issue is one of the objectives of our study in the short term.

A complex domain is generally made up of several simple domains whose knowledge bases are partially overlapping. We propose to build a platform for such a complex domain by using a double integrator (a metamodel integrator based on the MetaSem [32] architecture, and an ontology integrator). These two integrators should allow to determine the knowledge basis associated with the complex domain platform. Our proposal has the following characteristics:

- The simple domains are described in the context of a metamodel constructed from a UML metamodel. Such metamodels are introduced into our MetaSem architecture. It is necessary to have available an ontology (or at least a thesaurus) describing the vocabulary that forms the core business of each of the simple domains.

- The MetaSem architecture allows to build (by integrating appropriate metamodels) the reference formal system of the complex domain. The integration of ontologies describes the corresponding categorization. It is thus essential that the integration of ontologies and metamodels is conducted in a consistent way. Several strategies are in general conceivable (with control of choices given either to the metamodel or to the ontology). It 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 thus form spots of semantical variations between the simple systems and the complex system.

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

- Other tests can be conducted directly at the metamodel and model levels (if constraints have been explicitly introduced 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 challenge is to be able to introduce various verifications in model transformations. Some examples could be: non-application of transformation rules on the spots of semantical variations without validation by a domain expert, conformity of transformation rules with metamodels, consistency of transformation rules with the metamodel-ontology coupling.

5. CONCLUSION

Our objective in the short term is to develop several domain platforms in order to be able to define several strategies in using the platform (according to the desired semantical consistency). One of the important challenges is 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


