Ontology-based Layered Semantics for Precise OA&D Modeling

Jean Bézivin[1], Richard Lemesle[1,2]

[1]Laboratoire de Recherche en Sciences de Gestion, Université de Nantes,
2 rue de la Houssinière BP 92208 Nantes cedex 03, France
email : jbezivin@acm.org
[2]Société Soft-Maint, 4 rue du château de l'Eraudière BP 588 44074 Nantes cedex 03
email : lemesle@iae.univ-nantes.fr

Abstract. OSMOSIS [Bézivin1995] is a research platform intended to investigate the various forms of products and processes in object-oriented software production. The kernel of this system is made of a minimal representation support called sNets, a typed, reflective and modular kind of semantic network. Each model represented in this network is composed of a number of typed entities (nodes) and relations between these entities (links) i.e. each model is a partition in the sNet called a *universe*. For each such universe, there is another one called its *semantic universe* defining the corresponding *ontology*. In short, an ontology specifies the concepts that may be used and the possible relations between these concepts. Our kernel sNet notation may be qualified of a NOON (Non Object-Oriented Notation) because the concepts of class or object are not built-in in our system. One reason for this choice is to cater to many different semantics for classes, objects and instanceOf/isA relations. We stress here some of the consequences of these choices on the architecture of meta-levels and show the strong relation between this architecture and the precise definition of the instanciation relations in different contexts. Our illustration will be based on CDIF [Ernst1997].

1. Introduction

The arriving to industrial maturity of object technologies is opening a new period that should see narrowing the differences of concerns between the software engineering and the knowledge representation communities. The announced arrival of the Unified Modeling Language UML [Booch1995] by OMG strengthens this feeling. This compromise between standardization and openness may only be expressed within a framework where models and ontologies are considered as first-class concepts. The likely choice of CDIF in the OMG efforts, as well as the layered architecture of metamodels, clearly opens the debate on the key contribution of modeling and knowledge representation techniques in this new deployment period of object technology. This paper presents some of the practical problems resulting from the multiplicity of models and ontologies in the current forms of software systems development. More particularly it addresses the important concern of model semantic interoperability and shows how an adequate architecture of meta-levels may allow defining with precision the different kinds of instantiation relations that may be found in a practical system.

2. Meta-modeling : how many Layers?

A classical problem is the number of layers used in meta-modeling. The first layer, always present and usually known as the meta-meta-model, is the base of this layered architecture. The second one represents meta-models and the third layer always represents models. At this point, you can either consider the architecture complete, or you can also add another layer to describe data. Some proposals use an unlimited number of layers to represent these data. But one has to keep in mind that the number of layers depends on the instantiation relation used between entities from each of these levels (or layers). In order to explain how these different layers are identified, let’s first examine the example of a classical four layers architecture.

3. A four Layer Classical Architecture

This is the architecture used by CDIF. The four layers are presented below:

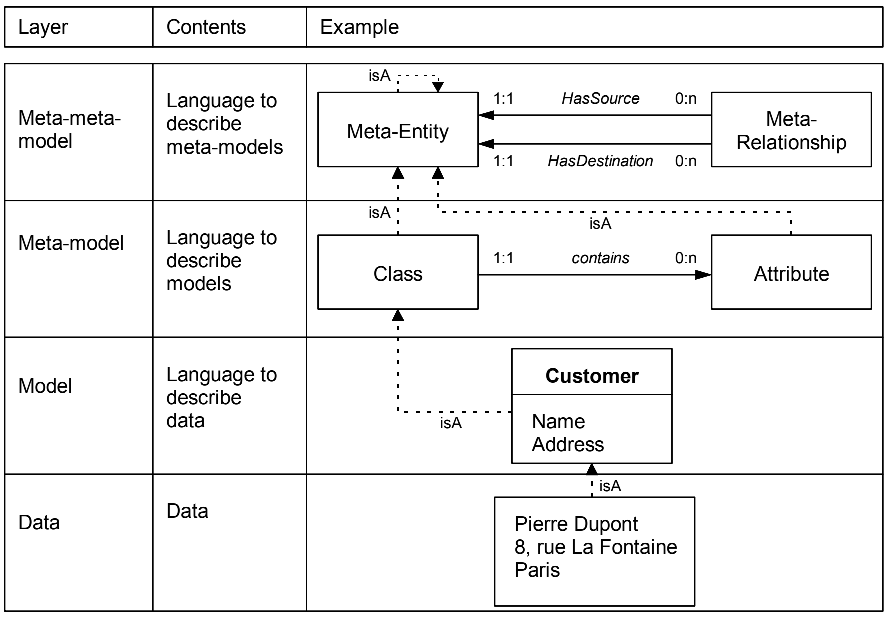

Fig. 1. Four layer classical architecture

Layered architecture is associated to a given instantiation relationship [Odell1995] (called *«isA»* there). Accordingly, we have the following predicates:

- A layer contains entities.
- Stating that an entity belongs to a layer means that this entity has an instantiation link (isA) to an entity of the previous layer.

So, there should be a precise and unique definition of this instantiation relationship in order to find out which layer an entity belongs to. The previous figure seems to be consistent with these predicates. But do all the *isA* links have the same meta-definition? Attempting to represent these meta-definitions, the following scheme is obtained:

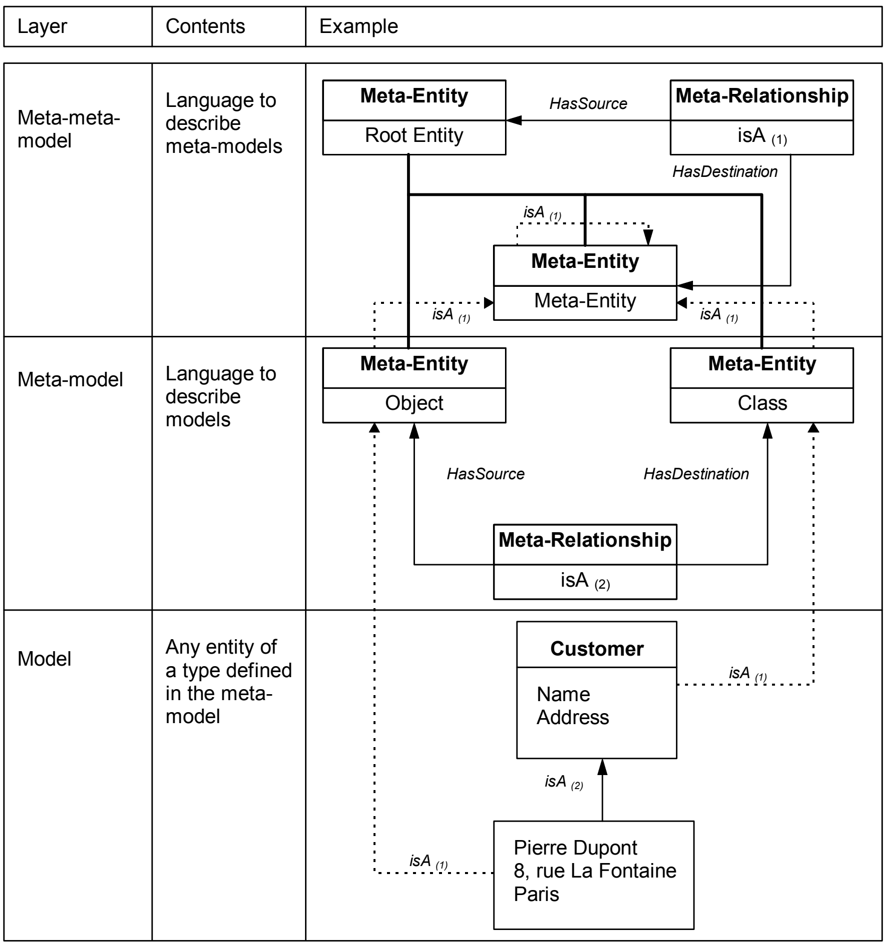

Fig. 2. Three-layer architecture with instantiation meta-relationship represented.

In this figure, bold lines represent an inheritance relation used to make the $isA_{(1)}$ meta-definition unique. This figure shows that there are two different instantiation relations. The first one, used to identify meta-modeling layers, is called $isA_{(1)}$. The second one, called $isA_{(2)}$, should be defined in a meta-model that corresponds to a specific given object space, based on a specific definition of classes. Such a space defines an instantiation relation between objects and classes, but this relation is totally different from the $isA_{(1)}$ relation. Obviously there are several of these $isA_{(2)}$ relations, as there are several notions of classes in different definition spaces (semantic universes).

4. Conclusion

There is often some confusion about the architecture of meta-levels in current proposals. We have suggested here that the number of layers, in meta-modeling layered architectures, is closely linked to a basic instantiation relationship. This relationship must be unique in order to clearly separate each layer. And it must be precisely defined by a single meta-relationship, which belongs to the first layer. Consequently, having that property, the number of layers that can be defined is three. Other layers may be included as parts of the third one.

As part of this third layer, we may find different specialized definitions of the notions of class, instance and of the various customized *isA* relations between these concepts. As a consequence we can work on a precise framework where the different models are well separated and well defined. This gives us a way to achieve preciseness as proposed by [Kilov1994] and to deal with the huge number of different semantics usually found in object-oriented or in non object-oriented systems [Lamb1996]. The sNet kernel, in the OSMOSIS project, offers a general framework where all these notions may be defined, applied and analyzed.

References

[Bézivin1995] Bézivin, J. **Object-Oriented Requirement Elicitation in the OSMOSIS Project** IEEE International Symposium and Workshop on Engineering of Computer-Based Systems, Tucson, Arizona, (6-9 March 1995), p. 289-298.

[Booch1995] Booch, G. & Rumbaugh, J. **Unified Method for Object-Oriented Development** Documentation Set Version 0.8, Rational Software Corporation, (Oct. 1995).

[Ernst1997] Ernst, J. **Introduction to CDIF**, Integrated Systems, Inc., (Jan. 1997).

[Kilov1994] Kilov, H. James Ross. **Information Modeling: an Object-oriented Approach**. Prentice-Hall, Englewood Cliffs, NJ, (1994).

[Lamb1996] Lamb, D.A. Editor, **Studies of Software Design**, Lecture Notes in Computer Science, Volume 1078, Springer Verlag, (1996).

[Odell1995] Odell, J. **Meta-modeling**, OOPSLA'95 Workshop on Metamodeling in OO, (October 15, 1995).