

Foundations of Meta-Pyramids: Languages vs. Metamodels

Episode II: Story of Thotus the Baboon¹

Jean-Marie Favre

ADELE Team, Laboratoire LSR-IMAG
University of Grenoble, France²

Abstract. Despite the recent interest for Model Driven Engineering approaches, the so-called four-layers metamodelling architecture is subject to a lot of debate. The relationship that exists between a model and a metamodel is often called *instanceOf*, but this terminology, which comes directly from the object oriented technology, is not appropriate for the modelling of similar meta-pyramids in other domains. The goal of this paper is to study which are the foundations of the meta-pyramids independently from a particular technology. This paper is actually the second episode of the series "From Ancient Egypt to Model Driven Engineering". In the pilot episode, the notion of megamodel was introduced to model essential Model Driven Engineering concepts. The notion of models was thoroughly discussed and only one association, namely *RepresentationOf* was introduced. In this paper the megamodel is extended with one fundamental relation in order to model the notion of languages and of metamodels. It is shown how Thotus the Baboon helped Nivizeb the priest in designing strong foundations for meta-pyramids. The secrets of some ancient pyramids are revealed.

1 Introduction

Model-Driven Engineering (MDE) is becoming increasingly popular, or at least is drawing more and more attention, both in the research and the industrial communities. The notion of model is central to this approach. It makes no doubt that standard such as UML has played a significant role in the diffusion of the concept of model in industry. At the end of 2000, the OMG launched a new standard coined Model Driven Architecture (MDA) [2]. In fact, MDA provides a grand vision for the future of software development. MDA is a set of industrial standards rather than a single standard [3].



1. Episode II of the series "From Ancient Egypt to Model Driven Engineering" [1]
2. <http://www-adele.imag.fr/~jmfavre>

This corresponds to a shift from strategy at the OMG. The old Object Management Architecture (OMA) has been replaced by the new Model Driven Architecture (MDA). Simply put this corresponds to a change of motto: "*Everything is an Object*" becomes "*Everything is a Model*" [4]. Obviously, these slogans are provocative on purpose. They should be interpreted as an approach in which everything is considered a priori to be an object or model, to see what this could mean and study what happen if it is not actually the case. The goal is just to push the paradigm to its limits and to discover these limits.

This paper is the second episode of the series entitled "From Ancient Egypt to Model Driven Engineering". In the previous episode the concept of model was thoroughly studied [5]. There has been a strong debate about Model Driven Engineering, and though enthusiasts see MDE as a revolutionary paradigm, they are also many sceptics. The 4-layers meta-pyramid from the OMG (see Figure 7) is subject to sarcastic remark: "*So, MDE is just about Egyptology?!*". When the idea behind MDA are presented they continue "*There is nothing new in MDE. All these concepts had existed for ages*". This series goes in the same direction because all this remarks are in part true. But as shown in the pilot episode, this is what make the strength of the MDE approach [5].

It makes no doubt that model had existed for long. For instance in Ancient Egypt, masks of pharaohs where used as models of the Pharos, so that gods could knew who had been mummified, what had been its social status, and so on. Conversely, statues of gods were models for the usage of humans, so they can learn what was the god good for, how bad it was, etc. The following figure shows four objects from Ancient Egypt. The formalism used is UML, except that photos are integrated for readability purposes. All the objects presented played at their time the role of model. And they still play this role, though some naive visitor may consider them as simple objects from the past.

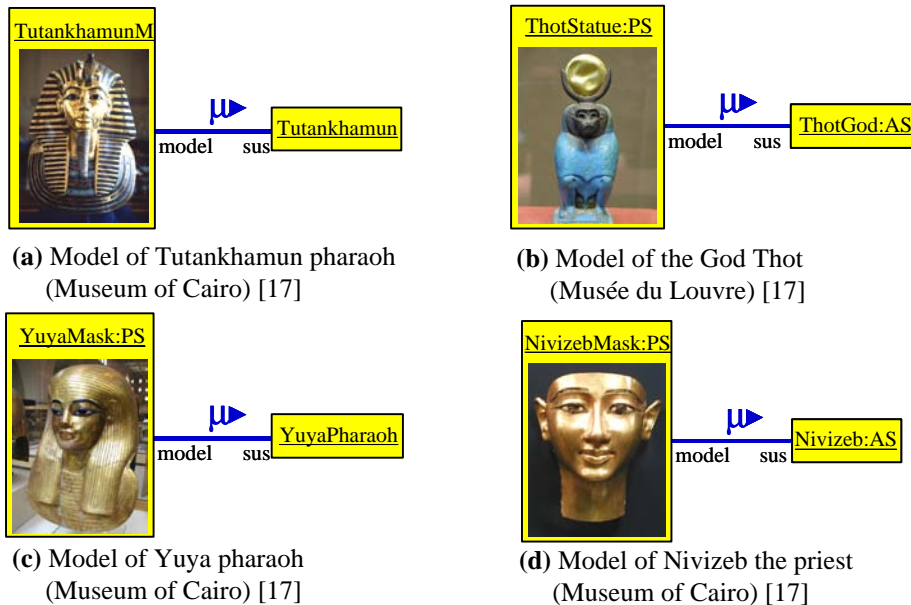



Figure 1 Models from Ancient Egypt

Various definitions of "model" were studied in the first episode [5], but summing up, a *model* is a *system* that enable to give answers about a *system under study* without the need to consider directly this system under study (SUS). A model is not expected to be perfect or to say everything about the SUS. It is simply considered as "good enough" for a given purpose.

The systems described above matches perfectly this description. The story of Antonio and Tom visiting the Louvre museum also illustrates this aspect. Only Tom and Antonio know what they did that day. If you want to know you have to ask them; or, and this is a certainly a better alternative, you might consider the two models of the story provided in Figure 2 and in Figure 3. Though these models are expressed in two different languages they model the same story but from a slightly different perspective.



Tom and Antonio at the Louvre Museum

Tom wanted to know more about the god Thot because he read somewhere that Thot was who gave Meta-modelling to mankind. Tom could directly ask Thot. This was what Nivizeb the priest would do in Ancient Egypt [5] (see Figure 3 on the right). But since getting answers directly from gods was not easy for Tom, he decided to meet Antonio, and they went together to the Louvre Museum. Museums are large model repositories [5]. Tom and Antonio knew that the best way to get answers about Thot was to look at some of its models displayed in the museum. They finally arrived in front of a statue representing Thot. Antonio was familiar with Egyptology [5]. He knew much about the meaning of egyptian antiquities, so he directly looked at the statue (centre of Figure 3). But Tom was a novice in Egyptology. So just like most other visitors he read the notice near the statue (left of Figure 3). The notice was actually a model of the statue: instead of asking questions directly to the statue, visitors could read the notice. Tom could read French, and as a MDE expert he directly noticed that the top part of the notice board was a model of the statue that gave him information about Thot. From the bottom part he can get answers about the statue but not about what it modelled. Tom was not surprised because he knew that the RepresentationOf relation was not necessarily transitive [5]. Anyway he took a photography of the statue with its digital camera (he made a digital model of a physical model). Back to Belgium he print the photography, put it on a nice frame and display it above of his bed, as a souvenir from the Louvre Museum. A few days later, Tom visited www.louvre.fr. His favorite web browset got a digital model of the statue and displayed an image of it on the screen".

Figure 2 A Model of the story of Tom and Antonio at the Louvre Museum

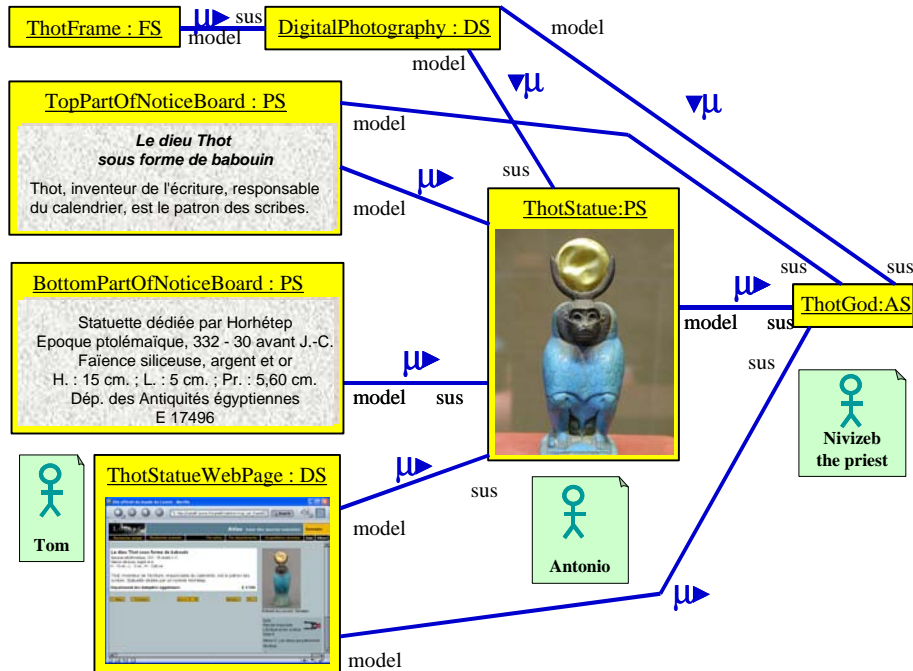



Figure 3 Another Model of the story of Tom and Antonio at the Louvre (μ -graph)

The models above show that the world is full of complex nets of systems, models, models of models, models of models of models, and so on. Figure 3 shows this aspect by representing a μ -graph, that is a graph of systems connected by μ links. Those links were noted  in Ancient Egyptian [12].

It makes no doubt that the content of Figure 2 is certainly easier to understand than the content of Figure 3, at least for those readers who unfortunately missed the pilot of the series "From Ancient Egypt to Model Driven Engineering". For instance such a reader might not guess that μ links read "Represents", and that the RepresentationOf relation is fundamental to MDE. This relation and its properties were thoroughly studied in the first episode of this series [5]. For the purpose of this paper, it should be enough for the reader to consider the model presented below. This model summarizes the core part of episode I. Many of the questions that were answered in [5] can be answered by this model. At least for the reader that understand the language used in Figure 4

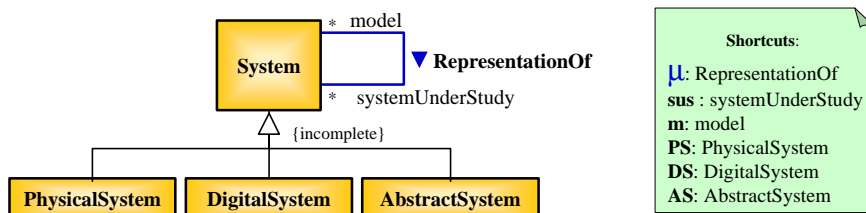






Figure 4 The μ -MegaModel (Episode I, [5])

There is a strong issue there. If one want to understand a model one has to know in which *language* this model is expressed. So let's provide the information. Figure 2 is expressed in English. Figure 3 is on the one hand expressed in UML and in the other hand expressed in terms of Figure 4. Finally Figure 4 is written in UML. That's look quite complicated, at least for what concern Figure 3.

But in fact, things get even worse. Mentioning the language does help only those readers who at least a mental model of that language. In other words, if one want to understand Figure 2 it is not only necessary to know that this model is written in English. A model of the English language is also required, either to check if the model is conformant to the English grammar or to understand a new English word. But what is then *a model of a language*?

According to the definition of model given previously, this is just a system that could provide answers about the language under study. For instance the english dictionary and the grammar book of the author are physical models of the English language: these model provide answers to the author when he has questions about the English language. Similarly, those readers that do not have a mental model of the UML language in mind are invited to take a model of the UML language, for instance a UML book, or the UML specification document, or the UML meta-model. So a UML meta-model is a model of the UML language. Note that conversely UML is a language of models. But languages of models should not be confused with *models of languages*, and *models of languages of models*, which are actually *metamodels*. But in which language the UML metamodel is defined? According to the MDA standard all metamodels must be written in the MOF language to be MDA compliant. But what if the reader don't have in mind a good model of the MOF language. Then a solution would be to read the MOF specification document which is written in English. Another solution is to study the MOF metamodel. In which language is this metamodel written? Attentive readers will answer "in the MOF language", because as said before each metamodel had to be written in the MOF language to be MDA compliant. Getting confused or lost?

Let summarizes this discussion by modelling its content in the form of an UML instance diagram. Hopefully the model provided in Figure 5 will give answers to the reader who had unanswered questions. As the reader can see only two kinds of links have been used in this model: μ links and ϵ links (respectively  and  [12]). The overall structure of the model is however not apparent. Only attentive readers will note repetitive occurrences of patterns such as the $\mu\epsilon\mu$ pattern, referred as a *meta-step* in section 4.1. In fact the model in Figure 5 makes it possible to answer many questions. For instance one can get "backward" slices of this graph by computing transitive closure of μ links in the reverse direction () and of ϵ links in the forward direction (). The resulting set of systems indicates which systems can be considered to understand the source of the slice. For instance, if Nivizeb the priest want to understand the StoryOfTom&Antonio he would have to understand all systems depicted in $\mu\epsilon$ -graph (Figure 5). If he want to understand what Figure 3 said about this story, then he would have to understand both the MegaModel and the UMLLanguage. Then he could choose to study either Figure 4 or the firstEpisode. If he choose to study Figure 4 he had to learn the UMLLanguage, the MOFLanguage and eventually the EnglishLanguage.

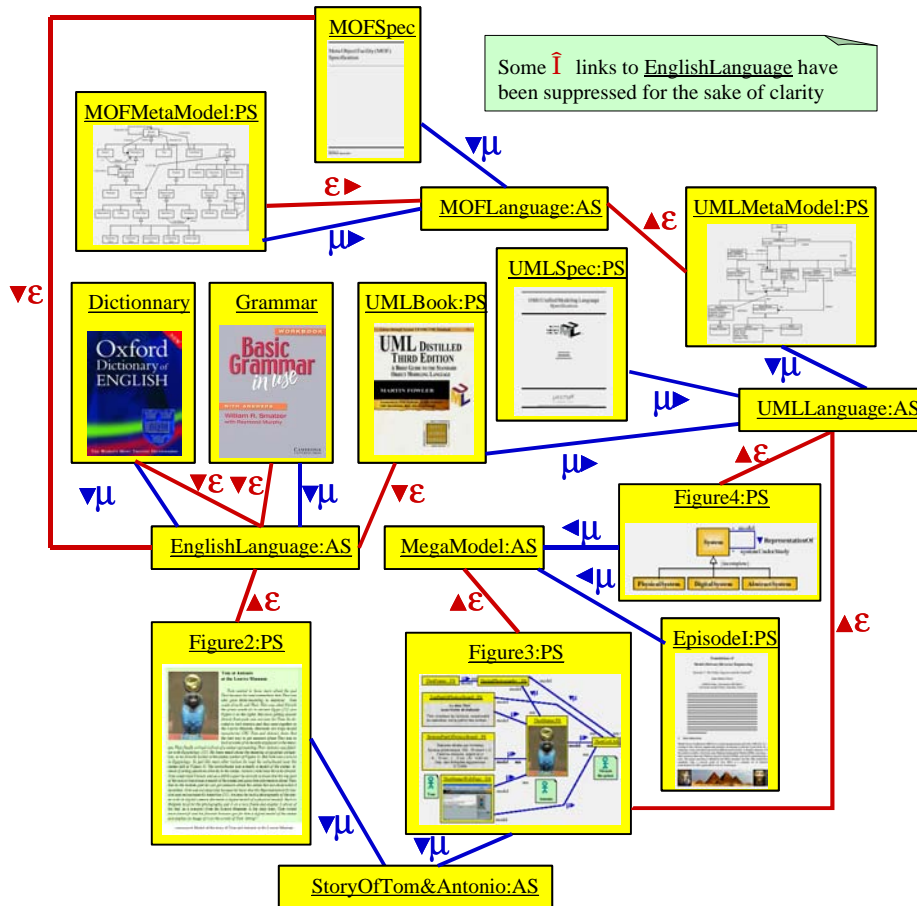


Figure 5 A model of the discussion on the previous page ($\mu\epsilon$ -graph).

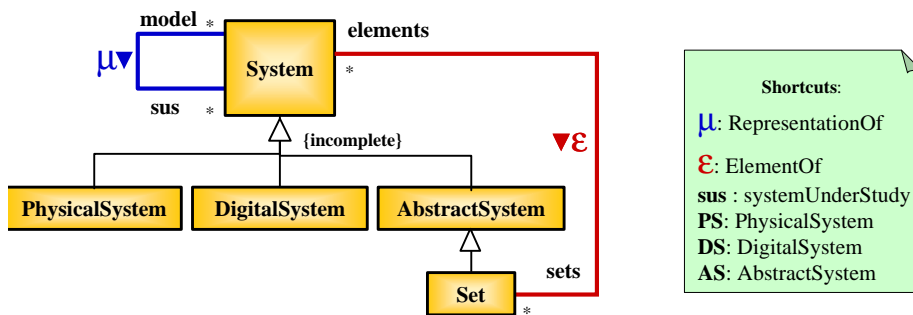


Figure 6 $\mu\epsilon$ -MegaModel

Unfortunately, Nivizeb will be very disappointed because none of the languages in the backward slice are connected to Hieroglyphics, Copt or Greek, ones of the numerous languages he can read [12]. So Nivizeb would not understand anything at all! That's a pity, because at his time Nivizeb was a precursor in Model Driven Engineering [5]. This story raises a serious question: from where the top-level language come from? According to ancient egyptians, the god Thot is who gave language to humans.

The first goal of this paper is to make models such as the one presented in Figure 5 easy to understand. Since the μ relation was introduced in Episode I, this Episode is devoted of the ϵ relation. It is shown how this concept that directly come from the set theory is connected to the notion of *language*, *modelling language*, and *metamodels*. Following the approach that characterize this series, the MegaModel presented in the previous episode is augmented with the ϵ relation. So the so-called μ -MegaModel that modelled Episode I (Figure 4), is replaced by the $\mu\epsilon$ -MegaModel which models this second episode. Figure 6 presents a partial model of the $\mu\epsilon$ -MegaModel. This model is sufficient to describe the information contained in Figure 5.

The second goal of this paper is to provide a framework that help in understanding the global structure of complex $\mu\epsilon$ graphs. This is where the concept of architecture comes. The 4-layers meta-pyramid from the OMG defines how the various standards provided by this organization fit together. As said before this "architecture" is subject to a lot of debate. This paper tries to clarify the relationships between the elements of this pyramid. This is not so easy. For instance, the reader is invited to try to guess how the various elements of Figure 5 could fit in the meta-pyramid.

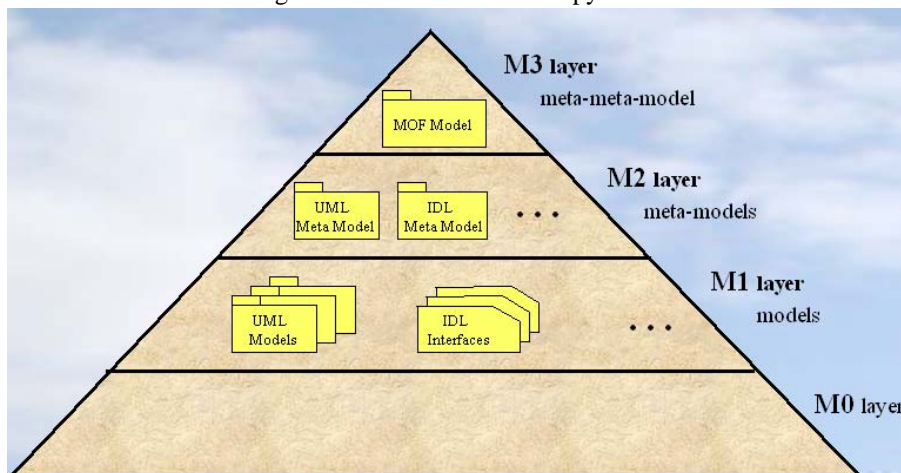


Figure 7 The four-layers metamodeling pyramid (OMG Dynasty, 2000 A.D.)

Note that OMG meta-pyramid is not strictly speaking an innovation. The first stone building on Earth is the Saqqara "step" pyramid (Figure 8.a). It was built more that four thousand years before the OMG pyramid. One of the last exemplars of pyramid can be found in the court of the Museum of the Louvre (Figure 8.b). Just like the OMG meta-pyramid, this pyramid has been subject of much debate. This pyramid makes a symbolic link between the future and the past. It will be shown in section 5 how the evolution of pyramids is connected with the evolution of Model Driven Engineering.

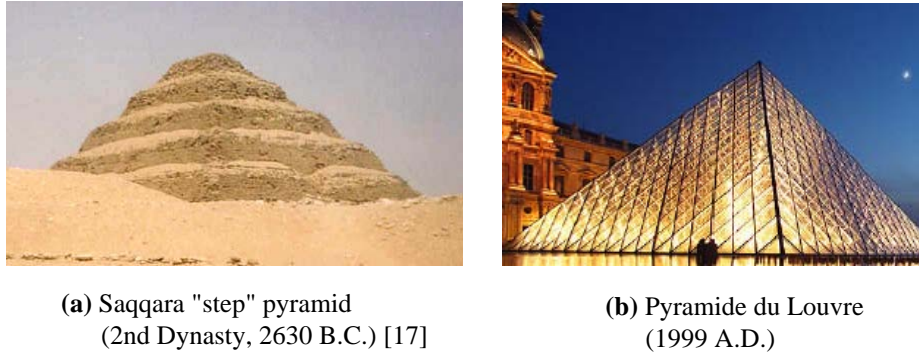


Figure 8 Evolution of pyramid architectures

The remainder of this paper is structured as following. In section 2 the concept of sets and the concept of language are introduced. The first concept comes from the set theory, the second comes from the language theory. Then section 3 shows that the combination of these concepts with the notion of model, leads both to modelling languages and to meta-models. In section 4 the concept of meta-pyramid is discussed. In section 5 it is shown how Nivizeb the priest designed, with the help of Thotus the Baboon, a series of meta-pyramids. Finally section 6 concludes this episode.

Note the description of Model Driven Engineering continues in the next episodes. For instance, while the notion of metamodel is defined in this episode, no information is provided about the methods that lead to the production of metamodels. The foundations of Metamodel (Driven) (Reverse) Engineering will be discussed in Episode III. Similarly, the notion of meta-model presented in this paper is the weakest one. The discussion of the notions of syntax and semantics, which are natural extensions to the concept of language, and therefore to metamodels, are deliberately postponed to further episodes. This episode deals with the truly minimal set of concepts to introduce metamodels. No more. The lesser, the better.

2 Sets and Languages

In the previous episode, we learnt the power of the RepresentationOf relation μ . It was shown how this relation was extensively used in Model Driven Engineering. In fact, μ was used alone for ages. Each system was produced and studied separately, by single individuals. By then, the μ relation was enough. Nevertheless, this situation later changed with the increase in complexity.

On the one hand, with the increase of the number of systems to be produced or to be studied, it became more and more important to consider commonalties between these systems. What was needed then was *a mechanism to think about systems collectively*.

On the other hand, with the increase of the number of people that had to reason about larger and larger systems, the need for rigor was also increasing. What was needed in that case was *a mechanism to formalize and exchange models*.

The remainder of this paper is devoted to the concept of metamodel which fulfil these needs. It should be clear however that the concept of metamodel is *not* a prerequisite to speak about models. Plenty of models on this planet were produced without metamodels or at least without making explicit metamodels.

The situation has changed however. Using explicit metamodels is now considered as a strong requirement in modern model driven engineering. In particular, explicit metamodels is a prerequisite in MDA. This is because emphasis is on automatization, and because automatization means explicit meta-models. In Episode I, various definitions of model that were cited [5]. None of these definitions make reference to the notion of language or meta-model. By contrast, Kleppe, Warmer, and Bast give in [18] a more restrictive definition of what is a model.

"A model is a description of (part of) a system written in a well-defined language" [18]

Note that this definition is given in the context of the MDA. The concept of model in this context is strongly connected to the notion of language and therefore to the notion of metamodel. These two concepts are often confused. So the rest of this section clarifies the relationships between the concepts of set, language, modelling language and meta-model.

2.1 Sets and "ElementOf" (\mathcal{E} , 𐀓)

Dealing with more and more systems or models leads to the abstract concept of *set*. The term set refers here to the mathematical concept of the set theory. Nothing more. The relation between the elements of a set and the set will be called *ElementOf* in our MegaModel. This relation is noted \mathcal{E} in Greek. It was noted 𐀓 in Ancient Egypt [12]. This association corresponds to the \in mathematical symbol. Nothing more¹. As shown in the class diagram below a set is an abstract system. A set *does not* have any materialization per se.

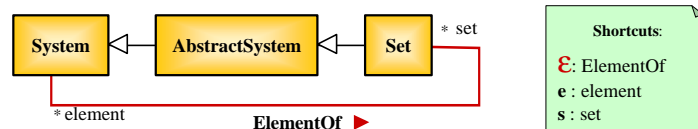


Figure 9 MegaModel: ElementOf (\mathcal{E} , 𐀓)

Nivizeb produced the Fidus Papyrus when he was young [5]. At that time he just wanted to model a single system, the solar system. At that time, he had felt no need for generality. However, as he produced and studied more and more systems, he started to discover the importance of structuring his knowledge. Years after years, he had observed so many planets, that he found convenient to think in terms of "sets" of planets with similar properties. These sets were just abstractions produced by his imagination, because obviously he could not gather together the planets as he wanted. This was far

1. Though the megamodel is here described in UML, we are also in the process of defining other models of it using other languages such as Z, Prolog and Hieroglyphics. We use the symbol \mathcal{E} instead of \in to avoid confusion with the metalanguage in the Z version and to ensure that the megamodel is independent from a particular language.

beyond the power of this great priest. Nivizeb defined however two sets: "dark planets" and "god planets" (Figure 10.a). Nobody understood how this classification was done. Nobody dared to asked. Nivizeb was a great priest. And his crocodiles were very bad. Anyway, Nivizeb was able, given any planet, to say if it was a dark or a god planet.

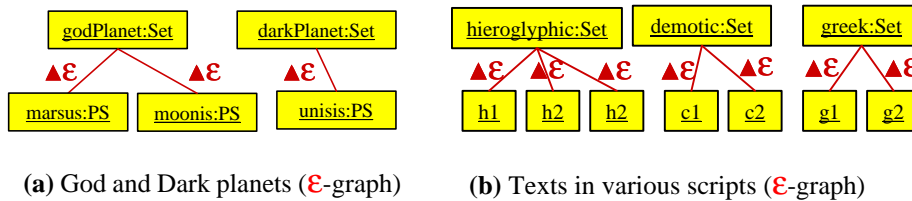


Figure 10 Example of uses of Sets by Nivizeb

Nivizeb was brilliant. At the age of 20, he could speak many languages and knew many scripts. During his life he collected a lot of documents. Though his workshop was always a real mess, he mentally classified each document using "sets" (Figure 10.b). Nivizeb knew that each set was virtually infinite because one could always write new texts. Anyway, each time he received a document he knew in which set it pertained. This was enough to characterize the set.

In fact, over the years the great priest discovered, thanks to his sacred animal, that sets have interesting properties. Nivizeb had indeed a lot of sacred animals. Much of them were spending most of their time wandering around the Nivizeb' temple. In particular, Nivizeb had a baboon. He had called him Thotus after the god Thot, in some sort of kind mockery (Nivizeb was a great priest, but also a joker). Thotus the Baboon escaped very often and climbed on the roof of the temple. During many years, Nivizeb didn't care about what Thotus was doing on the roof. He preferred to observe his crocodile scaring foreigners that dare coming too close to the temple. One day, however he was so upset by the disorder of his animals that he tried to establish a very structured classification of sacred animals, a classification in which each animal would stay quiet at a stable position. Obviously he was just building a mental model with his bright imagination. That day Nivizeb discovered that sets could be "included" in other sets (ζ in the Megamodel, includes in mathematics), and that sets could themselves pertain to other sets (ϵ). As shown in the next figure, Nivizeb was very fund of pyramidal structure.

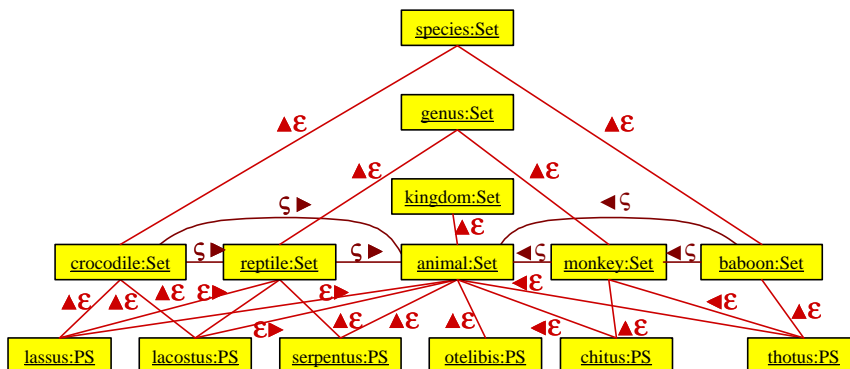


Figure 11 Nivizeb' mental classification of sacred animals ($\epsilon\zeta$ -graph)

Nivizeb discovered indeed the basics of the set theory. He used sets as abstract systems without any kind of materialization. He was totally unable to control all his animals. Nivizeb knew that some sets were infinite, but that they could be anyway characterized by means of a predicate, that is a criteria that indicated whether a given element pertained to the set or not. He established some interesting abstract properties on sets. In particular the "includedIn" (ζ) relation is transitive but the "ElementOf" (ϵ) relation is definitively not. The reader is invited to check that these properties are satisfied in the $\epsilon\zeta$ graph above (Figure 11). For instance lacostus is *not* a species, but crocodile is included in animal. Though not fundamental, the IncludedIn relation (ζ) can be added in the MegaModel as a derived association (Figure 12).

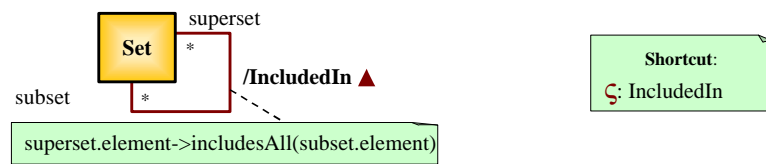


Figure 12 MegaModel: IncludedIn (ζ)

2.2 Classes and Languages

The examples above illustrate that the mathematical notion of set is used in practice for at least to purpose:

- to classify real-word entities and concepts,
- to reason about languages.

This distinction leads to two different fields of application for the set theory: the study of ontologies and the study of languages. Figure 10.a and Figure 11 fit in the first category, in which the various sets might be called *classes*. By contrast, sets in Figure 10.b might be called *languages*. Note that in fact there are only sets, though the usage of this concept might be different in different domains. The difference between ontological classification and linguistic classification is further discussed by Atkinson and Kühne in [9]. The reader is invited to compare Figure 11 of this paper and Figure 4 from [9].

If Nivizeb had considered all animals, not only his sacred animals he could have invented a biological classification. When he was very young, he was a strong supporter of what he called "object-orientation". At that time he used a strange terminology as he had invented a strange technology. With this technology, he modelled ϵ links using what he called "InstanceOf" relationships. Similarly ζ links were represented by "Inherits" relationships. But, this was just a particular technology.

Remember that this paper aims at establishing foundation of MDE independently from specific technologies. The set theory is good for that. Though ontological classification is important per se, in the remainder of this episode, attention will be focused on languages. The difference between ontologies and languages will be discussed in another episode of this series. The following definition will be "good enough" for this episode.

A language is a set of systems.

This definition is definitively very weak. But remember that the goal of this series is to define incrementally the foundations of MDE. Many readers may ask, what about syntax, what about semantics, and so on. These notions, that clearly make the richness of a language, could or could not be considered as integral parts of the language. The lesser, the better. More will be given later.

In fact this definition actually comes from the language theory, where a language is formalized as a set of sentences. The term set refers to the mathematical notion as described above. A language is indeed an abstract system itself. It does not have any materialization per se. For instance the english language is an abstract system that certainly exists, although it has no materialization. You can't gather all english documents on Earth. The same is true for the Copt, the language used in ancient Egyptian, but also for Greek, for the Java language, etc.

These sets are obviously infinite because we can always say something new. In fact, we know that the English language exists because we can say, given a sentence, if it is english or not. Here there is a serious issue. While the author of this paper thinks that the sentence he is currently writing is english, the reviewers might argue that it is not. What is needed is a practical means to check whether a particular sentence is english or not. The solution to this problem is to use some *models of the language*. Remember that models provide practical means to get answers. That's what is needed.

3 Modelling languages and Metamodels

The notion of a model of a language involve both the concept of set and the concept of model. In other words that means that it is necessary to combine the μ and ϵ relations. In episode I, μ -graphs were considered. So far this episode has introduced ϵ -graphs. So let's now consider $\mu\epsilon$ -graphs. It will be shown below that models of sets ($\mu\epsilon$) and sets of models ($\epsilon\mu$) do not represent the same concept, but that both concepts are necessary to define meta-models ($\mu\epsilon\mu$).

3.1 Models of sets ($\mu\epsilon$,)

The dictionary used by the author at the time of writing this paper is a physical system that models the english language (Figure 13.a). Similarly, the spelling checker used by the author is an (imperfect and really incomplete) software model of the english language. Remember that a model "*should be able to answer questions in place of the actual system*" [10]. This is exactly what the spelling checker does: given a word, it answers whether if it is english or not (Figure 13.c). Unfortunately, the spelling checker is a very partial model of the english language. A grammar checker would certainly help the author to improve this paper.

Note that nothing prevents a system to play at the same time the role of element of a language and the role of model of that language. This funny structure is in fact very common. One need to read English to understand what is written in the English dictionary, because this dictionary is itself written in English (Figure 13.c)! Egyptian solved this circular problem that by saying that the God Thot gave language to mankind. The same is true for Chinese.

The MDA standard solves this problem by saying that the MOF metamodel is both an element of the MOF language and a model of the MOF language [3].

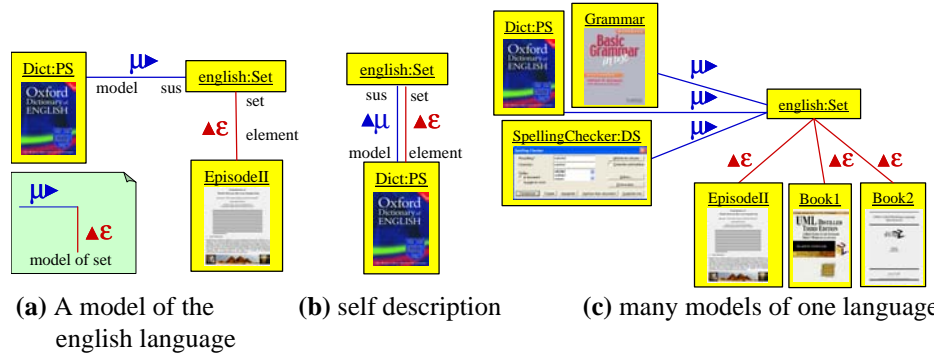


Figure 13 Models of a set ($\mu\epsilon$ pattern)

Emphasis should be put on the fact that making the difference between a language and the models describing this language is important. Episode III will show that this distinction plays a central role in the story of the Rosetta Stone [12]. It is also fundamental to understand what is meta-model (reverse) engineering [12]. This distinction is also well established in the language theory, in which languages and models of languages are not the same. Grammars are models of languages. A common mistake is to confuse the grammar with the language modelled by this grammar. But they are not the same. For instance the Java language is the set of all valid Java programs. Java is a abstract system which is infinite. A Java grammar is a finite system (e.g. a finite set of grammatical rules) that model the Java language by means of rules. Note also that the Java specification document is a physical model of the Java language, since it explains which sentences are valid. This is the same for a book on Java. Finally a Java parser is a software model that provides a practical means to determine whether a given program is an element of the java language or not. Sets should not be confused with models of sets.

3.2 Set of models ($\epsilon\mu$,)

What about the other way around, that is sets (or languages) of models?

A modelling language is a set of models.

In other words a modelling language is a set whose elements are models. UML is obviously a modelling language. Most of the figures presented in this paper are elements this modelling language. All UML models on Earth are elements of UML. Thought these models are modelling totally unrelated systems, they are elements of the same language. This is clear in Figure 14.

Note also that UML is a popular modelling language, but the most popular one is English, because one can also use English as an informal modelling language. Remember that Figure 2 and in Figure 3 were both model of the Tom&AntonioStory: Figure 2 was written in English, while Figure 3 was written in UML.

Obviously this is a very weak definition of a modelling language. But this is due to the weakness of the definition of language. The lesser, the better. More later.

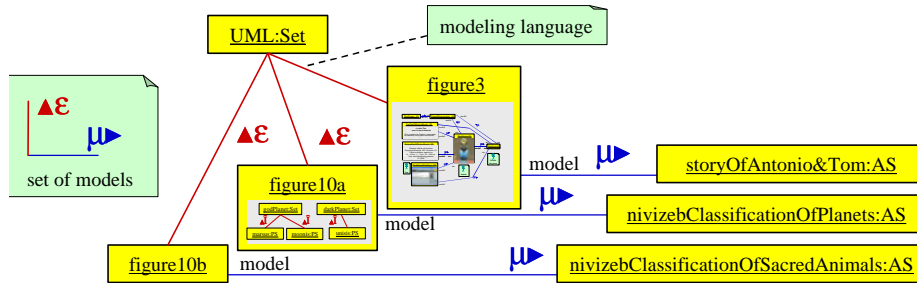


Figure 14 A set of models ($\epsilon\mu$ pattern)

3.3 Meta-models and models of sets of models ($\mu\epsilon\mu$,)

In fact, introducing the concept of modelling language makes it possible to give a precise definition of what is a metamodel.

A *metamodel* is a model of a modelling language.

This definition provides a criteria to decide whether a system can play the role of a metamodel or not. For instance Figure 15.a shows that the english dictionary of the author can play the role of metamodel because (1) this physical system models the english language (μ), (2) the author hope is that this paper is close to an english text (ϵ), (3) the content of this paper models the ideas of the author about MDE (μ).

Similarly the UML specification document is a physical system that can play the role of meta-model (Figure 15.b) because it models the UML modelling language, which is by definition the set of all UML expressions, which are in turn assumed to serve as models. As pointed out before, emphasis must be put on the fact that the notion of model and meta-model are roles, not intrinsic properties of systems.

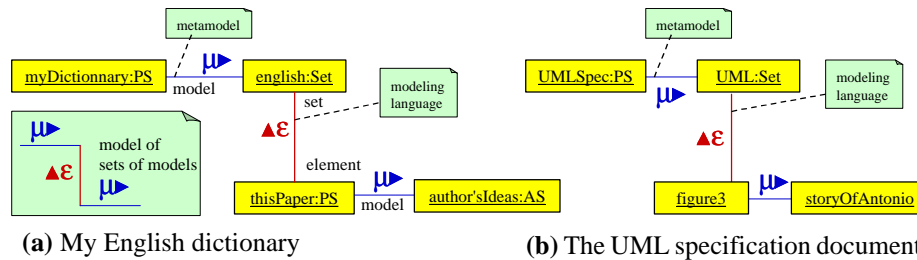


Figure 15 Metamodels ($\mu\epsilon\mu$ pattern)

Note that these concepts just result from the composition of the concepts presented in the two previous sections: models of sets ($\mu\epsilon$) and sets of models ($\epsilon\mu$). Flattening the previous definitions leads naturally to:

A *meta-model* is a model of a set of models.

The $\mu\epsilon\mu$ pattern, visible in Figure 15, will be referred as a meta-step in section 4.1.

It is amazing to consider that the systematic combination of the μ and ϵ leads to a definition that is in fact compatible with most of the definitions we found in the literature. For instance the MOF standard defines the notion of metamodel as following:

"A meta-model is a model that defines the language for expressing a model" [6]

We can compare also our definition to the one provided by Seidewitz.

"A metamodel is a specification model for a class of SUS where each SUS in the class is itself a valid model expressed in a certain modelling language" [8].

From our point of view, though valuable, this last definition has various flaws. First the term modelling language is not defined at all in [8]. Second, and this is much more important, defining a meta-model as a "specification model" is a mistake. Just like other models, a meta-model can be either "descriptive" or "prescriptive" (see Episode I [5]). Episode III will deal with this very important aspect [12].

Finally note that if conciseness is taken as the most important quality, then the winner of the shortest definition will be the author's of the MDA Guide with their glossary:

"metamodel: a model of models" [8], page A-2

The shorter, the better? This definition is correct. But those who missed the last "s" propagate the wrong idea that "a metamodel is a model of a model". A lot of such definitions can be found on internet. So don't miss the "s" (s stands for "set").

3.4 Metamodels and "ConformsTo" (χ , \rightsquigarrow)

Introducing the distinction between meta-models and modelling languages is distinctive feature of our MegaModel. The power of this feature will be further revealed in the next episodes. We admit however, that it is sometimes convenient to use short-cuts for the $\mu\epsilon\mu$ pattern (\rightsquigarrow). So let's review which short-cuts make sense taking the following figure as an example.

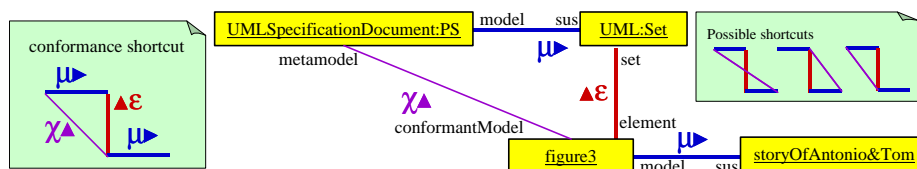


Figure 16 Example of a "conformance" short-cut (χ)

As shown in on the right of the figure, only three kinds of short-cuts are possible. Let's consider them from left to right.

- The short-cut from the system modelled (storyOfAntonio&Tom) to the metamodel (UMLSpecificationDocument) makes almost no sense, because these two systems are almost unrelated.
- The short-cut from the system modelled to the language is not of interest either. The story of the Antonio and Tom is not related with the UML language.
- By contrast, the last short-cut makes a lot of sense because we can state that Figure 3 is "conformant" to the UML specification document.

Out of three kinds of short-cuts, only one is meaningful, but this one is very important. In fact, though the concept of set and \mathcal{E} have not been identified in other mega-models, these mega-models introduce the third kind of short-cuts as it was a basic relation. It is called *ConformantTo* in Bézivin’ mega-model [4] and *LinguisticInstanceOf* in Atkinson and Kühne mega-model [9]. This relation is simply called *InstanceOf* in the context of the MDA, but this terminology is not adequate. In Ancient Egypt, this relation was written χ . In this series, the χ Greek letter will be used for short.

Anyway, besides the terminological debate, one can wonder if this relation is really useful in practice. The answer is definitively yes. The *ConformantTo* short-cut, is far more useful than the *ElementOf* (\mathcal{E}) relation, which is only interesting for understanding purposes. As said before languages (i.e. sets) are usually infinite and are really abstract systems. \mathcal{E} is not operational. This contrasts with *ConformantTo* which relate models. For instance the spelling checker is a concrete software system that plays the role of meta-model when it is used to check if a given text is conformant or not. Similarly a java parser is a meta-model that is used daily by programmers to check whether their programs are conformant or not. Since the *ConformantTo* short-cut between a model and a metamodel is often used, it deserves the extension of the *MegaModel* by a derived association (Figure 17).

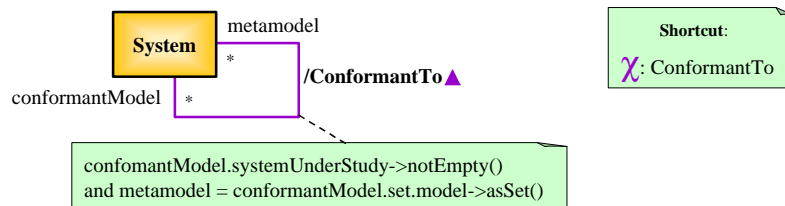


Figure 17 MegaModel: ConformantTo (χ , χ)

Note again, that this definition of the conformance relation is very weak. In particular, one can argue that checking the conformance between a model and a metamodel is one thing, but that very often we need more. For instance, a metamodel allows to decompose a system, to check to conformity of each element of this system, etc. They are various levels of conformity and various services that can be associated with that relation. This episode introduces only the weakest form. The lesser, the better. More later. The relation as introduced here is "good enough" for the remainder of this episode.

4 Meta-pyramids

So, now that the *ElementOf* (\mathcal{E}) and the *ConformantTo* (χ) relation have been introduced, the structure of complex $\mu\mathcal{E}$ -graph can be considered. The goal is to see if there is some tool to better understand the global architecture of Figure 15 for instance. As said in the introduction, some recurring patterns usually appear in $\mu\mathcal{E}$ -graphs. The χ relation can be used to get χ -graphs or $\mu\chi$ -graphs that are usually simpler to grasp. This section shows that meta-steps are the basic elements of meta-pyramids.

4.1 Meta-steps and meta-meta stuff

Metamodels make modelling languages explicit. Since metamodels are models, one can go one step further and apply the $\mu\epsilon\mu$ pattern once more. This leads to the notion of meta-metamodel (Figure 18). One step more and one would obtain a meta-meta-meta-model, etc. We call each such step, a *meta-step*. The prefix *meta* is mostly used to warn the reader that after each step things are getting more abstract and difficult to grasp. In practice however, the notion of meta-metamodel is not really useful, because the ConformantTo (χ) relation is not transitive.

Consider for instance the figure below. The model named Figure 3 is not related to the MOFSpecification. In other words, one could say that MOFSpecification plays the role of meta-metamodel with respect to Figure 3 but this statement is almost useless. The meta-meta prefix only helps in understanding that the system we talk about is two level higher than another one.

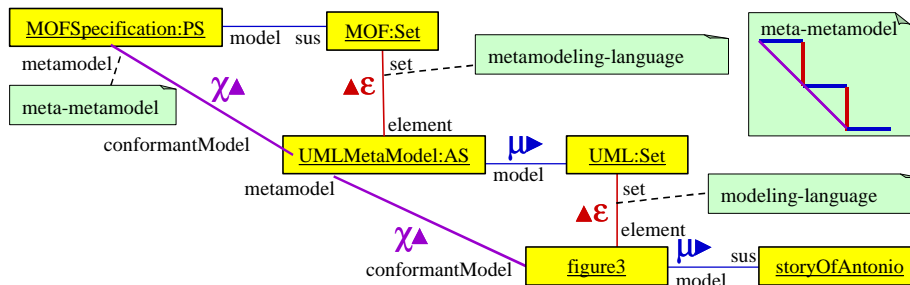


Figure 18 Example of a meta-metamodel (χ)

Episode I showed that the notion of model was a relative notion. This characteristics of the μ relation, also holds for χ . For instance, the MOF specification is *not* a meta-metamodel per se. It just plays this role in the figure above, but this can change. The MOF could be used for instance instead of UML to model the Story of the Antonio and Tom at the Louvre Museum. In this case MOFSpecification would just play the role of meta-model, losing its role of meta-metamodel. What is more, one can see the Figure 4 as a model of the language used to describe Figure 3. That is Figure 3 is conformant to Figure 4, which is conformant to UMLMetaModel, which is conformant to MOFSpecification, which is conformant to itself. One could said that the MOF specification is a meta-meta-meta-model or a meta-meta-meta-meta-model or even more if one use the circular structure. All this just shows that the position of a system in a meta-step chain is relative, not absolute. Being a meta-meta-meta-model, a meta-metamodel, a meta-model or simply a model just depends on the context considered.

4.2 Modern meta-pyramid(s)

Considering things as absolute is a common mistake. With this respect, the various OMG standards largely contribute to this confusion. In the famous "4-layer metamodels architecture", the various levels were coined M0, M1, M2, and M3 (see Figure 4). A much better terminology would be M+1, M+2, M+3, etc. In recent versions of OMG standards, the name of the layers are still the same. More emphasis has been put however on the relative nature of the meta-steps.

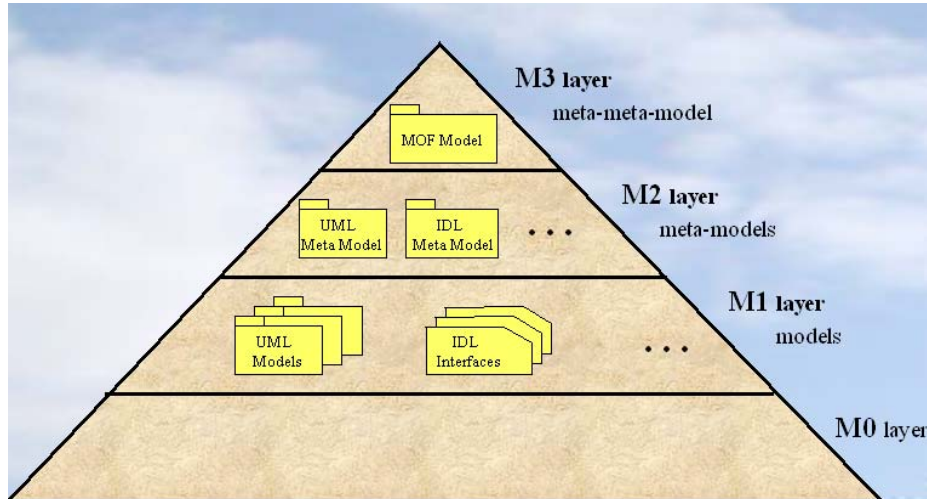


Figure 19 Coloured version of Figure 2.2 from the MOF specification [6].

For instance the specification of UML 2.0 infrastructure [14] says: "*This structure (meta-step) can be applied recursively many times so that we get a possibly infinite number of meta-layers; what is a metamodel in one case can be a model in another case, and this is what happens with UML and MOF. (...) MOF is commonly referred to as a meta-metamodel, even though strictly speaking it is a metamodel (...)*".

Einstein was certainly right when he claimed "everything is relative". Nevertheless, it is convenient in most common situations to use a simpler and more static model. For instance Newton' model of physics is "good enough" in most cases. Though incorrect and older, this model is also simpler to catch. Simplicity is important for many practical purposes, and in particular for dissemination. The OMG' 4-layers architecture has proved extremely valuable with this respect. While meta-modelling techniques stay for long confined in the labs, the OMG standards have largely contributed to its recent success. These standards are used and are useful. Their content should just be interpreted as in a "relative" rather than "absolute" way. For instance the statement "*the MOF specification is a meta-meta-model*", should be read "*in the context of the MDA, the MOF specification typically plays the role of meta-meta-model (though this is not necessarily the case)*".

While, the OMG' meta-pyramid is certainly the most referenced in the literature, various authors provided other models to describe the meta-modelling space. Some proposals are shown in Figure 20. In [14] the software space is decomposed further according to various dimensions. Many proposals have been made recently, but as pointed out by Bézin in [4] it will take some time to come with a good model of the meta-modelling space. The goal of this episode is not to study all the variants of the meta-pyramids. It would take too much space. On the contrary, the next section shows how a safe meta-pyramid can be built using only the relation described so far, that is RepresentationOf (μ), ElementOf (ϵ) and ConformantTo (χ).

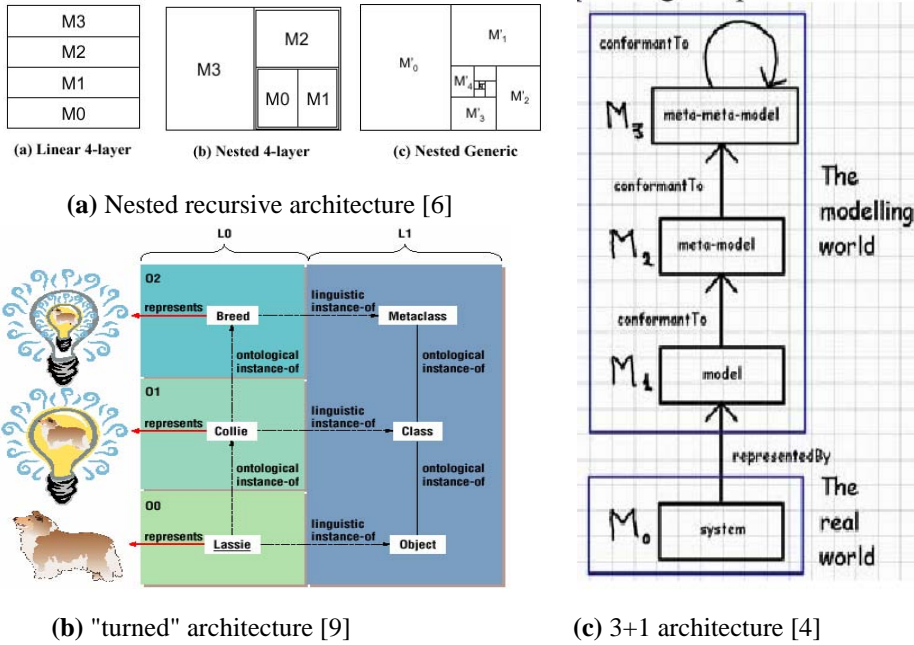


Figure 20 Other meta-modelling architectures

5 Story of Thotus the Baboon

Let's go back to the historical foundations of ancient meta-modelling. Thousands of years ago, Nivizeb the priest had already studied this problem. Since he dedicated all his life to these kinds of matters, it is interesting to see the evolution of his thoughts.

5.1 From the Temple of objects to the Temple of models

When Nivizeb was very young he was fascinated by objects. He gathered a lot of objects and kept them in his temple. At that time Nivizeb motto was "Everything is an object". He collected objects such as god statues. For instance, the Thot statue displayed now in the Louvre Museum was one of its favourite object. As a great priest, Nivizeb spent a lot of time observing the objects stored in his temple, to see if these objects were sending messages. He was very upset with the Thot statue because he never received a message from that object.

After some years he found that his vision of the world as a set of object was not enough. Moreover, he found that this statue was better though as a model of the God Thot. From then he decided that his temple would be filled with models instead of object. His new motto became "Everything is a model". Episode I showed that Nivizeb the priest had perfectly identified the role of the RepresentationOf relation (μ) [5]. He used to represent this relation horizontally and since he built and stored most of his models

in the temple, he first used the so-called "temple architecture" (Figure 21) as way to organize model. In this model, the temple was representing the world of models, while the outside world was just the world where "ordinary" systems lived, such as Nivizeb crocodiles for instance. Nivizeb kept his models in the temple, while people and things outside where just systems under studies (SUS).

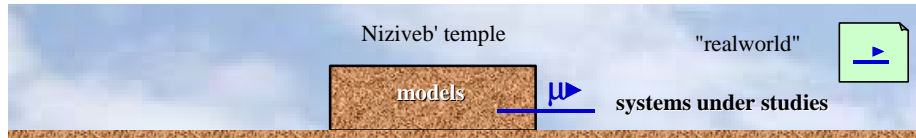


Figure 21 Nivizeb' Model Temple

5.2 Thotus the Baboon, on the roof; Thotis the Ibis, in the sky

Nivizeb had built many models, but he knew that the outside world virtually contains an infinity of systems, and therefore required much more space than his temple. His temple looked actually very small in the horizon of the desert. However, as Nivizeb collected more and more models he discovered the importance of sets and of modelling languages (section 3). According to Nivizeb, these concepts were so abstract that they were certainly provided by the gods. These concepts were certainly somewhere if Nivizeb theory was correct, but he didn't find where. This was a real enigma for Nivizeb.

Nivizeb had observed however, that his baboon was often climbing on the roof of the temple, and the one of his ibis was always in the sky around the temple. During years Nivizeb had not cared about that. Nivizeb had called his baboon Thotus, and his ibis Thotis, both after the God Thot (Thot was modelled either as a baboon or as an ibis in Ancient Egypt). Thot was precisely that God that invented languages. Nivizeb had spend a lot of time in looking for evidences of the existence of sets and languages. He had looked both inside and outside the temple. Thotus the baboon and Thotis the ibis, gave him the solution of the enigma. Nivizeb knew that languages was the creation of the God Thot, and as such they surely lived in the sky! If his theory was correct two property had to be satisfied:

- (1) "Ordinary" sets had to be in the sky above "ordinary" systems, outside the Model Temple, where Thotus the Ibis flew.
- (2) Modelling languages had to live in the sky above the Model Temple, where Thotus the Baboon spent most of his time.

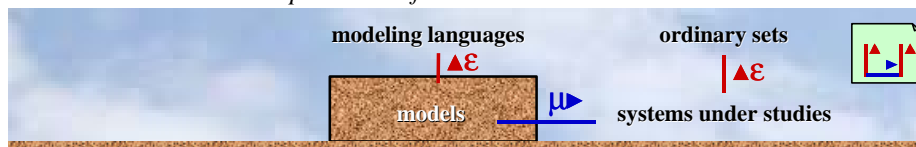


Figure 22 The temple and the sky

Nivizeb started with an attempt to validate the first property. In fact he already knew that it could be satisfied, because it was at least consistent with his classification of the sacred animals. Those animals lived in the real world, outside the temple (Figure 23). For instance, Lassus and Lacostus the crocodiles were on the floor outside the Model

Temple. Sets such as crocodile and reptile were in the sky. Sets of sets such as species or genus was even higher in the sky. Gods certainly didn't want humans to find their secrets so they had made very difficult for human brains to elevate so high in the sky. For instance, most people would not understand why species, genus and kingdom are not linked together, although they all pertained to the set named biologicalRank which is a set of a set of a set (not shown in the figure, too high in the sky). This definitively made a lot of sense: Gods really didn't want Nivizeb to find their secrets about the organization of the world.

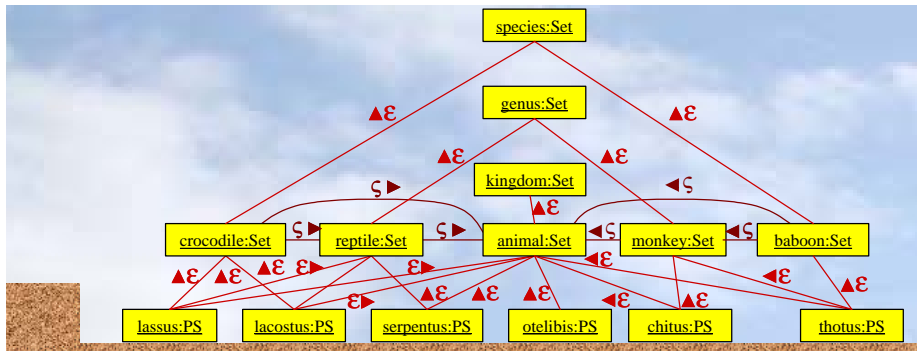


Figure 23 "Ordinary" sets pyramid outside the Temple of Models ($\epsilon\varsigma$ -graph)

To check the correctness of the second property, Nivizeb had decided to follow Thotus the Baboon. Very often Nivizeb went on the roof of the temple, to see if modelling languages were actually there (remember that modelling languages are languages of models). He had spent months and months there with Thotus. Eventually he understood that gods were not willing to provide him physical clues of the existence of modelling languages. He understood that he could nevertheless build himself models of these concepts. He could model modelling languages in the same manner he had modelled Lassus, his favourite crocodile. In fact, Nivizeb produced on the roof of his temple the first meta-models ever built on this planet. He worked on the roof just as he was working on the floor. Nothing had changed, he just modelled what was in his mind.

5.3 The first meta-pyramid on earth

After a while Nivizeb had produced various meta-models. He ordered to built a smaller temple on the roof of the Model Temple to kept the meta-models. Since meta-modelling engineering was an hard activity he didn't want to be disturbed. When strangers were approaching too close, he used to shout "meta" (that is "go away" in very Ancient Egyptian). Climbing on the roof required a substantial effort, but he soon understood that the meta-step was a little step for him, but a big step for the mankind.

Nivizeb was really brilliant. He took him only one week to understand on the roof of the meta-temple that he could just do the same and built a meta-meta Temple to store his meta-modelling languages. From there he had to shout very loud "meta, meta" to be heard by the strangers on the floor. Everybody in egypt thought Nivizeb had lost his mind. From then they started to considered the "meta" pyramid with fear.

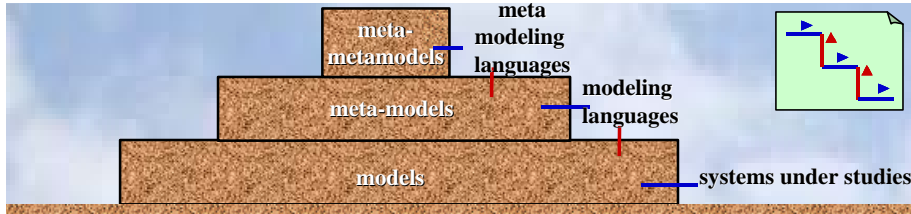


Figure 24 Nivizeb' meta-step meta-pyramid

Nivizeb had understood that he could climb as many meta-steps as he wanted, at least in theory. Though each steps would lead him closer and closer to Gods, he thought that climbing this stairways to heaven was be also quite boring after all. He knew that the pyramid had to stop raising at some point and that anyway defining a self-describing language was piece of cake. After all, he was speaking Copt when he was teaching Copt. Students were disturbed at the beginning but Nivizeb didn't care. Nobody dared to complain. Nivizeb crocodiles were very bad.

Nivizeb knew that future archeologists would probably thought that his pyramid had three levels. They might even use strange names such as M1, M2, and M3. Actually, he was not very concerned by the confusion his pyramid could generate in the mind of future generations. He thought at his pyramid as a relative structure and he carefully avoid to insert absolute references at the various levels. When he was working alone in his pyramid he didn't care at which level he was. He just had to know what he could find in the level above and in the level below. Nivizeb also knew that combining two meta-steps was not meaningful. It was even dangerous for simple minds. Only fools would jump from level N to level N-2.

5.4 Adding stairways

Nivizeb was getting old. Climbing on the roof and then entering to the next level was fun when he was younger, but he had to optimize the pyramid paths because he was constantly going up and down. He start to study the possibility of building short-cuts to simplifying his work. Only one kind of short-cuts out of three really make sense (Figure 25, top-left corner). Nivizeb ordered to built stairways inside the pyramid to get direct access from models to meta-models.

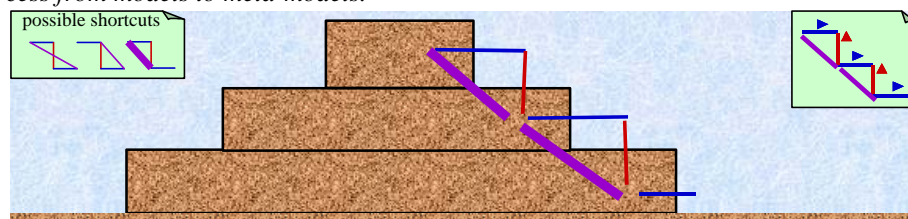


Figure 25 Stairways to meta-model

Stairways avoid him the burden of going to the roof to consider modelling languages. Thought he knew that modelling languages were actually there, he found no practical uses of these abstract systems. On the contrary, when he wanted to check if a model was conformant to a given meta-model is just had to run upstairs. This path was operational.

5.5 Evolution of Nivizeb meta-pyramids

Nivizeb was no longer using the platform on each step, so he decided to convert his pyramid to a geometrical pyramid by filling the steps. In fact, during his life Nivizeb had imagined many different architectures to structure his meta-modelling space. He had drawn many models on various kind of supports. These models are today kept in the basement of the Museum of Grenoble in a small chest (Figure 23.a). This chest also contains a model of Thotus the Baboon.¹



(a) Nivizeb' chest
(gold and ivory)



(b) Thotus the baboon
(painting from Nivizeb)

Figure 26 Artefacts at the Museum of Grenoble [17]

The history of Nivizeb designs can be decomposed in three periods. At the end of the first period (Figure 27), Nivizeb started to pay more attention to the volume of the pyramid, not only to his shape. And he was right. After some time, he founded that he spent much more time in the lower levels and that these levels were crammed full of models, while the upper levels contained fewer and fewer (meta)models. In other words the space was lost and the architecture was no more reflecting realm of meta-modelling. Flattening the pyramid was not enough (Figure 27.4). The upper levels were still almost empty and the lower levels too packed.

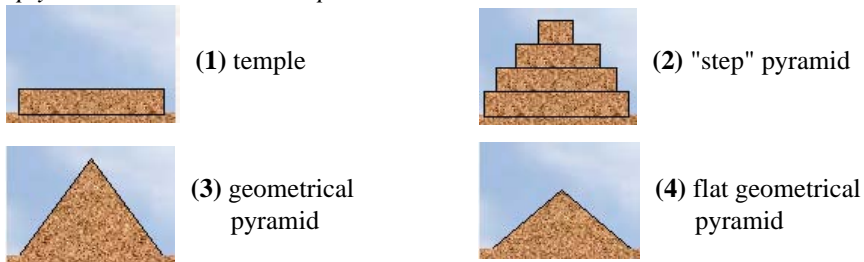


Figure 27 Nivizeb' meta-pyramid architectures. First period.

A recent study suggests that the volume required by each level decreases by one order of magnitude when going from one level to the level above [13]. Nivizeb was not aware of that dramatic reduction, but he drawn however various other pyramid architectures

1. At the time of writing this paper, the author was not allowed to photograph the original designs of Nivizeb pyramid, so the various figures in this section show only sketchy models of these designs.

in the second period (Figure 27). Note that (6) is certainly his closest approximation to a logarithmic reduction (though the pyramid are depicted in 2D they are actually 3D volumes). In this architecture a huge space was devoted to models, a smaller space for meta-models and on top, only a really small space to keep only few meta-meta-models. What had always been clear in Nivizeb' mind is that his pyramid would always be infinitely smaller than the Earth, because only a infinitesimal small sub-set of the systems from the "real-world" could be actually modelled. His pyramid looked only as small point in the horizon of Egypt.



Figure 28 Nivizeb' meta-pyramid architectures. Second period.

Nivizeb had never stopped his research on meta-modelling during his life. He had imagined more architectures. A few of them, from the third period, are depicted in Figure 29. Unfortunately the rationale behind these architectures disappeared when he died. For instance, the actual purpose of the "glass" pyramid (Figure 29.10) remains unknown. This architecture is certainly the most elegant and it certainly corresponds to a bright design. But nobody know which are the relations exhibited by this very regular structure. Further research is required on meta-modelling archeology.

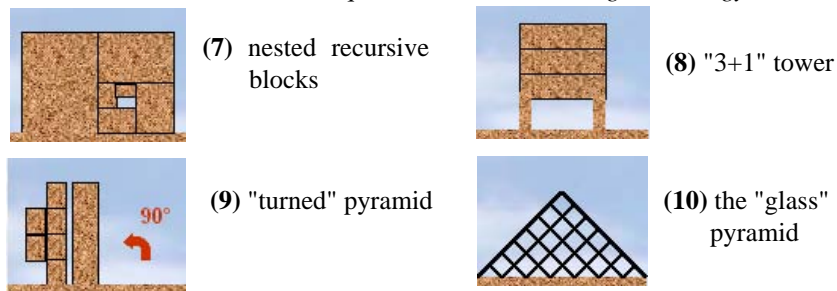


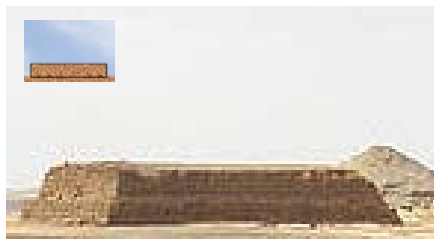
Figure 29 Nivizeb' meta-pyramid architectures. Third period.

It makes no doubt that though very clever, Nivizeb probably made some errors in his life. For instance the "3+1 tower" in Figure 29.8 might be an example of bad design. We can assumed that the tower is a representation of 3 meta levels. The tower seems to "float" in the air (four thin pillars could have been be used to sustain the tower). In fact, this representation of meta-modelling would be confusing because the μ and χ relation would be both represented vertically. This would lead to a $\mu\chi\chi\chi$ sequence similar to the one depicted in Figure 20.c. One can wonder who, in the real world, would dare to stay below that tower?

5.6 From meta-pyramids to pyramid engineering

In fact, during his life Nivizeb had only the opportunity to validate some of his models. He was for sure a great priest, but he also has a limited amount of resource. He perfectly knew that pharaohs would never pay attention to his research in meta-modelling. Getting funds was also very difficult at that time. But Nivizeb was clever and he had a strategy. Personally, he was not hurried to meet the Gods in the sky, but he knew that

pharaohs would certainly be interested by a stairway to heaven. And his plan worked very well indeed. He got huge amount of resources, thousands and thousands of slaves, hundreds of boats to bring materials on the Nile, etc. During his life he only had the time to direct the building of a few pyramids. But generation after generation egyptian architects used Nivizeb models to drive the engineering of the pyramids. Architects never knew what had been the original purpose of these models. Many pharaohs were buried in pyramids in the hope to reach the sky thanks to pyramid. Figure 30 presents the most famous pyramids of Egypt. They form Nivizeb's legacy to meta-model engineering.



(1) El Faraun



(2) Saqqara "step" pyramid



(3) Menkaure pyramid



(4) Dahshur "red" pyramid



(5) Dahshur "bent" pyramid



(6) Meidum pyramid

Figure 30 Nivizeb' legacy [17]

It is clear that all architectures designed by Nivizeb in first period and the second period were technically sound. The pyramids actually scaled up very well. By contrast, it is difficult to determine which of the architectures from the third period (Figure 29) were actually used by egyptian architects. It is clear however that they are plenty of ruins in Egypt (e.g. Figure 31).



Figure 31 Ruins of unidentified buildings [17]

6 Conclusion

The Saqqara "step" pyramid was built four thousand years ago [17]. At that time, the pyramid was isolated in the desert. It was the most prominent representation of meta-modeling in ancient Egypt. In fact, it played a role similar to the role played today by the MDA meta-pyramid [2]. The MDA set of standards is arranged within a unique pyramidal structure, with a single and unique meta-metamodel on the top, namely the MOF [3]. In episode III, it will be shown that seeing the MDA meta-pyramid in isolation was a wrong interpretation of the MDE approach. Though the first pyramids were grown separately in the desert, pyramids were later grouped in a structured way as shown in the figure below.



Figure 32 The Ghiza plateau: a organized set of technological spaces [12][17]

The story of the Ghiza plateau will be related in [12]. It will be shown how pyramids model of various height correspond to different technological spaces. A computer model from the University of Chicago will be used to show ancient bridges that existed between pyramids. The story of the Rosetta Stone will show that model and meta-model should not be considered in isolation, but that the key to understand real world is to bridges between sub-worlds [12]. It will be shown in further episodes how concepts such as decomposition, interpretation, syntax, semantics, transformation could fit in the mega-model.

7 Acknowledgments

I would like to thank Jean Bézin, Jacky Estublier, German Vega, Colin Atkinson, and Thomas Kühne, for the fruitful discussions we had on this topic. I would also like to thank Vincent Lestideau to introduce me to Thotis the Baboon, Jean Bézin to introduce me to Antonio and Stéphane Ducasse to introduce me to Tom. Thanks to all participants of the Dagstuhl seminar on Language Engineering as well as the participant of the AS MDE project [19]. Discussions largely contribute to this series, but all errors are mine. Finally I would apologize for the errors that might have occurred during historical narrations.

8 Photographic credits

The photos in this paper have been graciously provided by the following individuals or organizations. Special thanks to John Bodsworth for its help. Most of the photography have been processed to improve printing on grey-scale printer.

- J. Bodsworth, The Egypt Archive, <http://www.egyptarchive.co.uk>
- History Link101, <http://historylink101.net>, Figure 8.a, courtesy of Kersker.

9 Bibliography

- [1] "From Ancient Egypt to Model Driven Engineering", series and resources available at www-adele.imag.fr/mda
- [2] OMG, "OMG, "Model Driven Architecture (MDA)", ormsc/2001-07-01, July 2001, available at www.omg.org/mda
- [3] OMG, MDA Web Site, www.omg.org/mda
- [4] J. Bézin, "In Search of a Basic Principle for Model-Driven Engineering", Novatica Journal, Special Issue, March-April 2004
- [5] J.M. Favre, "Foundations of Model (driven) (Reverse) Engineering: Models - Episode I: Stories of the Fidus Papyrus and of the Solarus", post-proceedings of Dagstuhl Seminar on Model Driven Reverse Engineering, 2004, available at www-adele.imag.fr/~jmfavre
- [6] OMG, "Meta Object Facility (MOF) Specification" Version 1.4, April 2002
- [7] OMG, "MDA Guide Version 1.0.1", omg/2003-06-01, June 2003
- [8] E. Seidewitz, "What Models Mean", IEEE Software, September 2003
- [9] C. Atkinson, T. Kühne, "Model-Driven Development: A Metamodeling Foundation", IEEE Software, September 2003
- [10] J. Bézin, O. Gerbé, "Towards a Precise Definition of the OMG/MDA Framework", Proceedings of ASE'01, November 2001
- [11] J. Alvarez, A. Evans, P. Sammut, "MML and the Metamodel Architecture", available from www.puml.org
- [12] J.M. Favre, "Foundations of Metamodel (driven) (Reverse) Engineering - Episode III: Story of the Ghiza plateau and of the Rosetta Stone", available at <http://www-adele.imag.fr/~jmfavre>

- [13] J.M. Favre, ""Meta-models and Models Co-Evolution in the 3D Software Space"", Proceedings of ELISA, Workshop on the Evolution of Large scale Industrial Software Applications, Joint workshop with ICSM, Sept.2003, available at www-adele.imag.fr/~jmfavre
- [14] OMG, "UML2.0 Infrastructure Specification", ptc/03-09-15, September 2003
- [15] I. Kurtev, J. Bézivin, M. Aksit, "Technological Spaces: an Initial Appraisal", CoopIS, DOA'2002 Federated Conferences, Industrial track, Irvine, 2002
- [16] D. Karagiannis, H. Kühn, "Metamodelling Platforms", Third International Conference EC-Web – Dexa 2002, LNCS 2455, September 2002
- [17] J. Bodsworth, "The Egypt Archive", <http://www.egyptarchive.co.uk/>
- [18] A. Kleppe, S. Warmer, W. Bast, "MDA Explained. The Model Driven Architecture: Practice and Promise", Addison-Wesley, April 2003
- [19] CNRS, "Action Spécifique CNRS MDA", Centre National de Recherche Scientifique, Web site at <http://www-adele.imag.fr/mda/as>
- [20] J.M. Favre, "CaCophoNy: Metamodel-Driven Software Architecture Reconstruction", Working Conference on Reverse Engineering, November 2004