Formal Verification of Application Domain Using MDA and OWL

Silva J. B., Pezzin J., *Federal University of Bahia and Federal University of Espírito Santo.*

***Abstract*— The applications conceptual models, during the requirements identification phase, uses an independent platform model for abstract a great part of the system specification without a formal verification, be either syntactic or semantic. This paper proposes an architecture for constructing MDA models using OWL (Ontology Web Language) formalizing concepts about application domain and reducing the time of models maintenances during the business rules specification phase.**

***Index Terms*— Ontology, model, MDA, OWL.**

I. Introduction

The MDA (Model-Driven Architecture) considers the creation of models in different abstract levels, separating the interests of architecture to be implemented. Using the application conceptual model for generation of another one specific platform model, reducing drastically the number of the maintenances for the same application domain. The objective of this work is to consider the ontology in the construction of a domain independent model, because the combination use of UML and MOF (Meta-Object Facility) not is formally sufficiency for that.

There are some troubles in the MDA independent models use, some examples are: mix the application conceptual part and business rules within the meta-model, the use of architectural patterns for transformation rules still is not mature sufficiently and there is not a good synchronization between models and programming language. The UML language used in the MDA models construction, according with Freitas[1] cannot be considered a formal model of representation, by the absence of a inference engine or a formal semantics. Thus, cannot be formally measured the requirements satisfaction on the conceptual domain. Exclusively during the functional requirements specification phase, when the class models are enough generics, abstracting a lot of system specification part and are constructed using a human idiom that not provides a formal verification about the semantic and declarative stakeholders information.

Manuscript received April 2, 2007.

Silva J. B. is with the *Distributed Systems Laboratory (LASID),* Federal University of Bahia (UFBA), Salvador, BA 40302-370 Brazil (e-mail: jaguarac@ufba.br).

Pezzin J. is with the Software Engineering Laboratory, Federal University of Espírito Santo (UFES), ES 29060-900 Brazil (e-mail: juliana_pezzin@hotmail.com)

This paper considers the ontology in the construction of MDA models, with the objective to legalize the application requirements specification, reducing the maintenances in models during the specification of an application conceptual domain. The present paper is structured in five parts: The initial part explains the troubles about MDA-based application domain, the proposals considered in the paper and the organization it. Second part supplies to a general vision of their concepts and abstractions levels considered. In the third part, there is a brief presentation about ontology and the OWL language representation. The fourth part consists of a presentation about MDA model designed and formalized using an ontology-based tool within the architecture proposes for this paper. The conclusions and future work are in the fifth and last part.

II. Models and MDA abstractions levels

The MDA considers the models creation within the different abstract levels, separating the implementation interests from a specific architecture of an application conceptual model. These models are MOF-based, appraises four models levels such as: M3, M2, M1, and M0. The Figure 1 shows an example about theses meta-levels. The "umaConta" (M0) instance is an example of "Conta" (M1) class and this implements class (M2) "Classe", being an instance of the class "Classe" in MOF (M3).

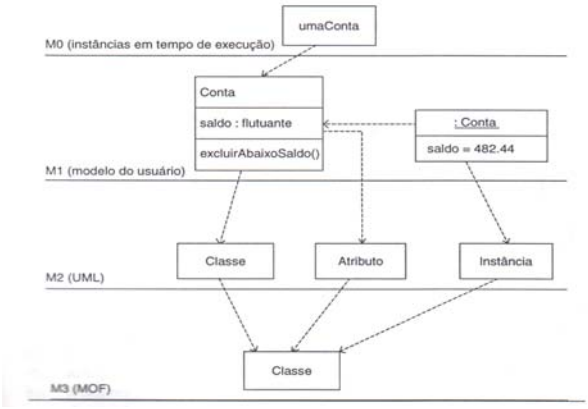

Figure 1. MDA abstractions levels [6], P. 48)

Each meta-model considered must have at least one or more implementations. A class diagram, for example, is

captured by one UML meta-model that describes as UML models can be structured, their elements and properties for a particular platform, thus as a UML profile[9]. The models will able to become related using mappings, which can be manually carried or across of automatic generation.

The mapping rules are basically defined for accepting one or more source models and these to be generated for a specific destination model. These rules are written in a meta-models level and all the origin models must to obey its meta-model specifications. Thus, for that it either possible to use on any UML models, a mapping is made on the model within the MDA tool, across of application domain created and interchangeable using XMI (XML Metadata Interchange). Schema inherited from XML (Extensible Markup Language) language format, UML and MOF stereotypes definition, and DTD (Definition of Type Document). The DTD declaration can be made external or internally on the document using tags as shows the table below.

Table 1 – Part of the XMI file generated using Protegê tool

```
<UML:Class xmi.id = 'a45' name = 'docente' isSpecification =
'false' isRoot = 'false'
  isLeaf = 'false' isAbstract = 'false' isActive = 'false'>
  <UML:GeneralizableElement.generalization>
    <UML:Generalization xmi.idref = 'a46'/>
  </UML:GeneralizableElement.generalization>
  <UML:Classifier.feature>
    <UML:Attribute xmi.id = 'a47' name = 'dataAdmissao'
visibility = 'private'
      isSpecification = 'false'>
      <UML:StructuralFeature.multiplicity>
        <UML:Multiplicity xmi.id = 'a48'>
          <UML:Multiplicity.range>
            <UML:MultiplicityRange xmi.id = 'a49' lower = '0'
upper = '-1'/>
          </UML:Multiplicity.range>
        </UML:Multiplicity>
      </UML:StructuralFeature.multiplicity>
      <UML:StructuralFeature.type>
        <UML:Class xmi.idref = 'a24'/>
      </UML:StructuralFeature.type>
    </UML:Attribute>
  </UML:Classifier.feature>
</UML:Class>
```

Table 1 shows an UML model in XMI format, following the standardization defined for the OMG (Object Management Group). The standard combines the benefits of the XML such as definition, validation and sharing of documents, with a visual language for modelling systems as the UML. The main objective of the XMI is to facilitate the interchange of meta-data between UML and MOF-based modelling tools.

III. OWL APPROACH

As well as XMI, OWL (Ontology Web Language) is a XML language. It is possible to express structure, concepts, relationships and to describe several special characteristics using logical axioms to form an ontology.

For Guizzard[2], in computer science, an ontology is a knowledge engineering device, consisting of a vocabulary or terms organized in a taxonomy. This definition is a set of formal axioms used to create new relations and to restrict its interpretations according to an intended direction. Falbo[3] proposes in this case, that an ontology can be seen as a model for an application domain, being used basically to specify and to develop semantic rules and increase their reuse.

Either category of an ontology is formed by sets of concepts, relations, properties, axioms and instances. A concept can be abstract or elementary or composed, real or fictitious or concrete. In a taxonomy will exist any concept formally justified. The man concept, for example, can be a sub-concept of a person concept. Relations between the concepts can also exist. For example, between the person concepts and a car, the relationship “be-owner”. The properties represent the relationships between the concepts. The axioms are rules or affirmations of the truth on concepts. An axiom example is to affirm that every person has a mother. There are also, instances that represent a previous knowledge of an ontology.

In knowledge base, the intentional (TBox) and extensional (ABox) knowledge can be enclosed. They are used across of DL (Description Logic) to carry through to mapping OWL language, in order to process the knowledge and to find implicit information about ontology using a reasoning tool, such as Rise[5]. Inferences are made to test the TBox through using a deductive system for concepts satisfaction, sub-classification, equivalence and disjunction. With ABox, is possible to check consistency of the semantic model using instances on just-time in the construction application domain phase and not during the development, where normally is used a programming language. With these tests, the class hierarchy and its relations, rules in the attribution of values to their attributes and properties can be checked. At last, the models verification time and maintenances number shall be decreasing.

IV. FORMAL VERIFICATION OF MODELS USING OWL

In this proposal, the MDA ontology-based model follows MOF standardization, using in the data interchange data generated using XMI schema between meta-levels. According with OMG specification, on M2 meta-level is possible to materialize the specified class in M3 (MOF) meta-level. Thus, on level 2 a model can be used to construction of MDA models, being implemented in the M1 meta-level, creating an independent platform model, across of the construction class diagrams and esterotypes for an application domain within the UML tool. The generation of the rules mappings for a specific platform occurs on M1 meta-level, using MDA models tools, after the

construction of the conceptual model in the M1 meta-level, using the ontology edition tool Protégé [4].

Figure 2 presents the architecture proposal, starting from the construction of an ontology for an application domain until the formal verification model, before of transformations in another one to a specific platform.

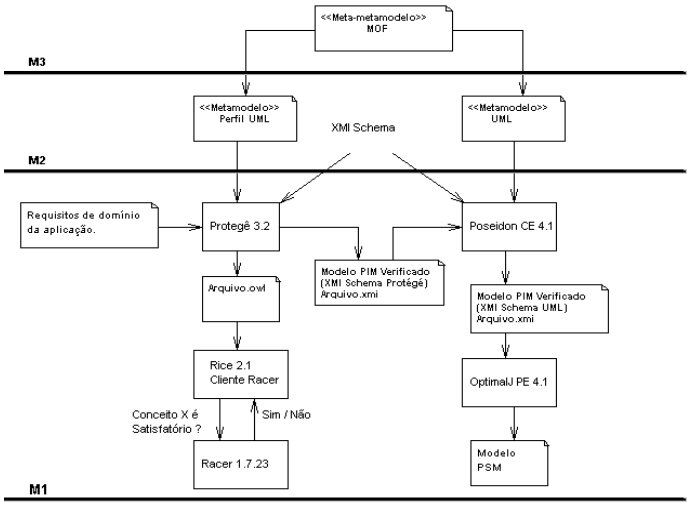

Figure 2. Architecture used for verify formally an application domain using OWL

All the conceptual model will be formally verified, creating class instances in the ontology logical tools, using description logics on the Racer tool [4]. For Ribeiro [5], the Racer (Reasoner will be Aboxes and Concept Expressions Renamed) can automatically, using OWL-DL (Description Logics) realizes inference on the OWL file generated. These inferences can be submitted using the Rice (Racer Interactive Client Environment) [4], that makes the use of a graphical interface to import OWL files.

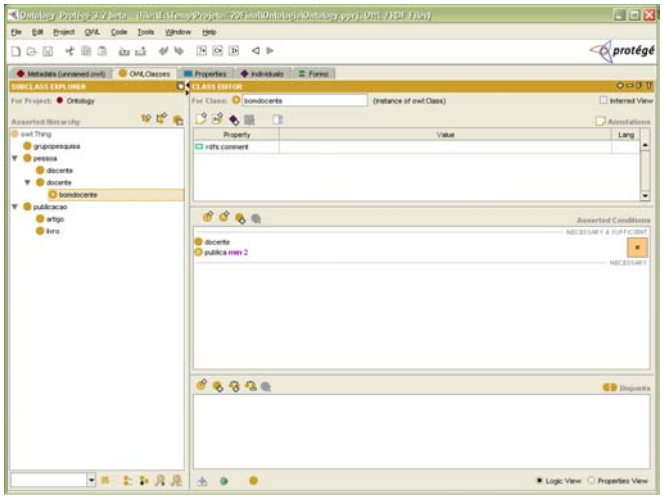

Figure 3. Ontology edition tool Protêgê [5].

The concepts, relations and properties identified in the ontology are formal and have a semantic definitions. In the ontology created in figure 3, a person, for example, it means a professor or a student who works in a university, the definitions of the concepts cannot be ambiguous. The axiom "publishes >=2", means that a good professor has a more than one publication, either a paper or a book. The ontology is constructed following the application functional requirements, and its model will be verified to satisfy it.

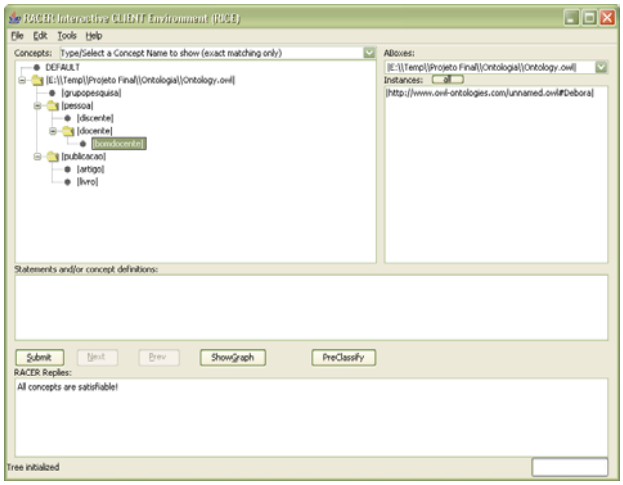

Figure 4. Formal verification of the application domain using a reasoning tool

In figure 4 is possible to visualize the inferences result on an application domain. In this example, all the concepts had been satisfied. The model example will be interchanged for an UML tool for the attribution of types of class, stereotypes and other UML meta-model elements, beyond other available profiles in the M2 meta-level.

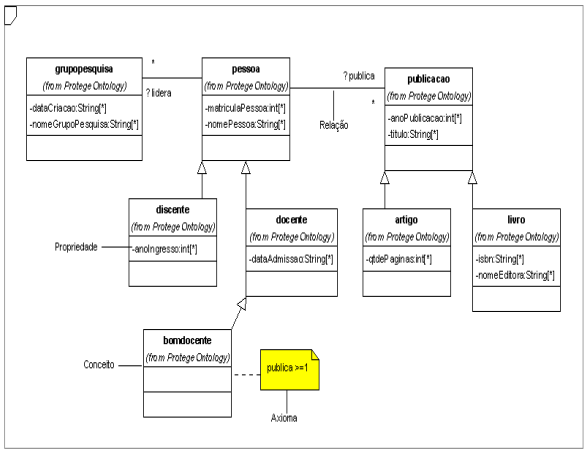

Figure 5. UML class diagram about ontology.

According to OMG specification, the data-exchange standard in the Model-Driven Architecture (MDA) is the XMI; used by UML tools to generate an independent platform model (PIM); the Poseidon [8] was used to import the XMI file generated from Protégé tools and to create an application model in the M1 level. This was necessary because of the differences found in XMI file exported from Protégé, being XMI archive generated by Poseidon accepted to importation in the MDA tool. The OptimalJ [7] was used to provide an transformations from PIM to PSM.

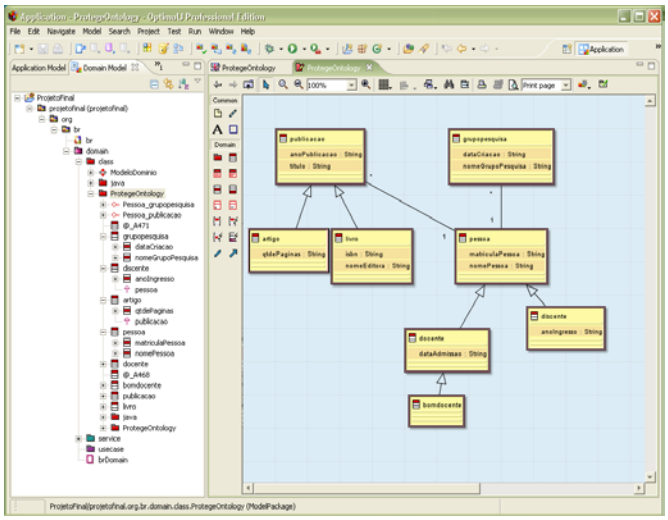

Figure 6. Creation of the PSM for database

In figure 6 it is possible to identify table elements, namely: primary and secondary keys and data types that are specific for database engine. With the use of a MDA tool, it is possible to transform models automatically, whose tool is used in the architecture because sufficiently easy to use and their development not to have been discontinued. In this proposal it was possible to identify two examples of models transformations: from PIM to PIM, being used in this case from OWL model mapping to UML, and from PIM to relational database (PSM).

V. Conclusions

The formal verification for an application domain is a valid alternative to prevent increase maintenances on the independent MDA models. During the development of this work a set of tools could be perceived to conceive models transformations and a lot of them free for commercial or personal use, however still there is not a mature level for professional use in contrast with the OptimalJ tool. With the use of the architecture proposal, the absence of a semantic definition using the UML language have been count out. The mechanism used between creation and validation of the application domain models using description logics, the semantics specifications can be guaranteed for having been used mathematical tests for its formal verification.

On sight to a future work is to use some software model process such as RUP (Rational Unified Process) or XP (Extreme Program) to measure the quality of the produced artefacts, benefits and to compare the use of this architecture in relation with other ones existing.

References

[1] Freitas, F. “Ontologias e Websemântica”. IV ENIA – Encontro Nacional de Inteligência Artificial, Campinas, Minicurso. In Anais do XXIII Congresso da Sociedade Brasileira de Computação, 52 p, 2003.

[2] Guizzard, Giancarlo. “Uma abordagem metodológica de desenvolvimento para e com reuso, baseada em ontologias formais de domínio”. Programa de Mestrado em Informática – Universidade Federal do Espírito Santo, Vitória, http://wwwhome.cs.utwente.nl/~guizzard/MSc/, Julho, 2000.

[3] Falbo, R.A., Guizzardi, G., Duarte, K.C. “An Ontological Approach to Domain Engineering”. Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering, SEKE'2002, pp. 351- 358, Ischia, Italy, 2002.

[4] Abdalla et al. “Módulo de Gestão do Conhecimento”. Universidade Federal da Bahia, http://twiki.im.ufba.br/bin/view/Residencia/Cursos, Abril, 2005.

[5] Ribeiro, M.. “Raciocínio em Lógica de Descrições”. Universidade Federal da Bahia, http://twiki.im.ufba.br/bin/view/Residencia/Cursos, Abril, 2005.

[6] Mellor, J. Stephen et al. “MDA Destilada: Princípios da Arquitetura Orientada por Modelos”. Editora Ciência Moderna Ltda., p. 15-169, 2003.

[7] OptimalJ. “Model-Driven Development for Java”. http://www.compuware.com/products/optimalj/, Julho, 2006.

[8] Poseidon. “Poseidon for UML 4.2”. http://gentleware.com/index.php, Julho, 2006.

[9] Brockmans, S., Haase P. A. “Metamodel and UML Profile for Rule-extended OWL DL Ontologies – A Complete Reference”. AIFB Institute, Karlsruhe Univertity, Germany.