MDA Based Tool for PLM' Models Building and Evolving

Onur Yildiz, Nada Aouadi, Aimad Karkouch, Philippe Pernelle, Lilia Gzara, Michel Tollenaere

MDA based tool for PLM' models building and evolving

Onur Yildiz[1,2], Nada Aouadi, Aimad Karkouch, Philippe Pernelle[3], Lilia Gzara[2], and Michel Tollenaere[2]

[1] Audros technology
F-69003 Lyon, France, 41 rue de la cité
oyildiz@audros.fr
[2] INP Grenoble, Laboratory G-SCOP
F-38000 Grenoble, France
lilia.gzara@grenoble-inp.fr
Michel.Tollenaere@inpg.fr
[3] University of Lyon 1, Laboratory DISP
F-69621 Villeurbanne Cedex, France
philippe.pernelle@univ-lyon1.fr

Abstract. Product Lifecycle Management (PLM) systems are sufficiently generic to propose models adapted to companies' specific needs without additional development. However, the implementation of such systems and their multiple reconfigurations may introduce coherence problems. To ensure structural and functional coherence while creating and evolving PLM models, a methodological approach is described in this paper. The proposed approach is based on Model Driven Architecture (MDA) principles in order to allow : [1] models' creation while respecting syntactic and semantic constraints, [2] models' evolution while respecting dependencies between the various PLM syste' components. This approach is being prototyping based on Eclipse framework.

Keywords: PLM system, MDA, metamodel, constraint, OCL, transformation, ATL

1 Introduction

Nowadays, industrial companies using PLM systems need to deploy models which are adapted to their requirements. PLM are information systems dedicated to managing the entire lifecycle of a product, from inception, through engineering design and manufacture, to service and disposal of manufactured products. In this research, we are focusing on these models' building and reconfiguration, more precisely from a PLM editor/integrator point of view. These models are generic enough to propose ones that can be adapted to each company' needs by doing some additional developments. However, in the case of initial PLM deployment, the elaboration of a PLM system that complies with the company's activities often requires one or more specific data models and a set of specific functions (processing).

Usually, the idea is to adapt the generic model, based on a set of templates which correspond to typical business needs (such as Catia data management, ProEng data management, MS Office reporting, etc.) so to reduce workload. Data model definition must be validated in order to avoid inconsistencies. This validation step consists of respecting a set of structural and functional predefined constraints. Thus, any new data model creation can be used permanently after the validation of data coherency.

Similarly, the various evolutions of already deployed systems made to adapt or add new functions in the data model are very difficult to perform for a company. In order to achieve this, companies are often accompanied by an editor or an integrator. In fact, a company must adapt to the ongoing development of its market or to any changes in its working methods. Therefore, it's often required to change the existent data model or system functionalities. But, in most often cases, these companies are not able to identify the impact of these changes ; there isn't any tool allowing them to have a global vision of the deployed PLM system in terms of components' dependencies. Without this global vision, changes may cause coherence problems or side effects. Thus, the phases of models' creation and evolving imply setting up a methodological approach for creating or maintaining the implemented system with a structural and functional coherence.

This research work aims at helping companies when they configurate or evolve their PLM system. The proposed approach consists of building a generic business model and providing a set of mechanisms to transform this generic business model into specific models according to each company' specificities. This approach is based on MDA. The rest of the paper is organized as follows. Section 2 describes the foundations of proposed approach using MDA. Section 3 gives the principles of proposed approach deployment. Section 4 describes the software prototype developed to support eh approach and finally section 5 gives some conclusions and perspectives.

2 Proposed approach foundations

MDA [6] [7] [1] is an architecture derived from MDE (Model Driven Engineering) approach proposed by OMG (Object Management Group). MDA uses the concepts of MOF (Meta-Object Facility) [8] and UML (Unified Modeling Language). It is based on multilevel modeling (or metamodels) and mechanisms for models' transformations. The MDA logic is to propose a refinement of models from a high level of abstraction to executable code. The fundamental rule of MDA is to separate the business logic from the implementation one. According to the three level approach proposed in MDA, models or metamodels are defined by a process of abstraction towards target platforms : Computation Independent Model (CIM), Platform Independent Model (PIM) and Platform Specific Model (PSM). In the rest of this section, we will describe these levels and mechanisms of transition between models of different levels.

2.1 Metamodeling within MDA

A metamodel aims to define the main concepts to be used in the PLM models [11] [5] . These concepts must be independent of the PLM system. They are used to characterize the conformity of the lower level models. The proposed elements in our metamodel [12] (fig. 1) describe the structural, behavioral and invariant concepts to be used in a PLM system. The diagram below shows the generic reduced metamodel with basic concepts for modeling business model.

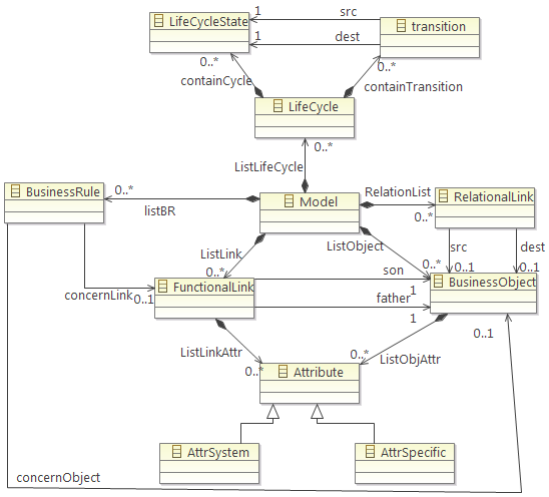

Fig. 1. Extract from Business metamodel

The fundamental metamodel elements make it possible to structure concepts in order to be used and deployed in a company. These concepts are structured according to descriptive elements (BusinessObject, AttrSpecific), structural relationship (FunctionnaLink), dependency relationship (RelationalLink) and Life cycle Management elements (LifeCycle). Business Rule concepts (BusinessRule) have to be considered as constraints that ensure business model consistency.

These concepts would enable building compliant business models. The business model includes business as in the activity sector and information system. This is justified by the fact that final implementation should respect PLM concepts (regardless of the platform). Therefore we believe to be consistent with the MDA approach "spirit" (the CIM is independent of execution platforms).

2.2 Validation and Conformity of models

The metamodel described in previous section is part of the proposed approach for building and reconfigurating PLM models process.It is one element among others

that allow developing consitent models . There is no standard metamodel. However, several studies have attempted to propose reference models [10] [4] .The proposed metamodel in this paper is generated from an abstract approach of these reference models . Concepts defined in metamodel are not always enough for the construction of robust models (they are not very sensitive to the context modifications). In fact, we used constraints in order to successfully express restrictions on built models [2], constraints must be defined.

A constraint represents a boolean-valued expression which can be attached to any metamodel element. It generally indicates a limitation, or gives further information on the model. They are used in most cases to specify invariants on the meta classes. These constraints complete existing diagrams by implementing business rules and making relationships more precise (without ambiguities). To add constraints on proposed models, we choose to use the Object Constraint Language (OCL) [9].

The model is a level (PIM) of modeling which has to enable implementing business concepts within a PLM system. In the context, the modeling is not unique and can evolve in time. The business concepts defining the models of this level characterize a contextual business terminology. Here the contextual meaning implies a low level of invariance for two reasons. On one hand concepts treated are very specific for industrial sector and the concepts used involve a certain ambiguity, on the other hand concepts treated are evolving over time

2.3 Model transformations

The aim of our proposal is to facilitate the conception and implementation of business models in PLM systems. Therefore, Meta Models of proposed approach (PIM level) have to be transformed into models that are dependent on target platforms (PSM level) according to the main steps of MDA approach. The last level, achieved by PSM generation, is then obtained by transformations between different levels, from CIM to PIM and from PIM to PSM

A PIM to PIM transformation consists of transforming a metamodel while respecting associated constraints. Likewise, a PIM to PSM transformation corresponds to the transformation of an independent functional model into a platform compliant model (with constraints specific to the chosen platform).

The transformation process could be described as a succession of 2 stages. In the first stage, the target metamodel is defined, and then mapped with the source meta-model. Mapping task is implemented using transformation rules. These rules allow producing a target model (conform to the target metamodel) starting from a source model (conform to the source metamodel). The second stage consists of executing the model transformations rules in a particular language (in our case, in SQL (Structured Query Language) script). Indeed, the PLM system configuration and data are stored in a database. Then, the last step of transformation process should make it possible to add, delete and update these data within the database. To sum up, a source model allows producing a target code thanks to the transformation process with Atlas Transformation Language (ATL)

3 Proposed approach deployment

As stated previously, providing generic concepts is not of much help when it comes to identifying enterprise's business concepts. This is because, most of the time, the experts to set up the PLM, are either business experts or IT experts but rarely both. On the one hand, our modeling's approach is independant of software tools and highlights the generic concepts of a PLM information system and business domains. On the other hand, we strongly believe that it should be supported by a methodological approach where various experts may simultaneously contribute to the business models' construction.

The suggested approach is based upon business-domain segmentation. Some business objects are well standardized to be used, as they are, in the enterprise. The main existent systems offer preconfigured models to ease building enterprise own business models and thus accelerating domain-specific model creation. The above-mentioned approach consists of building, progressively, parts of models by IT and business experts using each domain's invariant elements and business rules validation.

Specific templates

The domain is used to group concepts with adaptable granularity. In addition, some domains are sufficiently invariant (or normalized) in companies to be used as they are. We could mention for example : changes management domains, Engineering Change Request (ECR), Engineering Change Order (ECO) are CAD (Computer Aided Design) data management domains (part, assembly, ...). These domains can be elaborated by IT or business experts and then integrated progressively into the construction of the global PIM.

Business rules

Business rules Business rules characterize PIM or PSM concepts constraints and are, as mentioned before, described using OCL [3]. In the proposed methodological approach, they are used during the explicit validation, in addition to metamodels' implicit validation. Thus, business rules can be defined during every step during PIM and PSM construction. Similarly to domains, there could be two types of business rules : information system rules, business activity rule.

Models updates

In order for a company to continuously take into account economical environment changes, its information system and therefore its PLM system should be adapted. Updates can be identified at two levels : PSM's level and particularly its instance within the PLM (for example : adding some specific script) , PIM's level (for instance : business object modification, adding links, ...)

The advocated methodological approach is dual. On one hand, it is to make reverse transformations from instantiated PSM to PIM. On the other hand, it is

to provide PIM comparisons. This approach will enable experts to measure the impact of changes applied to the PLM.

4 Application

The suggested approach based on MDA is supported by a framework implementing models transformations and constraints validation mechanisms [13]. This framework is built on top of Eclipse platform and its extending capabilities likes EMF (implementation of the MOF providing tools for implementing DSLs (Domain-Specific languages). This framework is a way to verify and validate concepts introduced in the first part of this article.

Business model modeling

The construction of business model is done thanks to a graphical editor based on Eclipse GMF (Graphical Modeling Framework). This editor allows user creating his own business concepts based on a generic metamodel. The following diagram (Fig. 2) represents a business model with constraints. A violated constraint is marked with a red cross (i.e a BusinessObject can not have an empty name).

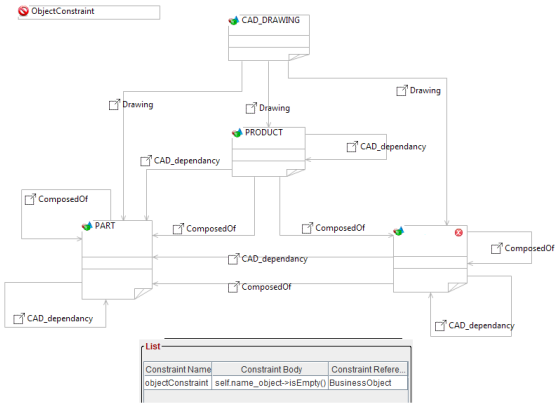

Fig. 2. Business model with constraints

Audros metamodel

In order for the model to be exploitable, an SQL script needs to be generated and applied to the PLM system. To achieve this, a metamodel defining Audros PLM's concepts have been created

The main goal is to generate an Audros model conform to the Audros metamodel from a business model edited with GMF and conform to the business

metamodel (introduced in the first part) by means of ATL transformation. The transformation results in a compliant Audros model. The transformation is done by iterating over the input model's content, comparing this content with the source patterns (correspondence rules defined in the .atl file) and producing target patterns in the target model whenever a matching occurs.

The following figure (Fig. 3) shows a case of ATL correspondence rule defined between the two metamodels. This rule takes a BusinessObject (defined by Audros metamodel) as an input and transforms it into an Object Class (defined by Database metamodel). The whole transformation process is used to translate and inject the Business Model into the database, to become accessible and usable within the PLM.

```
-- Règle de transformation d'un objet du modèle métier en une classe du modèle BD
lazy rule BusinessObject2Class {
    from
        b : Business!BusinessObject
    to
        c : DB!Class (
            name <- b.name_object,
            attribute <- b.ListObjAttr->collect(attr | thisModule.BusinessAttribute2Attribute(attr))
        )
}
```

Fig. 3. ATL rule

The obtained specific model is not yet exploitable in Audros PLM. It needs to be transformed into an SQL script to inject into the system database using existent tools in Eclipse (Acceleo and Liquibase).

The following figure (Fig. 4) illustrates Audros PLM's final state after the transformations' sequences applied to the model introduced earlier in this paper (the objects are usable within the PLM).

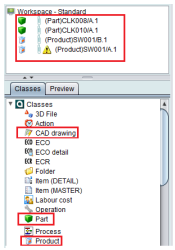

Fig. 4. Audros PLM result

5 Conclusion

The research work described in this paper aims at helping companies when they configurate or evolve their PLM system. The proposed approach consists of

building a generic business model and providing a set of mechanisms to transform this generic business model into specific models according to each company' specificities. The proposed approach is based on a model-driven architecture (MDA). One of the main goals of this approach, besides creating adequate models, is providing controlling mechanism in case of models evolution in order to keep total consistency in PLM systems. In the proposed approach, companies' specific needs are implemented through a refinement of models from a high level of abstraction to executable code. For that aim, three modeling levels are defined (CIM, PIM, PSM) and a set of transition mechanisms between levels is given. This approach is supported by software prototype which allows implementing concepts proposed in this paper. In this article, we concentrated on business model creation methodology. Now, a reflexive posture is conducted to address business model evolution methodology that ensures semantic and syntactic coherence with already existing PLM components.

References

[1] Bezivin, J., Gerbe, O.: Towards a precise definition of the omg/mda framework. In: 16th IEEE International Conference on Automated Software Engineering. San Diego, USA (November 2001)

[2] Cabot, J., Teniente, E.: Transformation techniques for ocl con- straints. Science of Computer Programming 68(3), 152168 (207)

[3] Dang, D.H., Cabot, J.: Automating inference of ocl business rules from user scenarios. In: Muenchaisri, P., Rothermel, G. (eds.) APSEC - 20th Asia-Pacific Software Engineering Conference - 2013. IEEE, Bangkok, Thaïlande (2013), http://hal.inria.fr/hal-00869234, atlanMod AtlanMod ITM-Factory, French FUI 14

[4] Eynard, B., Gallet, T., Roucoules, L., Ducellier, G.: Pdm system implementation based on uml. Mathematics and Computers in Simulation 70(5-6), 330–342 (2006)

[5] Le Duigou, J., Bernard, A., Perry, N.: Framework for plm integration in smes networks. Computer-Aided Design and Applications 8(4), 531–544 (2011)

[6] OMG: Omg model driven architecture [online]. http://www.omg.org/mda/ (2001)

[7] OMG: Mda guide v1.0.1 [online]. http://www.omg.org/mda/ (2003)

[8] OMG: Meta object facility (mof)[online]. http://www.omg.org/spec/MOF/2.4.1/ (2011)

[9] OMG: Object constraint language omg available specification ver- sion [online]. http://www.omg.org/spec/OCL/2.4/PDF (2014)

[10] Sudarsan, R., Fenves, S., Sriram, R.: A product information modeling framework for product lifecycle managemen. Computer Aided Design 37(13) (2005)

[11] Terzi, S., Panetto, H., Morel, G., Garetti, M.: A holonic metamodel for product lifecycle management. International Journal of Product Lifecycle Management 2(3), 253–289 (2007), http://hal.archives-ouvertes.fr/hal-00120019

[12] Yildiz, O., Gzara, L., Pernelle, P., Tollenaere, M.: Mda approach for plm system design. In: APMS 2012 International conference - Advances in production Management Systems. Rhodos, Greace (2012)

[13] Yildiz, O., Gzara, L., Pernelle, P., Tollenaere, M.: A framework for plm model design. In: 10th IFIP WG5.1 international conference, PLM 2013. Nantes, France (2013)