A Framework for PLM Model Design

Onur Yildiz[1,3], Philippe Pernelle[2], Lilia Gzara[3], and Michel Tollenaere[3]

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

Abstract. In this article we focus on the implementation of business model in PLM sytems. PLM may be an information system that can natively implement business model without any specific development. This implementation is a crucial step in any project deployment of these information systems. Its also an important step during their evolution. However, the lack of standards and methodologies does not simplify the design and reconfiguration. This paper proposes a modeling approach in PLM systems based on MDA approach. In this context, we propose different levels of modeling (CIM / PIM / PSM) as well as rule mechanisms based on constraints (OCL). This approach is being prototyping by the framework realization based on Eclipse.

1 Introduction

Information system design for industrial enterprises requires finding a good compromise between the standardization of functionalities from the main components (ERP, PLM...) and the specific development. PLM systems are sufficiently generic to propose models adapted to companies specific needs without an additional development. However, the initial implementation of such systems and its multiple reconfigurations are never easy to implement. There are several reasons for this :

- Lack of standard : There is no standard methodology which covers all the scope (from metamodel to PLM implementation). The existing models (or metamodels) [1] [2] [3] [4] [5] [6] [7] are restricted to a context (business domain context, exchange data context, mechanical context...).
- Costs of providing : Enterprises are often required to be assisted by an external consultant (editor or integrators). Furthermore, internal cost (development, configuration, maintenance) [8] is more important than software license cost.
- Consistency control : As example, in the case of an initial deployment, the need to manage CAD data is limited by CAD software constraints. The model design (Part, Sub Assembly, CAD Drawing) will have to be validated by this constraints in order not to create structural inconsistency. Another

A. Bernard, L. Rivest, and D. Dutta (Eds.): PLM 2013, IFIP AICT 409, pp. 159–169, 2013.

example, to adapt itself to their market, companies are often brought to modify their models. Yet, in most cases, the company is not able to identify the impacts of these modifications as problems of consistency or side effects. Currently, companies do not have the tools (or methods) to enable them to generate and modify their PLM business model by ensuring the overall consistent of data and processes.

So, creation or modification steps of the models imply to set up a methodological approach allowing to create or to maintain the system in a structural and functional coherence. In this paper we propose a coherent and global approach for design [9] and adaptation of structured models within PLM systems [10]. This approach is built around an MDA approach (Model Driven Architecture) and based on the design of models under constraints. Indeed, MDE (Model Driven Engineering) [11] allows, by various processing steps and constraint additions, to create one robust data model, with no inconsistency.

In the first section we present MDA concepts and modelling proposals according to MDA levels. We also also specify the typology of constraints applied to these models. In the second section we present a framework that allows the proposed approach as well as an implementation example.

2 MDA Approach for PLM

2.1 From Business Logic to Implementation PLM Platform

MDA is an approach based on MDE (Model Driven Engineering) proposed by the OMG (Object Management Group) [12] [13] which uses the concepts of MOF (Meta Object Facility) [14] and UML (Unified Modeling Language) as modeling tools. This approach is based on several levels of models (or metamodels) and mechanisms of transformation of these models [3]. It enables to make models operational, to transform a source model into another target model of the same system but on a different level. We can identify two types of transformations :

- Endogenous transformation from the same meta-model, the source model and the target are in conformity with the same meta-model. In this case, we can use two transformations : Model-to-model or model-to-code.
- Exogenic transformation starting from different meta-models, the source model and the target are not in conformity with the same meta-model.

In line with MDA, models transformations starts with business concepts and ends with executable code . This is very close to the modeling needs within PLM systems. Indeed, modeling in PLM systems is a recurring issue that should be asked at different steps of a PLM project :

- During the deployment phase of a PLM system in a specific business context: It is to model the objects which are managed in the PLM with the businness concepts in the area of activity of the company.
- During collaborative exchanges between companies: It is to model the combination of semantic concepts used in each company.

– During necessary adaptations to economic context or business: It is to enable the reconfiguration of models while ensuring structural and behavioral consistency of existing model.

The fundamental principle of MDA is the separation between business logic and implementation logic around three levels

– CIM level : It characterizes the design business of the highest level of abstraction independent of any system implementation. It defines the vocabulary resources shared with other models. The technical independence of this model enables it to keep all its interest over time and is changed only if the model of a company needs change.
– PIM level : It is independent of any technical platform and does not contain information on technologies that will be used to deploy the application. PIM models represent a partial view of an CIM and they describe the system, but do not show the details of their use on the platform.
– PSM level : It depends on the used platform (used for code generation). There are several levels of PSM. The first, derived from the transformation of a PIM. The others are obtained by successive transformations. These enable to obtain the code in a specific language. In our context, the PSM level stops at an execution level within a PLM system.

In the following paragraphs, we present models (and metamodel) for a framework based on MDA.

Metamodel for CIM. The purpose of our metamodel (fig. 1) is to define the key concepts that will be used by models (PIM). These concepts should be independent of PLM system. They are used to characterize the compliance models of lower level. The elements proposed in the metamodel allows to identify structural and behavioral invariant concepts that must be used by a PLM system.

Model for PIM. The PIM is a level 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 :

– Concepts treated are very specific for industrial sector and the concepts used involve a certain ambiguity.
– Concepts treated are evolving over time.

Template for PSM. Specific business model identified in the previous level finally have to be to implemented in a PLM platform. The level of implementation is different depending on the systems. In our case, the PIM concepts transformation does not produce executable code. Indeed, our approach is intended to be used in a context of PLM (or reconfiguration definition). the execution platform is implicitly a PLM system and generated elements are in fact elements that will

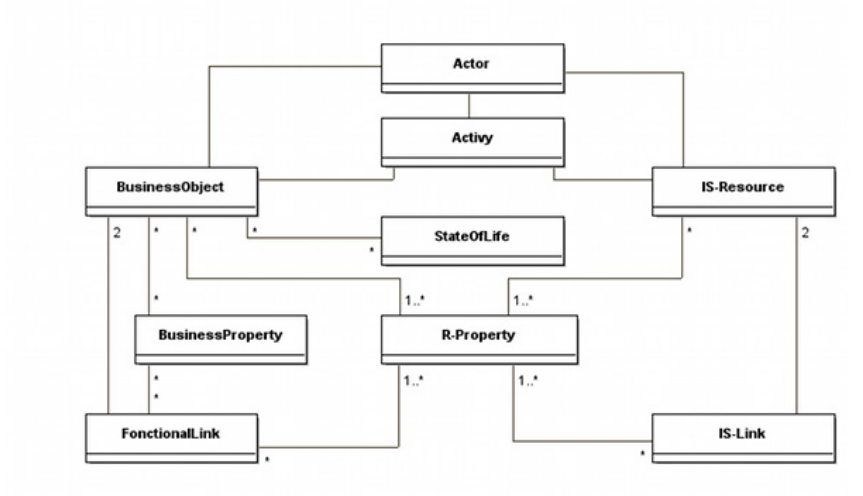

Fig. 1. Metamodel for CIM level

be instantiated in the basics of PLM by the application server. The purpose of PDM (platform description model) is precisely to describe the templates for the models transformation (from PIM to PSM).

2.2 Constraints-Based Approach

Relations (associations, composition...) proposed initially by UML are not enough to characterize business specificities. UML 2 proposes an extension of these relations by the definition of constraints. A constraint represents a boolean-valued expression which can be attached to any UML element. It generally indicates a limitation, or gives further information onto a model. They are used in most cases to specify invariants on the meta classify. An invariant expresses a predicative constraint on an object, or a group objects, which must be permanently respected (regardless its state). So, to define constraints of modeling and according to the expression of constraints in data modeling, it is possible to use :

- Pre-conditions and post-conditions on operations.
- Constraints on the value returned by an operation.
- Derivation rules of attributes.
- Description of targets for messages and actions.
- Conditions in dynamic diagrams.
- Type Invariants for stereotypes (particular to describe UML semantics).

These constraints complete existing diagrams and allow to return more precise relations and without ambiguities. To model constraints on our models, we chose to use OCL constraints (Object Constraint Language) [15]. OCL 2.0 was integrated into the definition of UML 2.0 in 2003. It is in accordance with UML 2 and with the MOF 2.0. Features brought by the integration of constraints OCL are the following ones:

– Model storage with their constraints.
– Syntactic and semantic validation of the constraints
– Code generation to verify the constraints in the execution.

To be able to automate the maximum of checks, it is necessary to define constraints at every level (CIM / PIM / PSM). For that purpose, we propose the following typology of constraints :

Conformity constraints: They have to guarantee the conformity of a business model and PLM models regarding to their higher levels.
Business constraints: These constraints enable to express, on the models, business rules on classes or relations.
Support constraints: These constraints are characteristic to the PLM system implementation. These are constraints on the concepts of PLM (uniqueness, traceability...).
Consistence constraints: These constraints are used to characterize the rules relating to the system dynamics.

This figure (Fig. 2) describes our modeling approach in the PLM systems context.

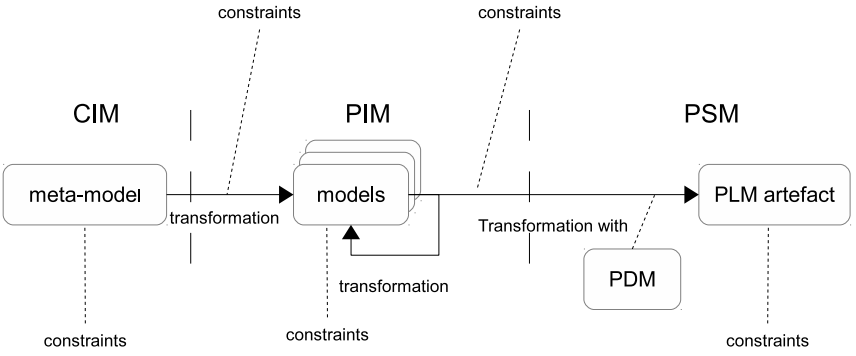

Fig. 2. Global MDA approach for PLM

3 Design of a Framework for PLM

3.1 Description of Framework

Our proposed approach with MDA is supported by a framework which is going to operate transformation models and checks of the constraints. This framework is realised relying on Eclipse Modelling Framework (EMF) [16] [17] [18] based metamodeling facilities. EMF is an implementation of the MOF particularly adapted to the applications based on a model of structured data, which is the case for PLM systems. In the Eclipse environment, EMF enables the implementation of DSL (Domain Specific Language). Furthermore, the genericity of Eclipse plugins allows leaning on EMF to propose inherited corresponding mechanisms in the models presented in section 2. So, we propose designer wizards to help him in his choices. We propose three wizards' levels.

- The first level correspond to the definition of rules of the business concepts. At this level, the designer must identify his main business concepts and the relationships between them. He must also characterize the conformity constraints by associating each of its concepts with elements of the CIM metamodels. This wizard enables to define a first PIM model.
- The second type of wizard enables to enrich or to specialize a PIM model with standard business concepts. For instance, you can complete a model with everyday objects : Documents, CAD objects, Engineering Change Request (ECR), Engineering Change Order (ECO)... This wizard enables to help the designer to produce a PIM model adjusted to his needs.
- The third wizard enables to configure the templates that will produce the elements in the target platform.

Thus, we propose a framework that helps the designer in the process of MDA modeling. The wizards (Fig.3) can not completely characterize the models. The other modeling elements (constraints) are made from a specific view of Eclipse.

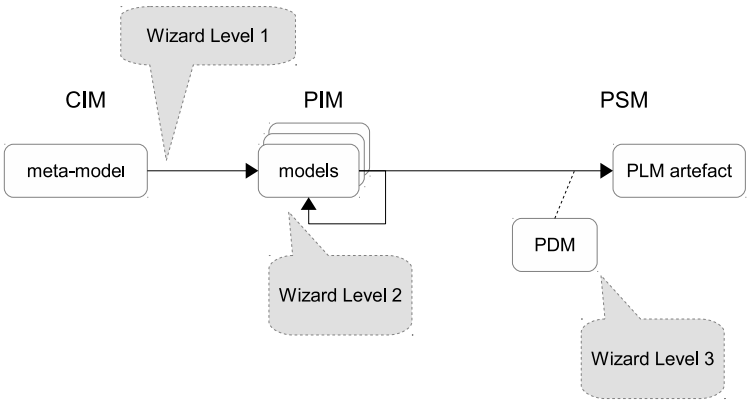

Fig. 3. Wizards for helping designer

In the next section, we present an example of using this approach with the PLM system.

3.2 Industrial Example for SMEs/SMIs

Our industrial example leans on the PLM system Audros. This system is used in of numerous domains by SMEs/SMIs. In this frame, we suggested realizing a version adapted to SMEs/SMIs in the field of the mechanics.

A PIM Model for SMEs/SMIs. The objective of this mechanics model is to propose a minimalist model that works for an SME. It must contain :

- Useful objects, define from the main client data models (part, product, drawing, document, CAD documents, BOM...).
- Standard viewpoints (design, engineering , manufacturing...).
- Standard change management (ECR, ECO...).

For enterprises working with CAD documents, a PIM model is given by the following diagram (Fig. 4). In this model, the metamodel concepts are represented by stereotypes (UML2).

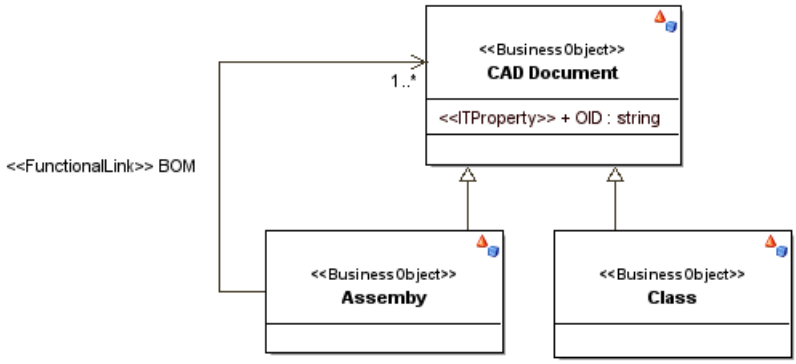

Fig. 4. Extract from PIM model - CAD Document

Another case is the viewpoint notion that is present in all PLM systems and that allows to characterize the access rights on the business objects (Fig.5).

The following diagram (fig. 6) shows an extract from the mechanics's PIM model.

At this level the model is not dependent on the Audros platform, compliance is checked by the metamodel and constraints.

Constraints. The constraints enable to characterize the verification rules at each modeling level. Some constraints are implicit. For instance, the relationships (composition, aggregation...) defined on the metamodel implies to be veriefied in the PIM models to ensure compliance. The explicit rules have been defined on different modeling levels. In table 1, we present some extracts rules defined for the mechanical model in SMEs.

Rule 1. This rule characterizes that in PLM system, every object has a unique reference and its creation is temporalized and is not anonymous.

Rule 2. This rule characterizes a consistence constraint in process management. A process that changes the life cycle of an object must concern at least one object.

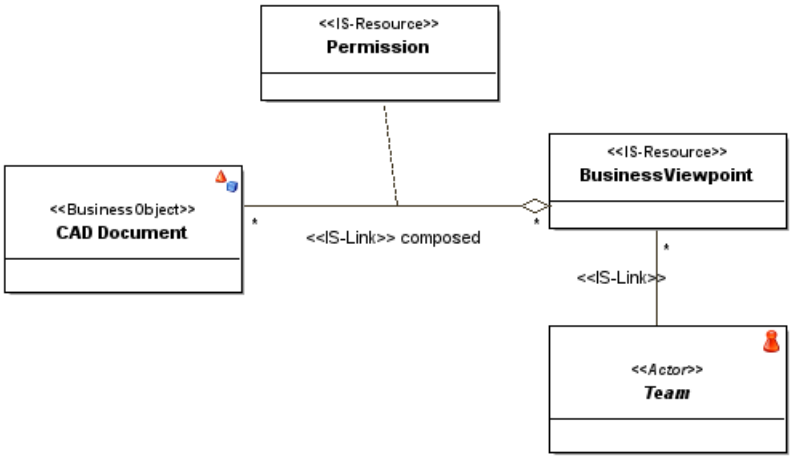

Fig. 5. Extract from PIM model - ViewPoint

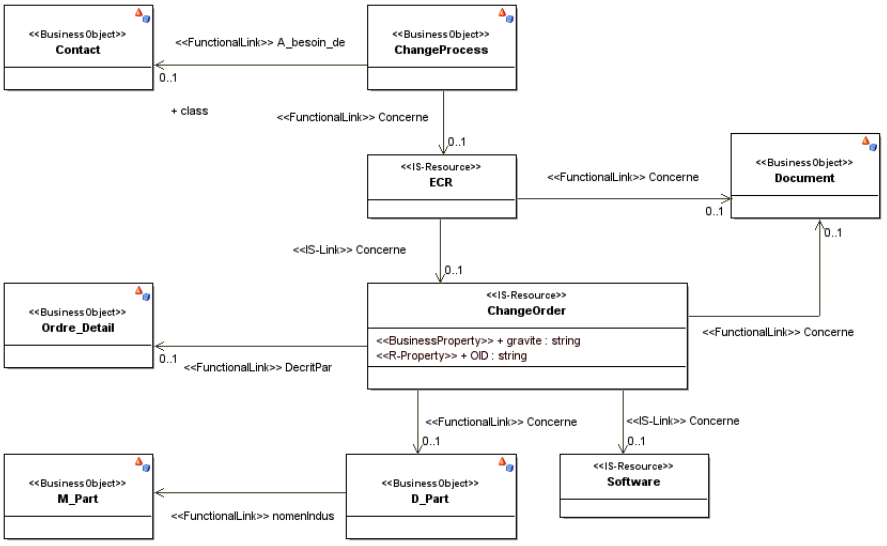

Fig. 6. Extract from PIM model

Table 1. Example of constraints

Type	Ref. Rule	Ref. OCL	Constraint description
Support constraint	rule 1		Any business object must have at least one attribute system.
Consistence constraint	rule 2		Check the existence of a link between a process and at least another object in the data model (otherwise it will be impossible to create a process instance).
Business constraint	rule 3		The BOM link (bom-link type 'BOM ') enables to connect CAD objects. The ends only contain (father, son) : ASM → PRT, ASM → ASM, ASMPRT → ASMPRT, ASM → ASMPRT, ASMPRT → ASM, ASMPRT → PRT.
-		OCL-1.1	The CAD link (cad-link type 'CAD') enables to define dependancies between CAD objects other than BOM link. The ends only contain (father, son) : ASM → ASM, ASM → PRT, PRT → PRT, ASMPRT → ASMPRT, ASM → ASMPRT, ASMPRT → ASM, ASMPRT → PRT. The DRAWING link (draw-link type 'DOC') enables to describe the models referenced by the CAD drawing. The ends only contain (father, son) : DRW → ASM, DRW → PRT, DRW → ASMPRT, DRW → PLT. The DOCBOM link (docBom-link type 'DOCBOM') enables to group different components of BOM. The ends only contain : Same definition that 'DOC' and 'BOM'.

Rule 3. This rule characterizes a business constraint related to the use of a CAD system.

The different examples below show some constraint definitions with OCL.

Listing 1.1. Exemple Constraint OCL-1

```
context c : cad-link FonctionnalLink
inv : c.link_type = 'CAO'
and (c.father.type = 'ASM' or
     c.father.type='ASMPRT' or c.father.type='PRT')
and (c.son.type = 'ASM' or
     c.son.type='PRT' or c.son.type='ASMPRT')
```

4 Conclusion

In this paper a Framework prototype based on Eclipse was proposed which goal is to enable companies to model their business concept in a PLM system. In this

approach, Framework enables to model the main business concepts, according to the different levels. So, the constraint definition can complete the modeling by adding global consistence rules. Thanks to the collaboration with Audros (PLM editor), we built a prototype that implements our metamodel. Although incomplete, we used this metamodel to generate standard modeling elements of a mechanical company. The next step will be the definition of transformation rules to automate model generation.

Acknowledgments. We would like to thank ANRT (Association Nationale de la Recherche et de la Technologie) and Audros Technology for the support in these research.

References

1. Sudarsan, R., Fenves, S., Sriram, R.: A product information modeling framework for product lifecycle management. Computer Aided Design 37(13) (2005)
2. 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)
3. Bach, J.C., Crégut, X., Moreau, P.E., Pantel, M.: Model transformations with tom. In: DTA - 12th Workshop on Language Descriptions, Tools and Applications - 2012, Tallinn, Estonie. ACM (2012)
4. Chettaoui, H.: Interoperabilite entre modeles heterogenes en conception cooperative par des approches d'Ingenierie dirige par les modeles. PhD thesis, Institut polytechnique de Grenoble (2008)
5. Srinivasan, V.: STEP in the context of Product Data Management. In: Advanced Design and Manufacturing Based on STEP. Springer (2009)
6. Le Duigou, J., Bernard, A., Perry, N., Delplace, J.C.: Generic plm system for smes: Application to an equipment manufacturer. International Journal of Product Lifecycle Management, IJPLM (April 2012)
7. Iraqi-Houssaini, M., Kleiner, M., Roucoules, L.: Model-Based (Mechanical) Product Design. In: Whittle, J., Clark, T., Kühne, T. (eds.) MODELS 2011. LNCS, vol. 6981, pp. 548–562. Springer, Heidelberg (2011)
8. Elkadiri, S., Pernelle, P., Delattre, M., Bouras, A.: Collaborative process control: Observation of tracks generated by plm system. In: Proceedings of the International Conference on Advances in Production Management Systems, APMS 2008, Espoo, Finlande (September 2008)
9. Turki, S.: Ingénierie système guidée par les modèles: Application du standard IEEE 15288, de l'architecture MDA et du langage SysML à la conception des systèmes mécatroniques. These, Université du Sud Toulon Var (October 2008)
10. Yildiz, O., Gzara, L., Pernelle, P., Tollenaere, M.: An MDA approach for PLM system design. In: Emmanouilidis, C., Taisch, M., Kiritsis, D. (eds.) APMS 2012, Part II. IFIP AICT, vol. 398, pp. 216–223. Springer, Heidelberg (2013)
11. Bézivin, J.: Model Driven Engineering: An Emerging Technical Space. In: Lämmel, R., Saraiva, J., Visser, J. (eds.) GTTSE 2005. LNCS, vol. 4143, pp. 36–64. Springer, Heidelberg (2006)

12. OMG: OMG model driven architecture (2001), http://www.omg.org/mda/
13. France, R.B., Ghosh, S., Dinh-Trong, T., Solberg, A.: Model-driven development using uml 2.0: Promises and pitfalls. IEEE Computer 39(2) (2006)
14. OMG: Meta object facility (mof) (2011), http://www.omg.org/spec/MOF/2.4.1/
15. OMG (2006), http://www.omg.org/spec/OCL/2.0/
16. Eclipse-Foundation: Eclipse modeling project, http://www.eclipse.org/modeling/
17. Steinberg, D., Budinsky, F., Merks, E., Paternostro, M.: Eclipse Modeling Framework. Pearson Education (2008)
18. Bézivin, J., Kurtev, I.: Model-based technology integration with the technical space concept. In: Metainformatics Symposium 2005, Esbjerg, Denmark (November 2005)