A New Methodology CIM to PIM Transformation Resulting from an Analytical Survey

Yassine Rhazali, Youssef Hadi and Abdelaziz Mouloudi
MISC Laboratory, Faculty of Sciences Kenitra, Ibn Tofail University, Kenitra, Morocco

Keywords: MDA, MDE, CIM to PIM Transformation, Business Process Modelling, Computer Modelling.

Abstract: Our paper shows a new methodology for controlling the models transformation from CIM to PIM into model driven architecture. In this proposal we founded on an analytical survey. Our methodology is based on creation of the transformable models in CIM level to facilitate the transformation task to the PIM level. We create a good PIM level, according to the three classic modelling views. Then, a set of transformation rules are established through ATL language to assure a semi-automatic transformation between CIM and PIM. Our methodology ensures the recommendations of MDA approach by presenting business process in CIM level through BPMN which is OMG standard for modelling business process. However, we founded on UML to model the PIM level, because UML is advisable by MDA in PIM.

1 INTRODUCTION

MDE (Model Driven Engineering) is an alternative approach of software engineering which allows the development of information system, This approach is founded on the creation of source models and transforming them to multiple levels of abstraction until having the source code automatically (Schmidt, 2006). Its objective is automated the process of software development which is followed manually by the information technology professionals. MDE is a generic approach viewed as a family of approaches, where MDA (Model Driven Architecture) (OMG-MDA, 2003) supported by OMG, is considered as the most interesting approach. MDA has the same principles of MDE, but it provides its own bases represented by three levels of abstraction (CIM, PIM, and PSM), exacts the respect of multiple requirements, and recommends the use of some standards.

Model transformation forms the main key in MDA. The transformation from CIM to PIM is the first kind of transformation into MDA that allows deducting PIM models from initial models built in CIM. The objective is to reword information contained in CIM models into PIM models, which assures that the business information will not vanish throughout MDA process. Then, transform PIM to PSM allows adding in PIM models a set of technical information of the target platform.

In practice, automatic transformation begins from PIM level to PSM level. However, our ultimate aim is to make the CIM a productive level, and a basis for building PIM level through an automatic transformation. The objective is that business models do not remain only simple documents of communication between business experts and software designers.

In this paper, we present a solution for automating the transformation from e CIM level to PIM level. In this way, we establish a set of well selected rules for automating the transformation from CIM level to PIM level. According MDA (OMG-MDA, 2015), the CIM level must be presented by business process models. However, we use the BPMN notation to represent CIM level in our methodology, because BPMN is the specialized standard, supported by OMG, for the modelization of business process. The PIM level is presented by information system view; nonetheless, UML is advocated by MDA in the PIM level. Then, we partition PIM level in accordance with the three UML classical views including: functional, static, and dynamic view. Our approach contains one or more models for each modelling view. The use case diagram model, presents the functionalities of the information system; this model shows the functional view. Next, the system states are modelled by state diagram who presents the dynamic view. Then, the model of class diagram presents of the system

Rhazali, Y., Hadi, Y. and Mouloudi, A.
A New Methodology CIM to PIM Transformation Resulting from an Analytical Survey.
DOI: 10.5220/0005690102660273
In *Proceedings of the 4th International Conference on Model-Driven Engineering and Software Development (MODELSWARD 2016)*, pages 266-273
ISBN: 978-989-758-168-7

classes and their relationships which show the static view. To ensure the semi-automatic transformation between CIM and PIM, our approach founded on transformation rules implemented in ATL (Atlas Transformation Language).

The rest of this paper is presented as follows. In section 2 we show the related works concerning transformation between CIM and PIM. In section 3 we present our proposal and transformation rules allow moving from CIM level to PIM level. In section 4 we illustrate our method in a case study demonstrating the transformation between CIM models and PIM models. Finally, in section 5, we conclude by specifying outcome of our work and determining future works.

2 RELATED WORK

In this section, we present the related works of transformation between CIM level and PIM level according MDA.

In (Kherraf et al., 2008) the authors present a method founded on patterns and archetypes to transform the CIM to the PIM. The authors use patterns to establish the CIM level, and apply archetypes to move at the PIM level. This approach is founded on two steps for modeling CIM. The first stage based on activity diagram model and use case diagram model to represent business processes, then, the second stage involves a detailed activity diagram model for modeling system requirements. Then the system components are transformed from the requirement elements as a first step in PIM level. Finally, a collection of four archetypes (Lefebvre, 2005; Coad, 1999) contributes in the transformation from the system components to the class diagram as a final step of the PIM level.

(Zhang et al., 2005) the authors propose a feature-oriented and component-based method, for transforming the CIM to the PIM. The authors use the feature model for structuring requirements in the CIM level. This model contains features and relationships between them. This approach uses software architecture for presenting PIM which contains a set of components and interactions between them. However, responsibilities considered as connectors between features and components to simplify the transformation from CIM to PIM.

An analytical method is presented by (Kardoš et al., 2010) for moving from the CIM models to the PIM models. The authors show business process in CIM with the Data Flow Diagram (DFD) (Qing et al., 2009; Hoffer et al., 2004). However, the PIM level is founded on activity diagram, sequence diagram, use case diagram, and domain model.

Transformation approach from secure business process to use case diagram model is presented in (Rodríguez et al., 2007). This method propose secure business process model, defined with BPMN, into CIM level. However, a collection of transformation rules defined in QVT (Query / View / Transformation) (OMG, 2011), refinement rules, and checklists, allows to obtain the use case diagram model which shows requirement and analysis design in the PIM level. In (Rodríguez et al., 2008; Rodríguez et al., 2010), the authors resume their previous approach and add the UML 2 activity diagram to model secure business model, and class enlarge diagram PIM level.

In (De Castro et al., 2011) the authors propose a transformation method from the CIM level to the PIM level for information system service-oriented development. The authors represent business view in the CIM level through BPMN notation, then they use value model (Gordijn et al., 2003) for specifying services**.** Next, ATL allows to move toward PIM level that represented by two extensions of UML activity diagram and two extensions of UML 2 use case diagram.

A methodology of transformation from model-driven goal-oriented requirement toward data warehouses is shown by (Mazón et al., 2007). The authors represent CIM level through UML profile using the i* modelling framework (Yu, 1997). Nevertheless, the QVT language allows moving to PIM that is represented by data warehouse design.

An approach in (Gutiérrez et al., 2008) is based on automatic generation of activity diagram models from use case diagram model. The authors represent an approach based on QVT to generate a transformation from functional requirements, modeled by use case diagram model in the CIM level, to activity diagram model that define the PIM level.

An approach for transforming process model to Information system model is shown by (Mokrys, 2012). The author based on BPMN to create business process model in CIM level. Nevertheless, PIM level is presented by UML 2 state diagram model and class diagram model.

An approach for transforming a CIM level to a PIM level is represented by (Bousetta et al., 2013). The authors describe the CIM level with high level business model, use case diagram model, and low level business model. Nevertheless, the PIM level is defined by domain class diagram model, and sequence diagram model for system external

behavior. Domain class diagram model is resulted from low business process model by using resource, and business rules and objects.

In (Fatolahi et al., 2008) the authors present a method of web-based applications deducted through a semi-automatic transformation from use cases. This methodology respects the MDA approach. The authors show the CIM level with requirement models described by default domain objects and use cases. Nevertheless, PIM level defined with user interface model, state machine model, and refined domain model.

A method to model user interface according MDA is shown by (Wu et al., 2007). The authors are founded on activity diagram, use case diagram, and robustness diagram for representing user requirements in the CIM. Nevertheless, the PIM level is presented by UML 2 class diagram model and sequence diagram model.

In (Rhazali et al., 2014), we proposed a method to transform business process models to use case diagram model and class diagram model. The approach based on BPMN for modeling business process in CIM. Nevertheless, PIM level is shown by UML 2 use case diagram model and class diagram model.

In (Rhazali et al., 2015) we present a disciplined approach to transform CIM towards PIM. Business process model is described by BPMN and UML 2 activity diagram. Use case diagram model, state diagram model and class diagram model represent PIM level. This method based on a set of transformation rules for moving from CIM to PIM.

In (Rhazali et al., 2015) we established a transformation approach for shifting from CIM to PIM. The approach is founded on BPMN and activity diagram for modeling the business process. The PIM level is presented by state diagram model, class and package diagram model. This approach based on improved rules allows shifting from CIM to PIM through a semi-automatic transformation.

In (Rhazali et al., 2015) we proposed a methodology allows transforming CIM models to PIM models. The approach is based only on UML 2 activity diagram to model business process. However, the PIM level including state diagram model, class diagram model and package diagram model. This method based on a collection of transformation rules for moving from CIM level to PIM level.

3 OUR PROPOSAL

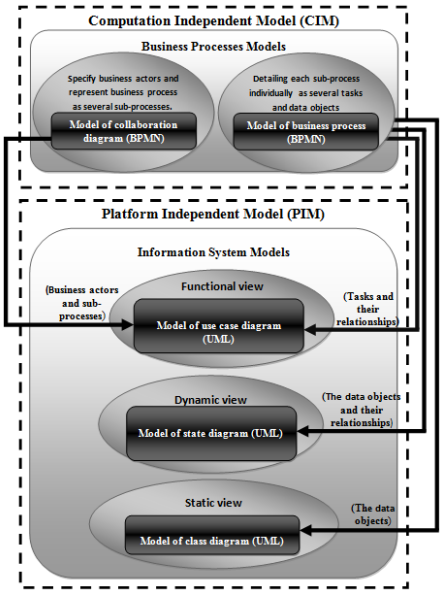

Figure 1: Scheme of our proposal.

According OMG (OMG-MDA, 2015), our proposal considers the business dimension in the CIM level, through the modelization of real business process, in order to keep the knowledge of business throughout the transformation to PIM. This ensures the development of quality information system. However, in (BPMN, 2011), the benefits of most standards of business process modelling converge in BPMN. Thereby, our approach founded on two diagrams of BPMN, collaboration diagram and business process diagram, for modelling the business process in CIM.

MDA recommends the use of UML in the PIM level, according (Blanc, 2005; Kleppe, 2003; Fowler, 2005).Then, in (Roques, 2004; Shin, 2000; Demuth, 1999) the UML diagrams can be divided into three classical modeling views: functional, dynamic, and static. In our approach we based on one UML diagram for modeling each view, thereby, the use case diagram model shows functional view, the state diagram model presents the dynamic view, and the class diagram model interprets the static view.

All PIM models are obtained by a semi-automatic transformation from CIM models (Fig. 1). Transformation is ensured through selected rules implemented in ATL language.

Below we describe construction rules of CIM level, and transformation rules to PIM level. Each transformation rule is described in human language,

and ATL language. Before start the transformation the designer can intervene just for choosing the not transformable elements (actor, task, gatway…). Nevertheless, in the beginning of each rule we verify through ATL language if the element is transformable, for that, in each rule, we call helpers already described with OCL language.

3.1 The Transformation from BPMN Models to Use Case Model

In (Fig. 2) we show the transformation rules which ensure the transformation from BPMN models to use case model.

Transformation rules in human language	Transformation rules in ATL
R1: "gateway xor" between two "tasks" corresponds to relationship "extend" between two "use cases".	`rule R1 { from gwo : MMbpmn!Or (gwo.isTransformableGatewayOr()) to extd : MMusecase!Extend (name <-gwo.name ) }`
R2: Each "sub-process" is transformed to a "package".	`rule R2 { from sbp : MMbpmn!Subprocess (sbp.isTransformableSubprocess()) to clf : MMusecase!Classifier ( name <- sbp.name, containsActor <- sbp.belongsLane ) }`
R3: every "task" corresponds to a system functionality is transformed to a "use case".	`rule R3 { from ts : MMbpmn!Task ((not ts.isManual()) and ts.isTransformableTask()) to uc : MMusecase!Usecase ( name <- ts.name, extendIn <- ts.gatewayOut, extendOut <- ts.gatewayIn, includeIn <- ts.flowOut, includeOut <- ts.flowIn, belongsClassifier <- ts.belongsSubprocess )}`
R4: each "collaborator" becomes an "actor"	`rule R4 { from ln : MMbpmn!Lane (ln.isTransformableLane()) to act : MMusecase!Actor ( name <- ln.name ) }`
R5: "sequence flow" between two "tasks" corresponds to relationship "include" between two "use cases". **R6:** "sequence flow" returning back is not transformable.	`rule R5R6 { from sqc : MMbpmn!SequenceFlow ((not sqc.isReturnBack()) and sqc.isTransformableSequenceFlow() ) to icld : MMusecase!Include (name <- sqc.name ) }`

Figure 2: Transformation rules from BPMN models to use case diagram model.

3.2 Transformation from BPMN to State Diagram Model

In (Fig. 3) we present the transformation rules allow moving from BPMN model towards state diagram model.

3.3 Transformation from BPMN to Class Diagram Model

In (Fig. 4) we show the transformation rules allow shifting BPMN models to class diagram model.

4 CASE STUDY

In this section, we illustrate our transformation methodology from the CIM to the PIM through a case study "booking services".

The rooms' catalogue can be browsed by the customer. He can also present the information about a room available in the catalogue, and then decides to reserve the room or not. In any time, customer can to add, update or delete booking options. Once booking options are well chosen, the customer begins booking while presenting his information, including payment information.

Transformation rules in human language	Transformation rules in ATL
R7: Each data object becomes a state.	`rule R7 { from dto : MMbpmn!DataObject (dto.isTransformableDataObject()) to nrstt: MMstatemachine!NormalState ( name <- dto.name +' '+ dto.state ) }`
R8: Each exclusive fork becomes a decision point.	`rule R8 { from exl : MMbpmn!ForkExclusive (exl.isTransformableForkExclusive()) to dcs : MMstatemachine!DecisionState (name <- exl.name ) }`
R9: Each exclusive join becomes a junction point.	`rule R9 { from exlj : MMbpmn!JoinExclusive (exlj.isTransformableJoinExclusive()) to jct : MMstatemachine!Jonction (name <- exlj.name )}`
R10: Each parallel fork node becomes a fork state.	`rule R10 { from frkp : MMbpmn!ForkParallel (frkp.isTransformableForkParallel()) to frks : MMstatemachine!ForkState (name <- frkp.name )}`
R11: Each parallel join becomes a joint state.	`rule R11 { from jnp : MMbpmn!JoinParallel (jnp.isTransformableJoinParallel()) to jns : MMstatemachine!JointState ( name <- jnp.name )}`
R12: Each parallel joint and fork becomes a joint and fork state.	`rule R12 { from prll : MMbpmn!parallel(prll.isTransformableparallel()) to frkjns : MMstatemachine!ForkandJointState (name <-prll.name )}`
R13: Each exclusive fork and join becomes a junction point.	`rule R13 { from exlsv : MMbpmn!Exclusive(exlsv.isTransformableExclusive()) to jct : MMstatemachine!Jonction (name <- exlsv.name )}`
R14: Each start event is transformed to an initial state.	`rule R14 { from strtevnt : MMbpmn!Start(strtevnt.isTransformableStartEvent()) to intstt: MMstatemachine!InitialState ( name <- strtevnt.name )}`
R15: Each end event becomes a final state.	`rule R15 { from ende : MMbpmn!End (ende.isTransformableEndEvent()) to fnls : MMstatemachine!FinalState (name <- ende.name)}`
R16: Each sequence flow becomes a transition.	`rule R16 { from sqcf : MMbpmn!SequenceFlow (sqcf.isTransformableSequenceFlow()) to trst : MMstatemachine!Transition ( name <- sqcf.name, source <- sqcf.source(), target <- sqcf.target() )}`

Figure 3: Transformation rules from BPMN model to state diagram model.

Transformation rules in human language	Transformation rules in ATL
R17: each "data object" is transformed to a "class".	`rule R17 { from dtobj : MMBPMN!DataObject (dtobj.isTransformableDataObject()) to cls : MMclass!Class ( name <- dtobj.name, operations <- dtobj.stateobject.name )}`
R18: Each "state" of a "data object" becomes a "class method".	`rule R18 { from stt : MMBPMN!StateObject (stt.isTransformableStateObject()) to opr : MMclass!Operation ( name <- stt.name ) }`

Figure 4: Transformation rules from BPMN model to class diagram model.

Declaration of the room reservation with the options specified by the customer is ensured by the

booking agent. Then the room, with specified options, is prepared manually by the maid. However, the butler examines the availability of options and verifies the quality of the room.

4.1 Presentation of the CIM Level

In (Fig. 5) we represent the model of business process through the BPMN collaboration diagram. In this model we show a general business process by representing just the sub-processes and their sequence, and by avoiding identification of tasks and connections between them. Nevertheless, we show the maximum possible of collaborators to represent a real business process, in which several business actors collaborate between them.

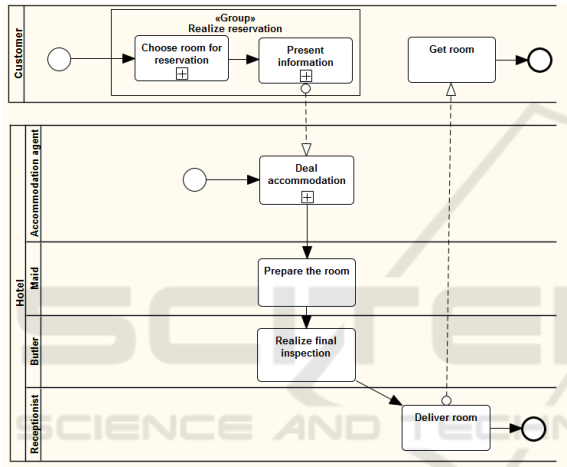

Figure 5: BPMN collaboration diagram model.

The representation of several collaborators facilitates the task of transformation from CIM to PIM. Indeed, when moving from CIM models to the use case diagram model, the collaborators will be transformed to the actors. However, we show averages sub-processes, for example, customer must show the sub-processes "choose room", "start reservation" and "present information", nevertheless the sub-process "start reservation" contain few tasks, for that, we have merged "choose room" and "start reservation" into single sub-process "choose rooms for reservation". Indeed, in this model we must indicate all manual tasks. We can make several refinements into basic models for obtaining transformable models in CIM.

The BPMN business process diagram model (Fig. 6) represents the second model in the CIM level. In this model, we itemized each sub-process into several tasks. Nevertheless, into this model we detailed the sub-process "choose room for reservation" into several tasks with their relationships. Then, in the output of each task we show an object node with its state.

4.2 Presentation of the PIM Level

The (Fig. 7) presents the use case diagram model. This latter model is transformed from the CIM models. Nevertheless, the sub-process “choose room for reservation” becomes a package of use cases. Next, the collaborator "customer" transformed to actor, and the tasks becomes use cases. Decision node that lies between two actions transformed to relationship "extend", then, the control flow which lies between two actions transformed relationship "include". Indeed, in this model, we do not present the flows that return backward. For instance, the relationship which returns from the task "add accommodation options" to "display catalog" is not shown in this model, in order to not complicate the use case diagram model, because this model must concentrates just on the identification of functionalities and not on their logical sequences.

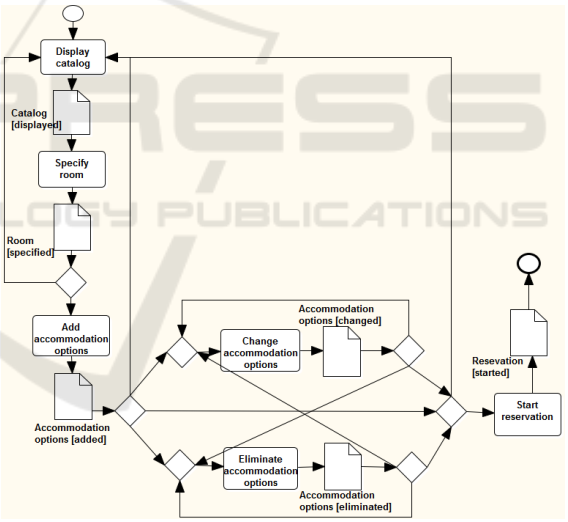

Figure 6: BPMN business process diagram model.

The (Fig. 8) from business process diagram model we obtain the second PIM model, that is the state diagram model. First, the states transformed from object nodes. Next, the "sequence flow" that lies between two tasks is transformed to a transition, e.g. the object node "catalog" with the state "displayed" transformed to state "catalog displayed" in the state diagram model. Nevertheless, the initial state transformed from the start event; the final state becomes from end event; the exclusive fork becomes decision point; exclusive join transformed to

junction point, finally, junction point becomes from exclusive fork & join node.

The establishment of class diagram model (Fig. 9) represents the ultimate objective of the PIM level. This model comes from model of BPMN business process diagram. In class diagram model, the classes are transformed from object nodes. Next, the object states transformed to class methods. Indeed, the object node "reservation" that contains state "started" becomes class "reservation" with method "started".

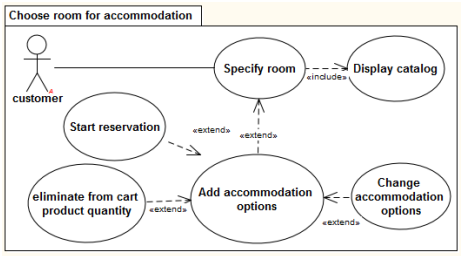

Figure 7: Use case diagram model.

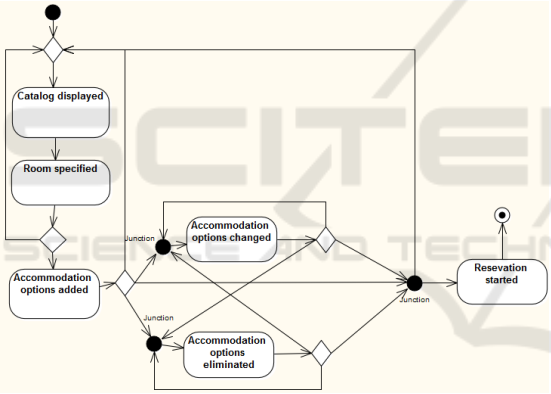

Figure 8: State diagram model.

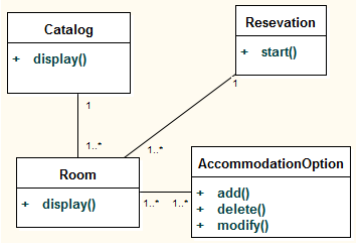

Figure 9: Class diagram model.

5 ANALYSIS AND DISCUSSION

For modelling CIM level, there is a stream that is founded only on system requirement model as in (Gutiérrez et al., 2008; Wu et al., 2007), but CIM level is independent of computation. Next, there is a hybrid current which is based on system requirement and business process for modelling the CIM level like in (Kherraf et al., 2008). In these methods, system requirements are modelled from the beginning into CIM, for facilitating transformation to PIM. In our proposal, business processes are modelled in CIM level. Indeed, according OMG in (OMG-MDA, 2015) the CIM level must be modelled with business processes which are independent of computation. However, in (BPMN, 2011), the benefits of most standards of business process converge to BPMN, for that, we based on BPMN to model business process.

We partition the models of the PIM level in accordance with the three classical modeling views (Roques, 2004; Shin, 2000; Demuth, 1999) includes static, dynamic, and functional. In (Blanc, 2005; Kleppe, 2003; Fowler, 2005) UML is advocated by MDA in the PIM level. Nevertheless, the intersection of our UML 2.0 models and modeling views is presented as follows: the functional view is interpreted by the use case diagram model, the state diagram model presents the dynamic view and the static view is shown by the class diagram model.

There is no method that covers the three modelling views, In the PIM level, except (Kardoš et al., 2010; Rhazali et al., 2015). However, various methodologies do not model the classes in the PIM like in (Zhang et al., 2005; Rodríguez et al., 2007; Castro et al., 2011; Mazón et al., 2007; Gutiérrez et al., 2008; Wu et al., 2007), although without classes the code does not readily results by transformation.

Our approach covers the three modelling views. However, in static view we based on class diagram model.

The rules of model transformation may be defined by human language that has less value, indeed, algorithm and programming language are more technical, however, model transformation languages are the most effective.

Most approaches describes the transformation with a human language as (Kherraf et al., 2008; Kardoš et al., 2010; Mokrys et al., 2012; Bousetta et al., 2013; Wu et al., 2007; Rhazali et al., 2014; Rhazali et al., 2015), There is one method (Zhang et al., 2005) founded on algorithm to define transformation, but transformation must be described by a transformation language as in (Rodríguez et al., 2007; Rodríguez et al., 2008; Rodríguez et al., 2010; Castro et al., 2011; Mazón et al., 2007; Gutiérrez et al., 2008).

There are several manners that approve the transformation approaches from CIM to PIM. We

can approve our approach by a theoretical case study, by a practical case, or by a developed tool which allows ensuring a model transformation.

Most transformation approaches, approves their methodology through a theoretical case study as (Kherraf et al., 2008; Zhang et al., 2005; Kardoš et al., 2010; Rodríguez et al., 2007; Rodríguez et al., 2008; Rodríguez et al., 2010; Mazón et al., 2007; Gutiérrez et al., 2008 Mokrys et al., 2012; Bousetta et al., 2013; Rhazali et al., 2014; Rhazali et al., 2015). In practice, there is one approach (Castro et al., 2011), based on Eclipse tool to implement transformation. Then one approach (Wu et al., 2007) does not approve his methodology. However, our approach is approved in practice through Eclipse.

6 CONCLUSIONS

One of the principal challenges in software development process is the foundation of a approach that allows shifting, semi-automatically, from models of the business process to models that present the design of software. Based on MDA, our approach proposes a solution to the problem of transformation from business models (CIM level) to the design models (PIM level). Our approach provides a set of useful classes in the process of software development. However, in the ongoing work we define a transformation from the PIM models to PSM models. Nevertheless, in our future work we plan to construct a transformation tool founded on MDA principles, indeed, our goal is to obtain the code from the business models by means of semi-automatic transformations.

REFERENCES

Blanc, X., 2005. *MDA in action*, Ed. Eyrolles. 1st edition.

Bousetta, B., El Beggar, O., Gadi, T., 2013, *A methodology for CIM modeling and its transformation to PIM*, Journal of Information Engineering and Applications, vol. 3, no. 2, pp. 1-21.

BPMN, 2011, Business Process Model and Notation (BPMN)-Version 2.0, In *OMG, http://www.omg.org/spec/BPMN/2.0, January 2011.*

Clark, J., Casanave, C., Kanaskie, K., Harvey, B., Clark, J., Smith, N., Yunker, J., and Riemer, K.: ebXML Business Process Specification Schema Version 1.01. *UN/CEFACT and OASIS. 2001.*

Coad, P., Lefebvre, E., De Luca, J., 1999, Java Modeling In Color With UML: Enterprise Components and Process, *Textbook Binding*.

De Castro, V., Marcos, E., Vara, J. M., 2011. Applying CIM-to-PIM model transformations for the service-oriented development of information systems, *Journal of Information and Software Technology 53, pp. 87-105.*

Demuth, B., Hussmann, H., 1999, Using OCL Constraints for Relational Database Design. In *UML'99 The Unified Modeling Language, Second Int. Conference Fort Collins, CO, USA, October 1999, Springer.*

Fatolahi, A., Somé, S.S., Lethbridge, T.C., 2008, Towards a semi-automated model-driven method for the generation of web-based applications from use cases, In *4th Model Driven Web Engineering Workshop, page 31, Toulouse, France.*

Fowler, M., 2005, Language Workbenches and Model Driven Architecture, http://www.martinfowler.com/ articles /mdaLanguageWorkbench.html.

Giaglis, G., 2001, A Taxonomy of Business Process Modeling and Information Systems Modeling Techniques.

Gordijn, J., Akkermans, J. M., 2003, Value based requirements engineering: exploring innovative e-commerce idea, *Requirements Engineering Journal 8 (2) 114–134.*

Gutiérrez, J. J., Nebut, C., Escalona, M. J., Mejías, M., Ramos, I.M., 2008, Visualization of use cases through automatically generated activity diagrams, In *Proceedings of the 11th International Conference MoDELS'08, Toulouse, France.*

Hoffer, J. A, George, J. F, Valacich, J. S., 2004, Modern system analysis and design. Prentice Hall ISBN 0-13-145461-7, 2004.

Kardoš, M., Drozdová, M., 2010, Analytical method of CIM to PIM transformation in Model Driven Architecture (MDA), *Journal of information and organizational sciences, vol. 34, pp. 89-99.*

Kherraf, S., Lefebvre, É., Suryn, W., 2008. Transformation from CIM to PIM using patterns and Archetypes, In *ASWEC'08, 19th Australian Software Engineering Conference, Perth, Australia.*

Kleppe, A., Warmer, G. J., Bast, W., 2003, MDA Explained: The Model Driven Architecture: Practice and Promise. *Addison-Wesley.*

Kriouile, A., Gadi, T., Balouki, Y., 2013, CIM to PIM Transformation: A criteria Based Evaluation, *International Journal Computer Technology & Applications, vol. 4, no. 4, pp. 616-625.*

Lefebvre, E., 2005, Building Platform-Independent Models with Business Archetypes and Patterns, *Montreal Conference on eTechnologies.*

Li, Q., Chen, Y. L., 2009, Modeling and Analysis of Enterprise and Information Systems. *Beijing : Higher Education Press.*

Mayer, R., Menzel, C., Painter, M., Perakath, B., de Witte P. and Blinn T. Information Integration For Concurrent Engineering (IICE) - IDEF3 Process Description Capture Method Report. *Technical Report September 1995 http://www.idef.com/pdf/idef3_fn.pdf.*

Mazón, J., Pardillo, J., Trujillo, J., 2007, A model-driven goal-oriented requirement engineering approach for data warehouses, In *Proceedings of the Conference on*

Advances in Conceptual Modeling: Foundations and Applications, ER Workshops, Auckland, New Zealand, pp. 255–264.

Mokrys, M., 2012, Possible transformation from Process Model to IS Design Model, In *the 1th International Virtual Conference Slovakia, pp. 71–74.*

OMG, MOF 2.0 Query/View/Transformation (QVT), 2011, V1.0. OMG Document – formal/2011-01-01. *<http://www.omg.org/spec/QVT/1.1/PDF/>*.

OMG-MDA, 2003, "MDA Guide Version 1.0.1," 1 juin 2003.

OMG-MDA, 2015a, Model Driven Architecture (MDA) FAQ, *http://www.omg.org/mda/faq_mda.htm.*

OMG-MDA, 2015b, MDA Guide revision 2.0, *ormsc/14-06-01. http://www.omg.org/cgi-bin/doc?ormsc/14-06-01.pdf.*

OMG-SoaML, 2012, Service Oriented Architecture Modeling Language (SoaML) – Specification for the UML Profile and Metamodel for Services (UPMS). *OMG document: ad/2012-05-10. <http://www.omg.org/spec/SoaML/1.0.1/PDF>*.

OMG-UML, 2011, OMG Unified Modeling Language™ (OMG-UML), Infrastructure, *http://www.omg.org /spec /UML/2.4.1/Infrastructure," August 2011.*

Qing, L., Yu-Liu, C., 2009. *Modeling and Analysis of Enterprise and Information Systems*, Springer Publishing Company. Beijing, 1st edition.

Rhazali, Y., Hadi, Y., Mouloudi, A., 2014, Transformation Method CIM to PIM: From Business Processes Models Defined in BPMN to Use Case and Class Models Defined in UML, In *16th International Conference on Model Transformation, Kuala Lumpur, Malaysia, pp. 1374–1378.*

Rhazali, Y., Hadi, Y., Mouloudi, A., 2015, Disciplined Approach for Transformation CIM to PIM in MDA, In *MODELSWARD'15, the 3rd International Conference on Model-Driven Engineering and Software Development, Angers, France, pp. 312 - 320.*

Rhazali, Y., Hadi, Y., Mouloudi, A., 2015, A Methodology for Transforming CIM to PIM through UML: From Business View to Information System View, In *WCCS15, Third World Conference on Complex Systems, Marrakech, Morocco.*

Rhazali, Y., Hadi, Y., Mouloudi, A., 2015, Transformation Approach CIM to PIM: From Business Processes Models to State Machine and Package Models, In *OSSCOM 2015, the 1st International Conference on Open Source Software Computing, Amman, Jordan.*

Rodríguez, A., Fernández-Medina, E., Piattini, M., 2007. *Towards CIM to PIM transformation: from Secure Business Processes defined in BPMN to Use-Cases*, *Business Process Management, vol. 4714, pp. 408-415.*

Rodríguez, A., Fernández-Medina, E., Piattini, M., 2008. *CIM to PIM Transformation: A Reality*, in *Research and Practical Issues of Enterprise Information Systems II, IFIP International Federation for Information Processing, Volume 255/2008. Vyd.: Springer Berlin / Heidelberg, strany 1239-1249.*

Rodríguez, A., García-Rodríguez de Guzmán, I., Fernández Medina, E., Piattini, M., 2010. *Semi-formal transformation of secure business processes into analysis class and use case models: an MDA approach*, Information and Software Technology 52 (9) (2010) 945–971.

Roques, P., 2004. *UML in Practice: The Art of Modeling Software Systems Demonstrated through Worked Examples and Solutions*, Wiley. 1st edition.

Schmidt, D. C., 2006. *Model-Driven Engineering*, IEEE Computer. vol.39, no. 2, pp. 25-31, February 2006, doi:10.1109/MC.2006.58.

Workflow Management Coalition. XPDL, Welcome to XPDL.org, *http://www.xpdl.org.*

Wu, J. H., Shin, S. S., Chien, J. L., Chao, W. S., Hsieh, M.C., 2007, An extended MDA method for user interface modeling and transformation, In *the 15th European Conference on Information Systems (pp. 1632-1641).*

Yu, E., 1997, Towards modeling and reasoning support for early-phase requirements engineering, In *Proceedings of the 3rd IEEE International Symposium on Requirements Engineering. RE, Washington, DC, USA, pp. 226–235.*

Yue, T., Briand, L., Labiche, Y., 2011, A Systematic Review of Transformation Approaches between User Requirements and Analysis Models, *Requirements Engineering (Springer), pp. 75-99.*

Zhang,W., Mei, H., Zhao, H., Yang, J., 2005, Transformation from CIM to PIM: A Feature-Oriented Component-Based approach, In *MoDELS, Montego Bay, Jamaica.*