Open Work of Two-Hemisphere Model Transformation Definition into UML Class Diagram in the Context of MDA

Oksana Nikiforova[1] and Natalja Pavlova[1]

[1]Department of Applied Computer Science, Riga Technical University, Riga, Latvia,
{oksana.nikiforova, natalja.pavlova}@rtu.lv

Abstract. Model Driven Architecture (MDA) is based on models and distinguish between a system functionality specification and this specification realization on a given technological platform. MDA consists of four models: CIM (Computation Independent Model), PIM (Platform Independent Model), PSM (Platform Specific Model) and code model, all these are parts of the MDA transformation line: CIM->PIM->PSM->code. A PIM model has to be created using a language which is able to describe a system from various points of view, system behavior, system's business objects, system actors, system use cases and so on. Current paper discusses the application of two-hemisphere model for construction of UML class diagram as a part of PIM. Several solutions for determination of elements of class diagram from two-hemisphere model are currently researched and described in the paper. As well as application of the transformations by example of insurance problem domain are presented in the paper.

Keywords: Business process diagram, class diagram, MDA, PIM, relationships among classes.

1 Introduction

Model Driven Architecture (MDA) is a framework being built under supervision of the Object Modeling Group (OMG) [1]. MDA separates system business aspects from system implementation aspects. MDA defines the approach and tool requirements to specification of systems independently of platforms, specification of platforms, choosing of particular platforms to the systems, and transformation of specifications of business domains into specifications that include specific information of platforms, which have been chosen. MDA proposes a software development process in which the key notions are models and model transformations [2]. Software is built by constructing one or more models, and transforming them into other models in this process. The common view on this process is as follows: input is platform independent models and output is platform specific models. The platform specific models can be easily transformed into an executable format [3].

There is a more generic view in Model Driven Architecture [4]. A difference between platform independent models and platform specific models is not dominant in this case. The key of this view is that the software development process is implemented by an intricate sequence of transformation executions that are combined in various ways. This makes system development much more open and flexible [3].

The MDA idea is promising – raising the level of abstraction, on which systems are developed. Therefore, it will be possible to develop systems that are more complex in a qualitative way. The basic goal in recent researches in the area of MDA is to achieve a system representation, which corresponds to business requirements, at the highest level of abstraction as possible. Nowadays, MDA tools support formalization of transformations between PIM and PSM stages, and researchers try to "raise" it up as high as possible to fulfill the main statement of MDA [5], [6]. The paper shows state of the art of two-hemisphere model application to generation of elements of class diagram and presents several solutions which address the problems stated above and try to "raise" up the level of transformations into the transformations inside PIM.

Section 2 describes two-hemisphere model and define main components of it suitable for generation of elements of UML class diagram. As well as structure of source and target models is defined in Section 2. Section 3 defines all possible transformations from elements of two-hemisphere model into elements of UML class diagram, which is defined as a target. A practical experiment with the transformations from two-hemisphere model into class diagram is processed in Section 4. Overall statements of the research are discussed in conclusions.

2 Model Transformations in Terms of Two-Hemisphere Model

Two-hemisphere model driven approach [7] proposes to use of business process modeling and concept modeling to represent systems in the platform independent manner and describes how to transform business process models into several elements of UML diagrams. For the first time the strategy was proposed in [8], where the general framework for object-oriented software development had been presented and the idea about usage of two interrelated models for software system development has been stated and discussed.

Two-hemisphere approach proposes to start software development process based on two-hemisphere problem domain model, where one model reflects functional (procedural) aspects of the business and software system, and another model reflects corresponding concept structures. The co-existence and inter-relatedness of models enables use of knowledge transfer from one model to another, as well as utilization of particular knowledge completeness and consistency checks [7].

MDA introduces an approach to system specification that separates the views on three different layers of abstraction: high level specification of what the system is expected to do (Computation Independent Model or CIM); the specification of system functionality (Platform Independent Model or PIM); and the specification

of the implementation of that functionality on a specific technology platform (Platform Specific Model or PSM).

Currently available methods do not support formal transformation from CIM to PIM, wherein the PIM would be sufficient for PSM generation [5]. The PIM received from the CIM should be refined in order to get the correct transformation into the PSM. Moreover, investigation of the PIM requires a more detailed description of it. That is why it was decided to divide the solution domain into two above-mentioned levels: the first one closer to the CIM and the second one closer to the PSM. As previously mentioned, the second level of the solution domain is an application level. Because this level is closer to the PSM, models with further automation should be represented here. A model on the application level of the solution domain is class diagram. A model on the application level should be sufficient for formal transformation to the Platform Specific Model. Therefore, data structure, attributes, system functions and main algorithms should be presented in the PIM ready to transformation into PSM [5].

CIM presents specification of the system at problem domain level and can be transformed into elements of PIM. PIM provides formal specification of the system structure and functions that abstracts from technical details, and thus presents solution aspects of the system to be developed, which enables model transformation to the platform level (PSM), named implementation domain in Figure 1.

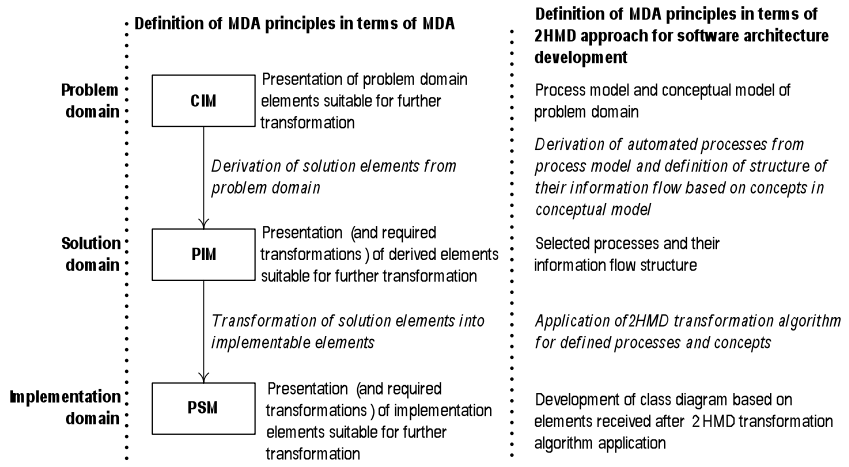


Fig. 1. Model transformation from problem domain level of knowledge representation into implementation domain level according 2HMD approach.

The details in the right column of the table in Figure 1 correspond to the two-hemisphere approach, which addresses the construction of information about problem domain by use of two interrelated models at problem domain level, namely, the process model and the conceptual model. The conceptual model is used in parallel with process model to cross-examine software developers understanding of procedural and semantic aspects of problem domain.

The main idea of MDA is to achieve formal system representation at the as high level of abstraction as possible. One of the most important and problematic stages in MDA realization is derivation of PIM elements from a problem domain, and PIM construction in the form that is suitable for the PSM [5]. It is necessary to find the way to develop PIM using formal representation, so far keeping the level of abstraction high enough. PIM model should represent system static and dynamic aspects. Class diagram shows static structure of the developed system and is the central component of PIM. But UML is a modeling language and does not have all the possibilities to specify context and the way of modeling, which is required always to be defined in a methodology. Therefore the construction of class diagram has to be based on well defined rules for its elements generation from the problem domain model presented in the form suitable for that.

The MDA framework implies system development based on modeling, not on programming activities. System development is divided into three stages according to the level of abstraction. Every stage is denoted with a model. The model is often presented as a combination of drawings and text [1].

A transformation tool or approach takes a model on input and creates another model on output, see Figure 2 [4]. The two-hemisphere model has been marked as input with mapping rules, the class diagram and transformation trace has been received on output. Transformation trace shows the plan how an element of the two-hemisphere model is transformed into the corresponding element of the class diagram, and which parts of the mapping are used for transformation of every part of the two-hemisphere model [1]. Figure 2 shows how a transformation tool takes input – the two-hemisphere model and receives output – the class diagram.

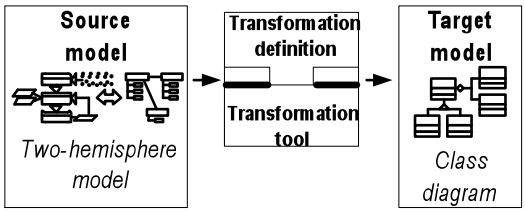


Fig. 2. Structure of model transformation tool in the framework of MDA.

All elements of the source model are shown in Table 1. It is elements of the business process model and concept model. A notation of the business process model is optional, however, it must reflect the following components of business process model: processes; performers; information flows; and information (data) stores [7]. Real-world classes relevant to the problem domain and their relationships are presented in concepts model. It is a variation of well known ER diagram notation [9] and consists of concepts (i.e. entities or objects) and their attributes. The notational conventions of the business process diagram give a possibility to address concepts in concept model to information flows (e.g. events) in process model. The elements of the source model are listed in the first column of Table 1. The second column describes the main elements of the source model.

Table 1. Elements of source model.

Elements of Source model	Description
Business process diagram/ Process process name description triggering condition performer expression duration start option end option no start option tag assignment	Business Process usually means a chain of tasks that produce a result which is valuable to some hypothetical customer. A business process is a gradually refined description of a business activity (task) [10].
Business process diagram/Event name transfer name set option repeat option	Events are defined (as a rule) in the moment when they are mentioned for the first time in BP or TD diagrams. Events are an input/output object (or more precisely - the arrival of an input object and departure of an output object) of certain business process. These objects can be material things or just information [10].
BP diagram/Data store store name comment ER model<ER name>	The data store is a persistent (independent of the current task) storage of data or materials. In the case of an information system, the data store most likely will be converted to a database with a certain data structure (Entity Relationship Model). On the highest levels of business models, the data store can be used to denote an archive of data or it can also be used to represent a warehouse or stock of goods [10].
Concept model/Concept name	Conceptual classes that are software (analysis) class candidates in essence. A conceptual class is an idea, thing, or object. A conceptual class may be considered in terms of its symbols – words or images, intensions – definitions, and extensions – the set of examples [11].
Concept model/Attribute name type	An attribute is a logical data value of an object [11].

The elements of source model are important for further system analysis and design. Information from these elements is significant, and nothing from it should remain without any usage on the lowest levels of abstraction. Business processes

or tasks present system functionality, its activities and operations. If this information is lost it is necessary to find system functions for implementing operations of classes in description of the problem domain. Events, data stores and data objects are parts of the data structure model. With these elements initial static structure is presented in the two-hemisphere model, and should be transited into the class diagram. Associations and attributes in the concept model are useful for definition of relationships between objects of system static structure (classes).

The elements of the target model are listed in Table 2. Only the main elements of the class diagram are shown there.

Table 2. Elements of target model.

Element of Source model	Description
Class diagram/Class	A class is the descriptor for a set of objects with similar structure, behavior, and relationships [11].
Class diagram/Class/Attribute name type	An attribute is a logical data value of an object [11].
Class diagram/Class/Operation name return type argument precondition postcondition	The UML formally defines operations. To quote: "An operation is a specification of a transformation or query that an object may be called to execute" [12].
Class diagram/Relationship type multiplicity role	A relationship between instances of the two classes. There is an association between two classes if an instance of one class must know about the other in order to perform its work [11].
Class diagram/Class/Stereotype	Stereotypes, which provide a way of extending UML, are new kinds of model elements created from existing kinds [11].
Class diagram/Constraint	A constraint is a condition that every implementation of the design must satisfy [11].

It is necessary to find the way how source model elements can be transformed into target model elements according to the definition of transformations in the framework of MDA.

3 Application of Two-Hemisphere model for obtaining of elements of class diagram

For the research the source model is two-hemisphere model, or business process and concept diagrams, and the target model is class diagram in terms of UML class diagram. The possible combinations of transition from two-hemisphere

model to class diagram will be discussed in this section. Input and output is necessary for any transformation. The transformation discussed here is a transformation inside the PIM – one of the MDA models. In other words, it is a transformation from two-hemisphere model to class diagram.

3.1 General Schema of Transformation Abilities

The detailed transformations between models proposed for the application of two-hemisphere model in the paper are shown in Figure 3. Two-hemisphere model consists of business process model (graph G1 on Figure 3) and concept model (graph G2 on Figure 3).

The notation of business process model have not a significant value, main requirement to the notation of business process model is possibility to define business processes, performers, events and data flows among business processes. For current research is used business process model constructed with GRAPES [10] notation. The second hemisphere is concept model. C.Larman defines concept model as "The concept model captures real-world concepts (objects), their attributes, and the associations between these concepts." [11].

In the case of two-hemisphere model authors avoid relations between classes in concept model at business level (of problem domain) and the relations will be defined according system realization at software level (of implementation domain).

For performing of transformation to class diagram the intermediate model (graph G3 on Figure 3) is introduced. Intermediate model is used to simplify the transition between business process and object interaction models, which now is presented in the form of UML collaboration diagram (graph G4 in Figure 3). Figure 3 shows all the transformations from the business process model (G1) and concept model (G2) into the class diagram (G5). Transformations are based on the hypothesis that elements of the class diagram can be received from the two-hemisphere model by applying defined techniques of graph transformation [13].

Intermediate model is generated from business process model using methods of directed graph transformation, when arcs of one graph (G1 on Figure 3) are transformed into nodes of another graph (G3 on Figure 3) and nodes of one graph (G1) are transformed into arcs of another graph (G2) [14]. Figure 3 presents the sequence of transformations from two-hemisphere model to class diagram with dotted arrows. Business process "perform action 1" is transformed into arc "perform action 1" of intermediate model (graph G3 on Figure 3). The next transformation create the method "perform action 1()" in collaboration diagram (graph G4 on Figure 3) from the arc of intermediate model. The last transformation of this business process defines the responsible class of this method in class diagram (graph G5). The element "performer 1" is transformed as a node of intermediate model, and as "actor 1" of collaboration model. This element is defined as "actor 1" in class diagram. Data types for elements "event 1" and "event 3" is defined as "DataType A" or "Concept A" of concept model.

Events are transformed into nodes of intermediate model, and then into objects like "Event1: Class A" in collaboration diagram, which serves as a base for classes

of class diagram definition. All attributes for classes are determined based on attributes defined in concept model.

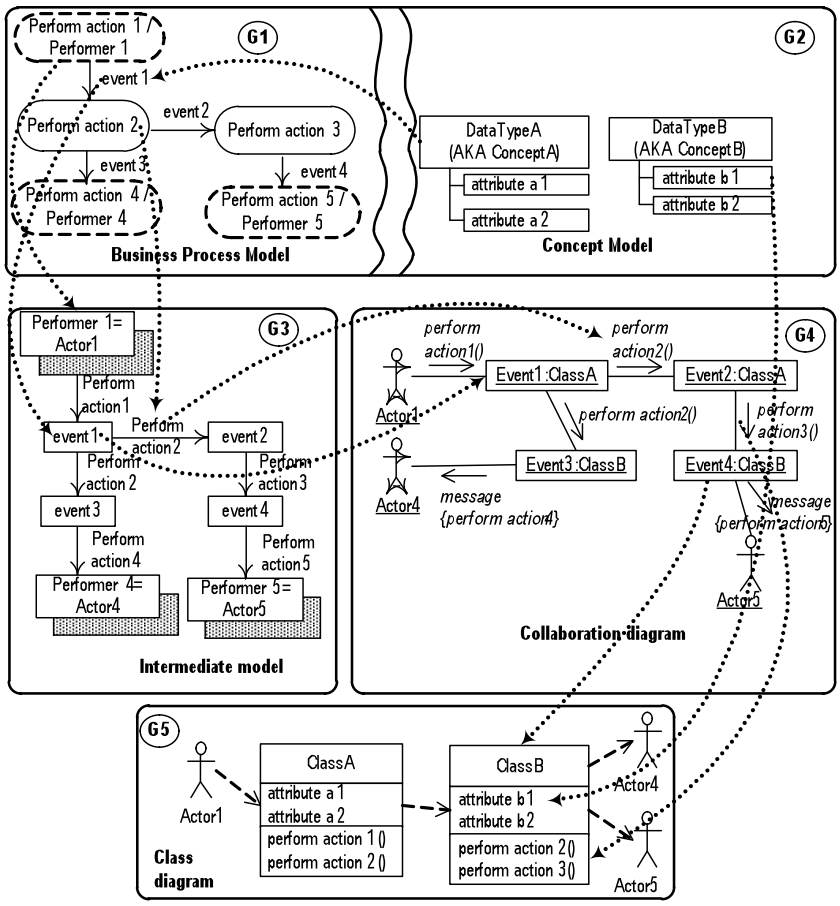


Fig. 3. Essence of application of two-hemisphere model for generation of elements of class diagram.

Figure 3 presents how elements of two-hemisphere model are transformed into elements of class diagram. The ways of receiving of the following elements are shown with arrows in Figure 3:

- Business process “perform action 1” is transformed into arc “perform action 1” of intermediate model. The next transformation create the method “perform action 1()” in collaboration diagram from the arc of intermediate model. The last transformation of this business process defines the responsible class of this method in class diagram.

- The element “performer 1” is transformed as a node of intermediate model, and as “actor 1” of collaboration model. This element is defined as “actor 1” in class diagram.
- Data types for elements “event 1” and “event 3” is defined as “DataType A” or “Concept A” of concept model. Events are transformed into nodes of intermediate model, and then into objects like “Event1: Class A” in collaboration diagram, which serves as a base for classes of class diagram definition.
- All attributes for classes are determined based on attributes defined in concept model.

Summarization of mapping of source model into target of model is shown in Figure 4.

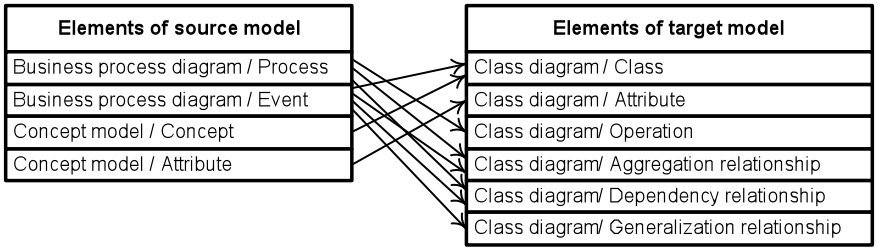


Fig. 4. Mapping of elements of source model into elements of target model.

As it is seen in Figure 4 not all the elements of class diagram defined in Table 2 are received from two-hemisphere model: stereotype and constraint are still under research. But the main components of class diagram are generated from different elements or their combinations of two-hemisphere model: classes, operations, methods and different types of relationships between classes. An illustrative example of definition of classes and their attributes and operations is described in Section 3.2. and transformation rules for definition of different types of relationships between classes in details are discusses in [15]. And not all the elements of two-hemisphere model are used for identification of elements of class diagram: data stores are avoided due to it duplication by element “task” with different meaning.

3.2 An illustrative example of class diagram generation from two-hemisphere model

For better understanding of main idea, the example of such model transformations is shown for a fragment of problem domain concerned with insurance activities [16]. Figure 5 presents only fragment of transformation. There is one process “Pay sum” which has output “policy”. Concept “policy” defines data type for the output of process. It is transformed into fragment of intermediate model with arc “Pay sum” and node “Policy”. Intermediate model allows to receive collaboration diagram, where initial process “Pay sum” is a method of object “Policy”.

Concrete object “Policy” belongs to class “Policy”, which is defined with corresponding concept. When a collaboration of objects is defined, it is possible to construct class diagram according to rules of object-oriented system modeling [8].

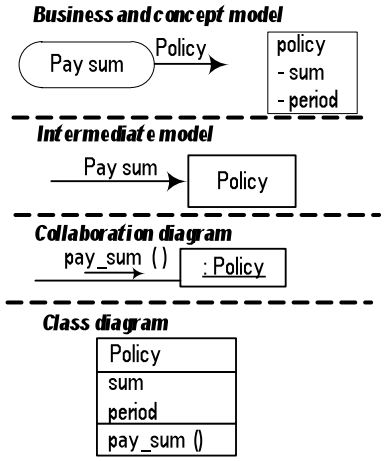


Fig. 4. An example of process and concept elements transformation into class elements.

4 Practical Experiment with the Processing of Transformations from Two-Hemisphere Model into Class Diagram

During the investigation of receiving of class diagram from two-hemisphere model all possible combination of number and types of incoming and outgoing information flows from nodes of processes are examined [8]. Different combinations give a possibility to receive different relations among classes.

The tool for business process modeling GRADE (GRADE) [17] gives a possibility to construct two interrelated models (business process and the concept ones) and to generate text description of models with permanent structure, therefore it is chosen as a tool for development of two-hemisphere model and further generation of textual files, which defines all the elements of the model and their relations each to other. Generated text files serve as an input information to support the processing algorithm of the transformations among graphs defined in the Section 3. And as the result the XML file, which contains description of structure of the class diagram generated from the source model. XML format of class specification gives a possibility to receive visual representation of class diagram in any tool, which support import from XML for class diagram development.

To check, that offered transformations are independent from problem domain an experiment with two-hemisphere model of insurance is performed. The transformations are applied for generation of class diagram from two-hemisphere mod-

model developed for insurance problem domain (shown in Figure 6) and the result class diagram is shown in Figure 7.

As far as business process and conceptual models are the built-in demo example for system development in GRADE the authors may suppose that the models (e.g. source information) are correct and constructed independently from author participation. Therefore it is possible to address the truthful verification of an experiment.

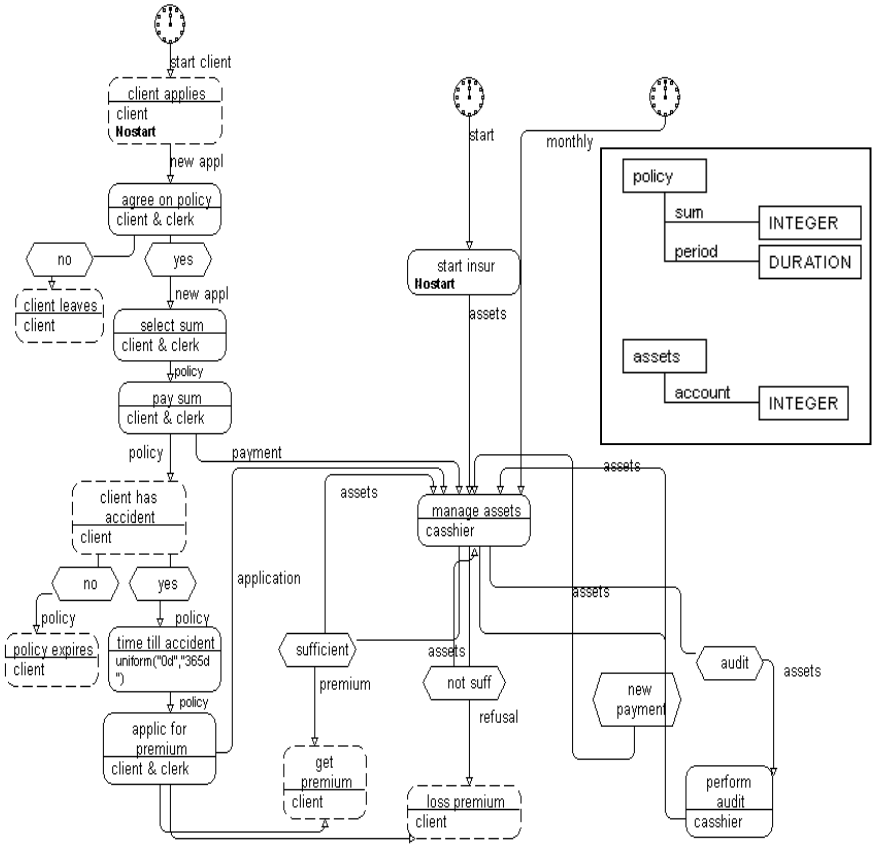


Fig. 6. Initial business process and concept models for Insurance problem domain.

The class diagram in Figure 7 has undefined relations, and unrelated classes, for which additional, detailed, business process models are required. Classes, which are highlighted as gray in Figure 7 are defined as classes which have a restriction. For current level of details it is impossible to define relations of this classes and belonging of method “manage_assets ()” without creating a subprocess diagram for corresponding fragment of business process. After the de-

tailed elaboration of the process it is possible to apply transformations one more time and receive more correct relations among classes.

According the restriction of graph transformation with multiple inputs and multiple outputs it can be possible to define processes, which requires additional detailed elaboration. This feature is realized only partly with reporting of the restriction places in the output text file.

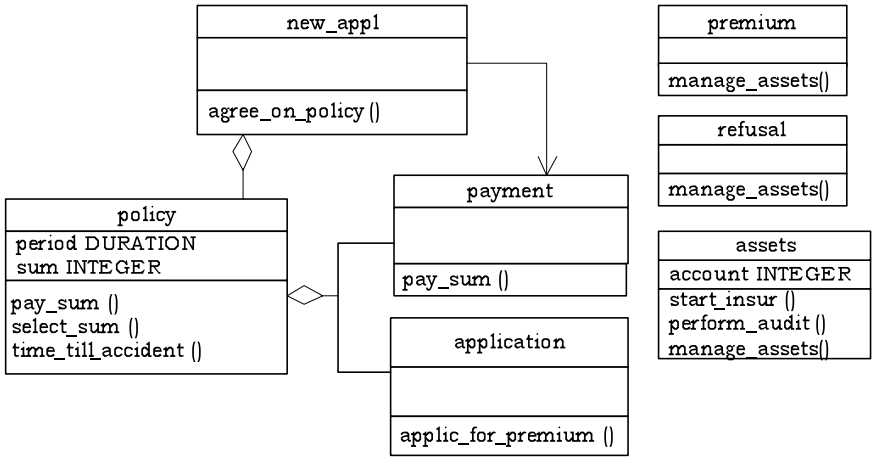


Fig. 7. Class diagram for Insurance business.

The visualization with indicating of a process is not realized because the tool developed for processing of transformations from two-hemisphere model into class diagram does not support yet diagramming abilities, but receiving of XML code gives a possibility to create class diagram with any tool, which support export from XML.

5 Conclusions

The elements of class diagram are received in the formal way during the transformation from the two-hemisphere model into the class diagram. For generation of class diagram elements, elements of the business process and concept models are used. Receiving of elements of the class diagram allows to define a class diagram at the conceptual level. It could serve as a base for further development of the system architecture.

Not all elements of the class diagram are received from the two-hemisphere model on the current stage of research. Definition of such elements as operation arguments, operation return types, stereotypes, constraints and so on is still researched. There exists the probability that for definition of this attributes, an extension of the initial two-hemisphere model will be required.

The proposed transformations are applied for two-hemisphere model of insurance and classes with attributes and different kinds of relationships are identified based on elements of process and concept models. The ability to define all the types of transformations in a formal way gives a possibility to automate the process of class diagram development from correct and precise two-hemisphere model.

The title of the paper is called “Open work of …” it means that the research is under development. Authors try to find the way to receive the rest elements of class diagram and moreover to find the possibility of define system dynamic in a more precise way.

Acknowledgements

The research reflected in the paper is supported by the research grant No. R7389 of Latvian Ministry of Education and Science in cooperation with Riga Technical University “Development of tool prototype for generation of software system class structure based on two-hemisphere model.” and by the European Social Fund within the National Programme "Support for the carrying out doctoral study program's and post-doctoral researches".

References

1. MDA Guide Version 1.0.1., http://www.omg.org/docs/omg/03-05-01.pdf.
2. Kent S.: Model driven engineering. In Proceedings of IFM2002, volume 2335 of LNCS. Springer-Verlag (2002)
3. Kleppe A.: MCC: A model transformation environment. In Proceedings of the ECMDA Springer-Verlag, pp. 173-187 (2006)
4. Kleppe A., Warmer J., Bast W.: MDA Explained: The Model Driven Architecture. Practise and Promise, Addison Wesley, 192 pp (2003)
5. Nikiforova O., Kuzmina M., Pavlova N.: Formal Development of Platform Independent Model in the Framework of MDA: Myth or Reality. In Scientific Proceedings of Riga Technical University, 5th Series, Computer Science, Applied Computer Science Vol.22. – Riga: RTU, pp.42-53 (2005)
6. Pavlova N.: Several Outlines of Graph Theory in Framework of MDA. In Advances in Information Systems Development, New Methods and Practice for the Networked Society, Vol. 2, Edited by Maguar G., Knapp G., Wojtkowski W., Wojtkowski W.G., Zupancic J., Springer Science+Business Media, LLC, pp. 25-36 (2007)
7. Nikiforova O., Kirikova M.: Two-Hemisphere Model Driven Approach: Engineering Based Software Development. In Proceeding of the 16th International Conference Advanced Information Systems Engineering Caise’2004. A. Persson And J. Stirna (Eds.), Lncs 3084, Springer – Verlag Berlin Heidelberg, pp. 219 – 233 (2004)
8. Nikiforova O.: General Framework For Object-Oriented Software Development Process. In Proceedings of Conference of Riga Technical University, Computer Science, Applied Computer Systems, 3rd Thematic Is-sue, Riga, Latvia, pp.132-144 (2002)

9. Chen P.: The entity relationship model – towards a unified view of data. ACM Trans. Database Systems, 1, 9-36 (1976)
10. GRADE Business Modeling, Language Guide. INFOLOGISTIK GmbH.- (1998)
11. Larman C.: Applying UML And Patterns: An Introduction To Object-Oriented Analysis And Design. New Jersey: Prentice Halls (2000)
12. Rumbaugh J., Jacobson, I., and Booch, G.: The unified modeling language reference manual. Reading, MA.: Addison-Wesley, (1999)
13. Grundspenkis J.: Causal Domain Model Driven Knowledge Acquisition for Expert Diagnosis System Development. Kaunas: Kaunas University of Technology Press, (1997)
14. Pavlova N., Nikiforova O.: Formalization of Two-Hemisphere Model Driven Approach in the Framework of MDA. In Proceedings of the 9th Confe-rence on Information Systems Implementation and Modeling, Czech Republic, Prerov, pp. 105-112 (2006)
15. Nikiforova O., Pavlova N.: Foundations of Generation of Relationships between Classes Based on Initial Business Knowledge. In the Proceedings of the 17th International Conference on Information Systems Development (ISD2008) "Towards a Service-Provision Society" (2008) (accepted for publication)
16. Pavlova N.: Approach for Development of Platform Independent Model in the Framework of Model Driven Architecture, Ph.D. thesis, Riga Technical University (2008)
17. GRADE tools, GRADE Development Group, http://www.gradetools.com/ (2006)