COMPUTER SCIENCE
ISSN1407-7493 DATORZINĀTNE 2007-30

TECHNOLOGIES OF COMPUTER CONTROL
DATORVADĪBAS TEHNOLOĢIJAS

DEVELOPMENT OF PLATFORM INDEPENDENT MODEL IN THE FRAMEWORK OF MDA

PLATFORMNEATKARĪGĀ MODEĻA IZSTRĀDE MODEĻVADĀMAS ARHITEKTŪRAS IETVARĀ

Yelena Chukina, *Division of Applied Computer Science, Riga Technical University, Kalku 1, LV1658, Riga, Latvia, B.sc.ing., Jelena.Chukina@gmail.com*

Natalya Pavlova, *Division of Applied Computer Science, Riga Technical University, Kalku 1, LV1658, Riga, Latvia, M.sc.ing., PhD student, Natalya.Pavlova@inbox.lv*

Oksana Nikiforova, *Division of Applied Computer Science, Riga Technical University, Kalku 1, LV1658, Riga, Latvia, Dr.sc.ing., asoc. prof., oksana.nikiforova@cs.rtu.lv*

MDA, PIM, methodologies, analysis models

1. Introduction

The Model Driven Architecture (MDA), being built under supervision of the Object Modelling Group (OMG), separates the system business aspects from the system implementation aspects on a specific technology platform [1]. MDA defines the approach and tool requirements for specifying systems independently of platforms, specifying platforms, choosing particular platform for the system and transforming business domain specification into one for a chosen platform. The MDA separates certain key models of systems and brings a consistent structure to these models. Models of different systems are structured explicitly into Computation Independent Models (CIM), Platform Independent Model (PIM) and Platform Specific Models (PSM) [2].

The PIM provides formal specification of the structure and functionality of the system that abstracts away technical details. There are tools to generate a code from PSM, but are not tools to generate a PSM from PIM. It is a serious problem, and this problem decision could be in precise PIM construction to transform it to PSM automatically. Also there has to be rules for PIM checking if it defines all problem domain concepts in the correct way.

The section 2 describes MDA and transformations during modeling process. The section 3 describes the stages of PIM development, and presents it in the graphics. Section 4 summarizes all information in one table. In this table is shown level of formalization of

development stage for every discussed methodology. The conclusions of this paper are depicted in section 5.

2. Model Driven Architecture

MDA and Model Driven Engineering (MDE) [3] propose a software development process in which the key notions are models and model transformation. Software is build by constructing one or more models, and transforming these into other models during this process. The common idea of this process is platform independent model as input and platform specific as output, such as the platform specific models can be easily transformed into a executable format. In other words, the model driven process is commonly viewed as a code generation process. [4]

2.1. Computation Independent Model

Computation Independent Model: is the model which defines how a business works without reference to software systems.

The purpose of CIM is to represent real world system and to define requirements. Programming concepts are not considered at this abstraction level. Sometimes CIM is called also for a problem domain model or business model.

2.2. Platform Independent Model

Platform Independent Model: is the model which resolves business requirements through purely problem-space terms and it does not include platform specific concepts.

The PIM provides formal specification of the structure and functionality of the system that abstracts away technical details. There has to be rules for PIM checking if it defines all problem domain concepts in the correct way.[1]

PIM is describing that part of information system specification, which is close to code, but is independent of platform specific features. PIM is representing information system in that way that will remain unchanged on any programming platform. Nevertheless PIM usually is accommodated to specific architecture style.[1]

2.3. Platform Specific Model

Platform Specific Model: is a solution model which resolves both functional and non-functional requirements through the use of platform specific concepts.

The platform definition can include wide range of conceptions in the context of MDA. It can be operation system, programming language, any technological platform, such as CORBA, Java 2 Enterprise Edition, also any specific vendor platform (for example, Microsoft .NET) [1]. Platform can imply any of engineering and technological characteristics, which are not important for program unit fundamental business functionality.[1]

3. PIM model development

The goal of PIM model is to show the system from a business point of view. PIM describes functional and nonfunctional requirements received from user interview, a system's environment and the way this environment interacts with a user. In the PIM isn't included any platform specific details for the functional requirements. PIM contains business environment information, necessary algorithms and calculations, and information structure – everything what is necessary for a software development. [1]. A platform is an engineering and technological characteristic, which are not under the fundamental functioning law. A platform in the MDA architecture is a wide concept – it can be an operating system or a programming language or a technological platform (for example: CORBA, Java2 enterprise edition), it can be a particular development platform also (Microsoft .NET).

Information about a system, about business environment in the PIM model is developed step by step. The four main steps of PIM model development are defined during the [5] and [6] works.

If the PIM model is transformed back to the PIM model, although with more detailed information, so called model refining is used [1]. In this case the model is transformed from a higher abstraction level to a lower, all the time PIM model is not dependent on other platforms however.

In the PIM model development the following stages are selected::

- context PIM;
- requirements PIM;
- analysis PIM;
- components design PIM.

3.1. Context PIM

In the first step of PIM context defining, system's software goals and system's clear definition must be chosen. PIM context defines system's business principles, system's boundaries and system's clear definition. PIM context's problem is to make system's description so clear, that a human with no experience could understand it. First action of this step is to finding out system's goals and business principles, description system's external actors, which interact with the system, identifying business services and business objects [5]. Being based on [5] and [6] context PIM definition can be divided in following steps:

- System's context definition. In this step a system's context is created. First, the system itself is defined, its services, business principles and standards, system's goals and relations between the system and actors involved. The system is defined and described using Class diagram.
- System's boundaries definition. In this step a system's boundaries, system's relations with other systems are defined. This step is represented using Use Case diagram.
  - Actors definition. Each actor is related to a system through request service according his interests. Actors are human and machines, which interact with the system.
  - Business object definition. Business object helps to find out who does use the system and what do actors give to a system.
  - Data flows between system and actors definition.
- System's functionality definition. Each system's use case describes functionality. System's behavior is represented with a Sequence Diagram.

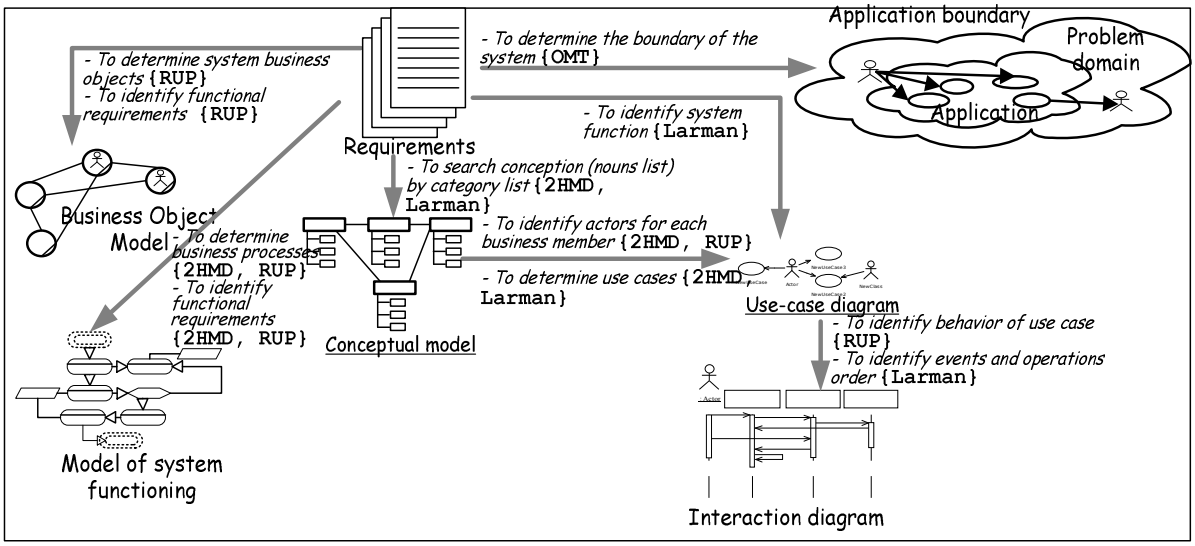

Figure 1. Information flow in Context PIM

Step of PIM context definition shows system's behavior with external objects, where internal systems behavior is not considered. System's collaboration with external actors shows system's behavior against functional areas.

Figure 1 shows the information flows and transformations required for first PIM part [7,8].

3.2. Requirements PIM

Once a PIM context is defined, requirements are specified. System is described from an external point of view in this model. All client's functional and non-functional requirements are specified. Main tasks of PIM requirements definition step's are context PIM refinement, services identification, business use cases and business objects, which are created and used with the system, actors, which interact with the system, to specify capabilities (functional use cases), to identify and to model relations between functional and non-functional requirements [5, 6].

- Functional requirements specification.
  - Refinement of system's context. System, actors, usage cases and business object are refined. Overall services and actors are identified. Relations between business objects and system are modeled in the Class Diagram.
  - Organization and identification of system's capabilities. Capabilities are functional use cases. Those are organized accordingly identified functional areas in context PIM. Capabilities help to find out and to check out model's elements, which represent system's functional requirements (business objects, business events, services). All the elements are described with a behavior, which is modeled with an activity diagram, scenarios, which are modeled with a Sequence Diagram.
- Nonfunctional requirements specification. Nonfunctional requirements are described and organized in higher level. Those are described by QoS (quality of service) or by a platform constraint (for example, operating system, memory size). QoS are build of QoS characteristics, using UML profile standard.

- Functional and non-functional requirements relation. In the beginning, requirement's relationships are shown with capabilities and non-functional requirements. After that, the QoS values are expressed – these are the QoS characteristics which were given to the context as values and applied to the functional elements of the model, where it is asked to.

The basic requirements model is created during process of refining of context PIM model and following refining rules between different abstraction levels. This step could be done partly or fully automatically with existing modeling tools.

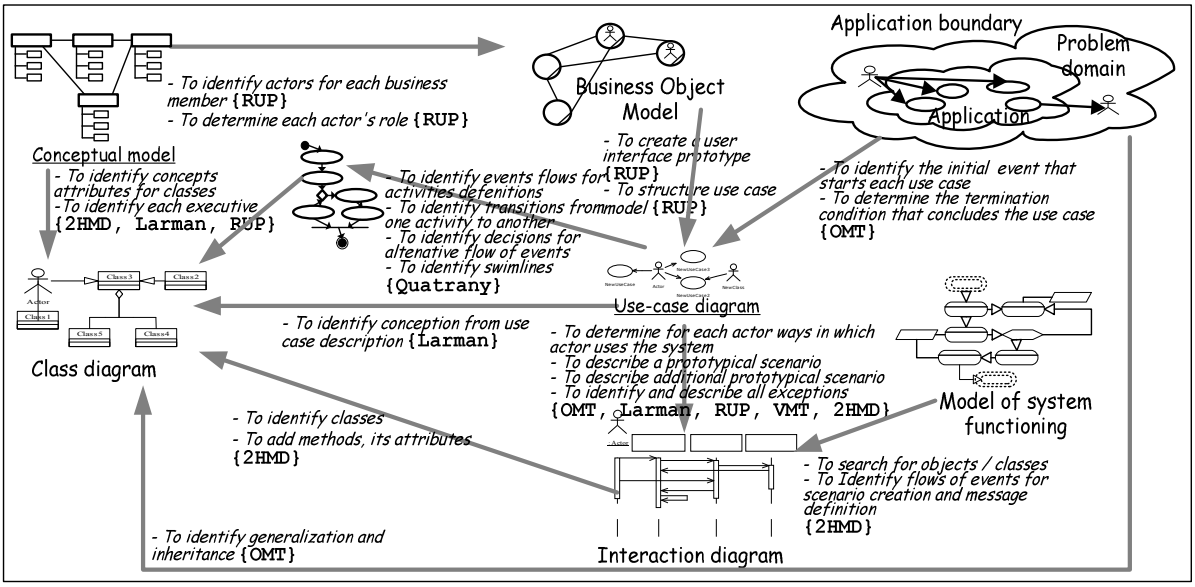

Figure 2. Information flow in requirements PIM

On Figure 2 is shown information, described in this part [7, 8].

3.3. Analysis PIM

Once requirements PIM identification and modeling is ready, PIM is being analyzed. Internal system's structure is created not considering technology or software. It shows system's functional specifications against environment and usage areas. Moreover, functional and non-functional aspects are divided into separate groups. Functional describes system's objects (for example, with classes, attributes, packages), system's functions (with operations) and its boundaries (with external interfaces) also. However non-functional aspect is developed according QoS notion [5, 6]. Analysis PIM steps are as follows:

- System's external interface maintaining. The external interfaces between system and actors are analyzed and maintained accordingly with requirements PIM specification. Class and Component diagrams are used.

- Domain analysis. Functional structure of the system is created. Using requirements PIM specification, functional capabilities are fulfilled and checked out accordingly to each scenario. All this is represented in Sequence Diagram.
- QoS analysis. Non-functional aspects in requirements PIM specification are defined. Forces and QoS are analyzed and refined.

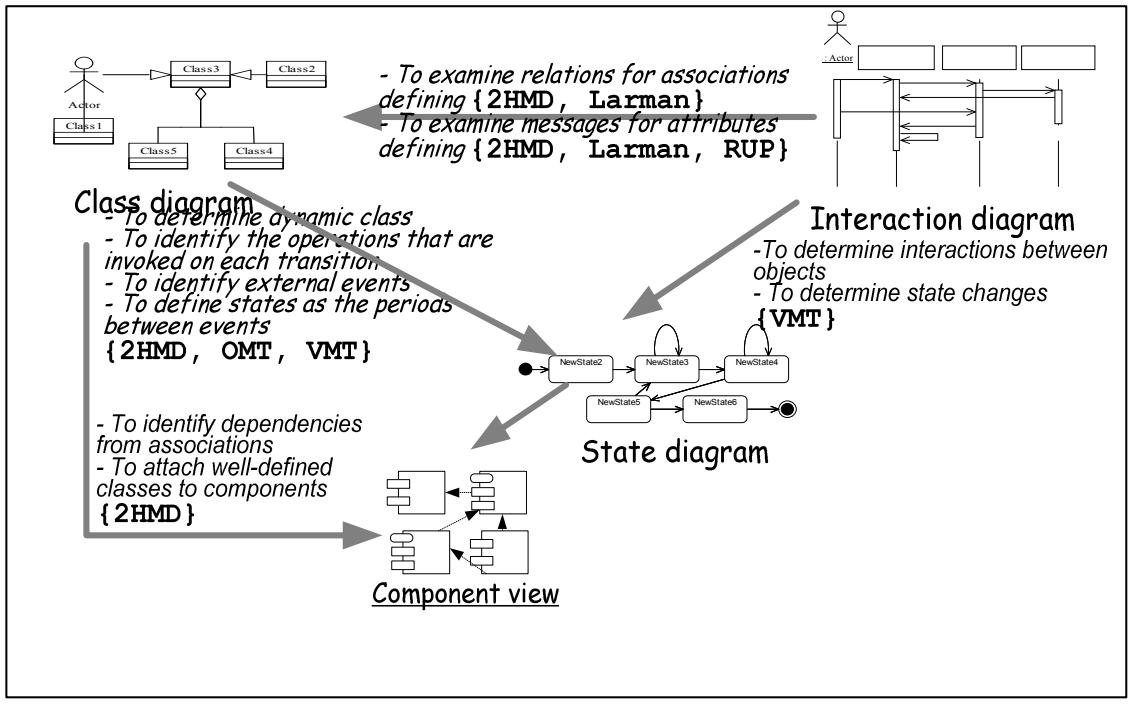

Figure 3. Information flows in analysis PIM

Figure 3 shows described in this part information flows[7,8].

4. Comparison and discussion of PIM development methodologies

Methodologies, which were used to show information flows in PIM parts is joined in Table 1. On the Figures 1, 2, 3 near to every information transition are enumerated methodologies, which propose **what** should be received from one modeling stage to another, but **how** it could be received methodologies describes in different level of formalization. To illustrate this formalization level is shown Table 1 where formalization level of every element receiving is shown.

In the Table 1 [9] are presented methodologies, based on which PIM was discussed in this paper. Solutions in Table 1 are analyzed according to the ability of PIM elements derivation from CIM and PIM transformation to PSM, which can be arranged according to the level of automation.

The scale is the following:

Manual transformations (1) – only human, an expert, may take such a decision. In this case all the decisions are made using an experience, subjective opinion. It is not possible to check the results or support such transformations with tools.

Transformations based on hints (2) – there are not any algorithms, how to perform transformation, but there are some hints or guidelines, how to make a decisions. Nevertheless, these transformations can not be supported by tools, because guidelines are not formalized.

Partly formal transformations (3) performed using special algorithms – in this case there are some formal algorithms, which define, how to represent particular element of one model into another model, but these algorithms don't cover all the elements have to be transformed. Thus, only part of the model may be transformed automatically.

Completely formal transformations (4) – all the process of transformation is defined in a formal way, so complete transformation may be performed automatically [9].

MDA proposes that all model transformations (CIM -> PIM -> PSM - > code) must be performed according to the fourth transformation type. Nevertheless, not any of the reviewed solutions supports full circuit of MDA realization.

Table 1. Analysis of transformation abilities and their levels of automation in MDA realization solutions

Methodology / System analysis phases	OMT	VMT	Larman	Quatrany	RUP	2HMD	2HMD'
Functioning definition	1	1	1	1	1	1	1
System boundary definition	1	1	1	1	1	1	1
System concept definition	2	2	2	2	2	2	2
Use case definition	1	1	1	1	1	1	1
Actors definition	1	2	3	2	3	3	3
System structure definition	3	2	2	1	2	3	3
System behavior definition	3	2	2	2	2	3	4
Relations definition	3	3	3	2	3	3	4
Activities definition	2	2	2	2	3	3	4
States of system artifacts definition	2	2	2	2	2	3	3
State changes definition	2	2	2	2	2	3	3
System components definition	3	3	3	3	3	3	4

Based on Table 1 we could conclude that for now more formal methodologies is 2HMD, which not only direct „what" should be done, but define „how" .

The conclusion we can use in our further researches, to modify 2HMD methodology for maximal formal and automatic approach to system analysis in the framework of MDA.

5. Conclusion

In this paper was described PIM development process and discussed PIM construction approaches. In the PIM construction three main stages are selected:

- PIM context definition,
- PIM requirements specification

- PIM analysis.

PIM context definition presents a system view. As well there system principles and goals are defined. Interaction of external actors and business objects defines boundary of the system. Business objects are needed for definition of system usage. The system functioning is described with use cases. Each use case describe one part of system functioning. In further modeling system context should be refined with behavior of system objects, and user requirements. In this way is defined system internal behavior and PIM level of abstraction is changed to more details.

The system requirements should be described during PIM requirements specification. System behavior and requirements are described without any platform specific details, using only functional specification. In this way PIM analysis stages are selected from nonfunctional aspects.

In this paper different PIM construction methodologies are discussed. The main problem is information transformations in PIM construction. In the paper the following methodologies were discussed:

- OMT (Object Modelling Technique by J. Rumbaugh) [10],
- VMT (Visual Modelling Technique by D.Tkach) [11],
- Larman (strategy, proposed by C. Larman) [12],
- Quatrany (strategy, proposed by T. Quatrany) [13],
- RUP (Rational Unified Process by J.Rumbaugh, I. Jackobson, G.Booch) [14],
- 2HMD (Two Hemisphere model driven approach, proposed by O. Nikiforova and M. Kirikova) [15]
- 2HMD' approach [8] (refined Two Hemisphere model driven approach, proposed by N.Pavlova and O. Nikiforova).

Every methodology proposes PIM construction steps. These steps were combined in figures 1,2,3. Figures present every stage of PIM constructions.

During PIM construction a lot of information was transformed. For every phase in every metodology of PIM construction there is defined formalization level. The Table 1 presents grades of formalization for every selected stage. Grade of formalization is in interval from 1 to 4, where 1 is undefined transformation, and 4 – fully formal transformation.

Analyzing Table 1 we can conclude, that more formal is refined 2HMD approach. Received here conclusion could be used in further researches – for defining of every methodology detail activities. And based on it will be possible to create common two hemisphere model for full PIM construction process using activities from different methodologies.

This work has been partly supported by the European Social Fund within the National Program "Support for the carrying out doctoral study program's and post-doctoral researches" and by a grant No. ZP/2005-02 of Riga Technical University within the project "Application of Two-Hemisphere Approach for Development of Flexible Architecture for Software Engineering Body of Knowledge

References

1. MDA Guide Version 1.0.1/ Internets.- http://www.omg.org/docs/omg/03-05-01.pdf
2. Anneke Kleppe, Jos Warmer, Wim Bast, MDA Explained : The Model Driven Architecture – Practise and Promise, Addison Wesley, 2003., 192.lpp

3. Stuart Kent. Model driven engineering. In Proceedings of IFM2002, volume 2335 of LNCS. Springer-Verlag, 2002
4. Anneke Kleppe. “MCC: A Model Transformation Environment”, A. Rensink and J.Warmer (Eds.): ECMDA-FA 2006, LNCS 4066, pp. 173-187, 2006. Springer-Verlag Berlin Heidelberg 2006
5. Process Model to Engineer and Manage the MDA approach, 2003, / Internets:.- http://modeldrivenarchitecture.esi.es/mda_publicDocuments.html
6. Daniel Exertier, Benoit Langlois, Veronique Normand, UML Specialisation Approach, MASTER: WP2 MDE Foundation, 2003., / Internets.- http://modeldrivenarchitecture.esi.es/mda_publicDocuments.html
7. Pavlova N., Ņikiforova O. An overview of advanced approaches for construction of platform-independent system model, Scientific Proceedings of Riga Technical University, The 5th Series – Computer Science. Applied Computer Systems, 2005
8. Pavlova N., Nikiforova O. Formalization of “Two-Hemisphere Model Driven Approach in the Framework of MDA” Proceedings of the 9th International Conference “Information System Implementation and Modelling” (ISIM’06), April 25-26, 2006, Přerov, Czech Republic. - Ostrava: Jan Štefan MARQ., 2006. – 105-112 pp
9. O. Nikiforova, M. Kuzmina, N. Pavlova, Formal Development of PIM in the Framework of MDA: Myth or Reality, Scientific proceedings of Riga Technical University, series – Computer Science, applied Computer Systems, 7th Thematic Issue, 2006 – in press
10. Rumbaugh J. “OMT: The Developing Process”, Journal of Object Oriented Programming, No.8, pp 14-18, 1995
11. Tkach D., Fang W., So A. ”Visual modelling technique: object technology using visual programming” – Addison Wesley, 1996
12. Larman C. ”Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design”, Prentice Hall, New Jersey, 2000
13. Quatrany T. "Visual Modeling With Rational Rose 2000 And UML". Second Edition, Addison-Wesley, 2000
14. Jacobson I., Booch G., Rumbaugh J., ”The Unified Software Development Process”, Addison Wesley, 2002
15. Nikiforova O., Kirikova M., “Two-Hemisphere Model Driven Approach: Engineering Based Software Development”, Proceeding Of The 16th International Conference Advanced Information Systems Engineering Caise’2004, A. Persson And J. Stirna (Eds.), Lncs 3084, Springer – Verlag Berlin Heidelberg, 2004., Lpp. 219 – 233

Čukina J., Pavlova N., Ņikiforova O. PIM izstrāde MDA ietvarā

Modeļu vadāma arhitektūra (MDA) ir jauna pieeja programmatūras izstrādē, kura paredz sistēmas funkcionālitātes atdalīšanu no platformas specifikācijas, uz kuras sistēma tiek veidota. Galvena uzmanība MDA ietvarā ir veltīta sistēmas no platformas neatkarīgajam modelim, no kā ir atkarīga no platformas atkarīgā modeļa un koda iegūšana. Lai no platformas atkarīgā modeļā būtu attēlota pilnīgi sistēma no PIM modeļa, PIM modelim jābūt formālam, tas ir, PIM konstruēšanas gaitā informācijas trasnformācijām jābūt automātiskām. Šis pētījums apraksta PIM modeļa konstruēšanas etapus un analizē dažādu metodoloģiju piedāvātas pieejas PIM modeļa formalizācijas iegūšanai. Dotajā raksā autores apskata populārākos sistēmas modelēšanas metodoloģijas, lai uzzinātu, nevis kas tiek transformēts PIM konstruēšanas gaitā, bet kā tas ir transformēts. Izpētot kā tiek transformēta informācija PIM modelī, var noteikt katras metodoloģijas piedāvātu formalizācijas pakāpi. Katrā metodoloģijā tiek izdalītas fāzes PIM konstruēšanā, un apkopojot to sastādīta tabula ar katras konstruēšanas fāzes formalitātes novērtējumu. Rakstā beigā tabula parāda katras metodoloģijas formalizācijas līmenī.

Chukina Y., Pavlova N., Nikiforova O. Development of Platform Independent Model in the Framework of MDA

Model Driven Architecture (MDA) as an OMG standard is one of the most effective amongst them. MDA is based on models which distinguish between a system functionality specification and this specification realization

on a given technological platform. A model helps to ensure it. 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. Each of the models differs by its level of abstraction which diminishes closing to a code. CIM model is called a business model, as it describes a system requests and an environment which this system will be used in. After a CIM model creation a PIM model is created, which describes CIM model business process' platform dependent software system. Using transformation tools or manually a PIM model is transformed into a PSM model, a platform dependent model (.NET, CORBA, Web Services, XMI/XML, Java). A code will be generated through a PSM model transformation. The most vital stage in a transformation line is PIM->PSM. To provide automatic transformation, a PIM model has to be created using a language which is able describe a system from various points of view, system behavior, system's business objects, system actors, system use cases and so on. MDA supposes different realization solutions. Each of the solutions provides its own approach to the full transformation line. A solution's base is modeling techniques. In the paper several strategies proposed by different authors are discussed. None of them realizes a full MDA automatic transformation line. However, each approach's ideas provide a good basis for new creations within automatic transformation line realization's future.

Чукина Е., Павлова Н., Никифорова О. Разработка платформо независимой модели в рамках МДА
Архитектура Управляемая Моделью (MDA – Model Driven Architecture) является наиболее еффективной среди них. МДА основана на моделях, делящихся на специфичные для функционирования системы и реализации этого функционирования для данной технической платформы. МДА состоит их четырех моделей –Модель Независимая от Вычислений (CIM – Computation Independent Model), Платформо Независимая Модель (PIM – Platform Independent Model), Специфичная для Платформы Модель (PSM – Platform Specific Model) и модель программного кода. Все эти модели входят в линию трансформации CIM→PIM→PSM→code. Каждая из этих моделей отличается уровнем абстракции, который уменьшается приближаясь к модели кода. Независимую от Вычислений Модель называют моделью бизнеса, так как она описывает системные требования, и среду в которой система будет использоваться. После создания Независимой от Вычислений Модели обычно конструируют Платформо Независимую Модель, которая описывает автоматизируемые бизнес процессы. Используя трансформационные средства, или вручную Платформо Независимую Модель преобразуют в Специфичную для Платформы Модель. Код должен быть сгенерирован используя Специфичную для Платформы Модель. В череде трансформаций наиболее важной является трансформация PIM->PSM. Для обеспечения автоматической трансформации Платформо Независимая Модель должна быть сделана используя язык моделирования, способный отобразить систему с разных точек зрения, поведение системы, бизнес объекты системы, актёров и варианты использования. MDA предлагает разные решения реализации. Каждое решение обеспечивает собственный подход к полной череде трансформаций. В статье рассмотрены метожики, предложенные различными авторами. К сожалению ни одна из перечисленных выше методик не обеспечивает полный проход трансформаций. В то же время идея каждой методики даёт хорошую основу для реализации автоматических трансформаций.