Business Knowledge Modelling Using the BORM Method

Martin Molhanec[1], Vojtěch Merunka[2], Iveta Merunková[3]

[1]Department of e-Technology, Czech Technical University in Prague, Czech Republic, e-mail: molhanec@fel.cvut.cz
[2] Department of Information Engineering, Czech University of Life Sciences Prague, Czech Republic, e-mail: merunka@pef.czu.cz
[3] Department of Gardening, Czech University of Life Sciences Prague, Czech Republic, e-mail: merunkova@af.czu.cz

Abstract. BORM (Business Object Relationship Modeling) is a development methodology used to store knowledge of process-based business systems. BORM is based on the combination of object-oriented approach and process-based modelling. Especially, in the case of agricultural, food and environmental enterprise informational systems we need to work with new flexible modelling tools, because processes and data are instantly changing and modifying through the whole life cycle of such systems. In this paper, we present BORM use as a tool for capturing process information required in the initial phases of information system development.

Keywords: BORM, business modelling, agriculture modelling, environment modelling, business process management, process simulation, Craft.CASE.

1. Introduction

The attitude of business towards Information Technology (IT) is constantly changing, and increasingly sophisticated. New systems and tools are becoming available. Additionally, there is a constant exchange of ideas between the IT and the business communities, arising out of the development of knowledge-based systems. Today, when modern visual programming tools, combined with the support of rapid web-based application development environments and sophisticated end-user hardware technologies, are available, it would appear that the whole software development process is becoming easier. However, this statement can apply only in those cases where the system complexity of the solution and of the users' requirements is relatively small.

The aim of our projects was to analyse and suggest improvements to business processes, company structure, data flows, organizational structure; as well as providing IT support for them. Especially, in the case of agricultural, food and environmental enterprise informational systems we need to work with new flexible modelling tools, because processes and data are instantly changing and modifying

In: M. Salampasis, A. Matopoulos (eds.): Proceedings of the International Conference on Information and Communication Technologies
for Sustainable Agri-production and Environment (HAICTA 2011), Skiathos, 8-11 September, 2011.

through the whole life cycle of such systems. We soon realized that we needed to carry out analyses of the problems that were supposed to be solved in order to be able to design the system and properly test their solution.

First, it is important to identify, document, and test a system in order to be able to analyse and design a more elaborated system. There are different methods and tools. On one hand, there are methods and tools oriented towards business modelling such as EPC (EPC, 2008) or varieties of Flow Charts. However, these tend to have paradigmatic gap in weak relationships with the software engineering. On the other hand, there are methods and tools of software engineering based on UML, which assume that the business requirements have already been correctly formulated and verified (Eriksson and Penker, 2000). We used several methods for our projects, but none of them were able to smoothly combine these two worlds of modelling. This is why we began our own research.

The second is the construction of an initial object model, often called an essential object or conceptual model, built from of a set of domain specific objects known as essential objects. Both these tasks should be carried out with the active participation of the stakeholders, in order to ensure that the correct system is being developed. Consequently, any tools or diagrams used at these early stages should be meaningful to the stakeholders; many of them are not 'computer system literate'.

The most common technique for specification of requirements in current object-oriented methodologies is Use Case modelling, and subsequent use of Sequence, Collaboration and State-Chart Diagrams. This is the foundation of most Object-Oriented development methods. However, this approach is often insufficient by itself to fully support the depths required for initial system specification. Business analyst highlights some deficiencies in this approach. There are many views on the effectiveness of Use Cases and related tools as a first stage in System Design. (Simons and Graham, 1999) for example describe a situation where Use Case Modeling obscures the true business logic of a system.

Finally, we conclude this discussion by a brief comparison of the BORM method with selected well-known other business (or software engineering) modelling methods presented in Table 1. It is clear that all compared methods have some deficiencies. More precisely, they do not cover all required features. Naturally, the primary objective of the developing of the BORM method is covering as many required features as possible.

Table 1. Comparison of some business (software engineering) modelling methods.

	ARIS	**BPMN**	**RUP**	**BORM**
theory behind	Petri nets	flow chart	object-oriented (UML)	combination of FSM and object-oriented
business modelling support	yes	yes	poor, use-case modelling only	yes
requirement support	only indirectly	only indirectly	yes	yes

software engineering support	no	no	yes	yes
verification and validation	process simulations	process simulations	using prototyping	using process simulations, instance-level modelling
has textual phase before diagraming	no	no	no	there is a use of process scenarios
consistency of concepts during modelling	no	no	using the MDA	constrains rules in the BORM meta-model
modelling phases	there are no phases	there are no phases	7, internally detailed	5, internally detailed

2. Method Background

The BORM (Business Object Relation Modelling) method is in continuous development since 1993 when it originally was intended as an instrument to provide support for building object-oriented software systems based on pure object-oriented languages such as Smalltalk and object-oriented databases. It has now evolved into a system development methodology that has been used successfully in about 30 projects. These systems range through all sizes of software development.

The most common technique for requirements specification in current software development methodologies is Use Case modelling as the start of UML documentation process. Indeed Use Cases are often the foundation of most object-oriented development methods (Jacobson, 1992). It is concerned with the identification of external actors, which interact with the software part of the system. This means that in order to employ Use Case modelling, it is necessary to already know the system boundary and distinguish between entities, which are internal and external to that boundary. It is our experience that the correct identification of the system boundary is a 'non-trivial' task, which often requires significant understanding of the proposed system and consequently can only successfully take place at the end of the requirements specification stage.

Our experience in system modelling suggests that UML is not suitable for the first stages of analysis, where business processes need to be recognized. UML diagrams are too complex for the business community as they often contain too much detail concerning potential software implementations. This means classes, inheritance, public/private methods, attributes, link classes, etc. Here we got almost the same experience as documented in (Simone and Graham, 1999).

We believe that the business community needs a simple yet expressive tool for modelling; able to play an equivalent role to that played by the Entity-Relation Diagrams, Data-Flows Diagrams or Flow-Charts over the past decades. One of the strengths of all these approaches was that they contained only a limited set of concepts (about five) and were comprehensible by problem domain experts after a few minutes of study. Unfortunately the UML/BPML approach lost this advantage of simplicity.

3. BORM Method

The BORM method has been developed on academic grounds since the 1990s. It unifies the MDA principle, using an object-oriented paradigm and a unified approach to business and IT system modelling. For more on the BORM method, see (Knott et al., 2000; Liu and Roussev, 2006).

3.1. BORM Fundaments

The three primary theoretical BORMs fundamentals are as follows.

- MDA approach that separates a specification of business system description (CIM – Computation Independent Model) from its computer implementation specification (PIM – Platform Independent Model); and this computer specification from the final solution on a concrete technological platform (PSM – Platform Specific Model). MDA is created and maintained by the Object Management Consortium (MDA, 2009).

- Object-oriented programming (OOP) approach has its origins in the researching of GUI and programming languages that took place in the 1970s. It differs from other software engineering approaches by incorporating non-traditional ways of thinking into the field of informatics. The basic element is an object that describes data structures and their behaviour. OOP has been explained in many books, but we think that this one (Goldberg and Rubin, 1995) written by OOP pioneers belongs to the best.

- Automata theory is a study of abstract finite-state automata and the problems they can solve. An automaton is a mathematical model for a device that reacts to its surroundings, gets an input, and provides an output. An automaton's behaviour is defined by a combination of its internal state and its accepted input. The automata theory is a basis for system behaviour descriptions.

3.2. Business Engineering – Business Models

The first part of the method covers the CIM field, i.e. business engineering. It transforms a project assignment into a model described by data hierarchies, process participants, process scenarios, various diagrams and generated reports. The main instrument of verification and validation is the process simulator.

For the following purposes, it is possible to use this part of BORM without any relation to a software engineering phase:

- Organizational consultancy projects. These are process analysis, organizational structure analysis, and drafts for processes or organizational structure improvement.
- Projects documenting processes and organizational structure. These are, for instance, projects whose aim is knowledge management, creating training materials, knowledge visualization, etc.
- Projects for preparing the groundwork for selection procedures for organizational consultancy, or other consultancy services.
- Projects for preparing the groundwork for selection procedures for the delivery of information systems, or other software engineering projects

3.3. BORM Interpretation of MDA

In BORM each concept has some of the following:

1. A Set of predecessor concepts from which it could be derived by an appropriate technique and a Set of successor concepts, which could be derived from it by an appropriate technique.
2. A validity Range - The phases where it is appropriate. In each phase of BORM modelling, only limited set of concepts is recommended.
3. A Set of techniques and rules, which guide the step-by-step transformation and the concept revisions between the system development phases. There are refactoring techniques; data normalization, design patterns and other programming-related techniques (Ambler, 1997) adopted for BORM concept transformations.

3.4. BORM Business Diagram

BORM respects UML and MDA, but uses an original diagram for business process modelling. It conveys together information from three separate UML diagrams: state, communication and sequence. The BORM group has found that it is clearly understood by business stakeholders.

- Each subject participating in a process is displayed in its states and transitions.
- This diagram expresses all the possible process interactions between process participants. The business process itself consists of a sequence of particular communications and data flows among participating subjects.

More formally, BORM process diagrams are graphical representations of interconnected Mealy-type finite state machines of particular subjects. Visual simulation of a business process is based on market-graph Petri net. Almost the same approach is described in detail by (Barjis and Reichgett, 2006). Therefore we can show states, transitions and operations for all subjects playing a role in a business process. This is a very powerful, but a simple diagram, see Fig. 5, onward.

3.5. Craft.CASE Modelling Tool

The Craft.CASE tool has been designed according to state-of-the-art principles of object-oriented software development, and uses a special Smalltalk programming language technology that has been evolving since the 1970's. The program consists of many user functions, has its own internal object-oriented database, several graphic editors and a programmable interface. The Craft.CASE tool provides all instruments for CIM (as business models) and PIM (as conceptual models), including their mutual interconnection and the possibility to undertake thorough testing (Craft.CASE, 2009).

The Craft.CASE can be used in process and organizational consultancy and in analytical projects and information system drafts while identifying requests on newly designed systems; also in component modelling and service oriented architecture. It works with a system model and its processes in an original way that is based on:

1. An ability not only to visualize processes and systems in the form of diagrams, but also to test them by means of cross-references and graphic simulations.
2. An ability to model not only symbolic terms (for example, a customer, order, payment, etc.) as drawn symbols in a diagram, but also the possibility of working with real objects (several concrete customers, orders, payments, etc. that differ in their real values such as a date, name, price, etc.). Craft.CASE supports both class-level and instance-level of modelling.
3. A method having diagrams as the result of a gradual deriving and checking.

More information about Craft.CASE such as its programming facilities, metamodel etc. is described in (Merunka et al., 2008).

4. Case Study

Our example of modelling using Craft.CASE is to model an information system for a company called FD (Food Delivery); dealing with organic food products delivered from suppliers to local customers. A typical example of such activity is shopping and delivering the goods to a home, or home delivery of pastries and milk and similar products. The information system is primary designed for communication between company and its delivery employees. It also provides an information portal for customers to whom the products are to be delivered.

The case study presented subsequently is structured according to three main phases of the BORM method development process, as briefly depicted in Fig. 1.

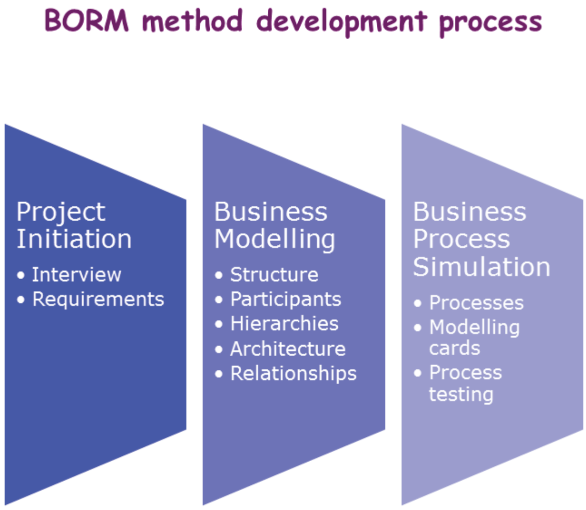

Fig. 1. Three main phases of the BORM method development process.

4.1. Project Initiation

The first stage of business modelling is the initiation of work on a project. At the beginning of this phase there are interviews, and then an initial structure and relations are modelled. The testing of generated reports is executed as the final part of this stage. This task is supported by free model drawing in Craft.CASE, see Fig. 2.

Interview & Requirements. Since kick-off meeting with the client, we have negotiated that the project should focus on ordering a meal through a customer process. The client expressed a preference for communication with customers via web pages. The client also wants work activity descriptions for the logistics manager and van drivers.

4.2. Business Modelling (Computer-Independent Model)

Structure. Structure definition is the first formal step of the method. An analyst should be capable of finding key subjects and process fields on the basis of performed interviews.

Participants. Based on the scope of the project and the allowed project time, eight participants were found (See Table 2). These are the job positions of the logistics manager and the van driver, as well as future software components of the web interface and database system; and, of course, a customer and some suppliers, since they take part in the processes.

Hierarchies. For the detailed mapping of problems to be solved and future use of information system creating, two hierarchies of concrete testing data were recognized and modelled. They are food delivery product catalogue and a list of suppliers for these products, see Fig 3.

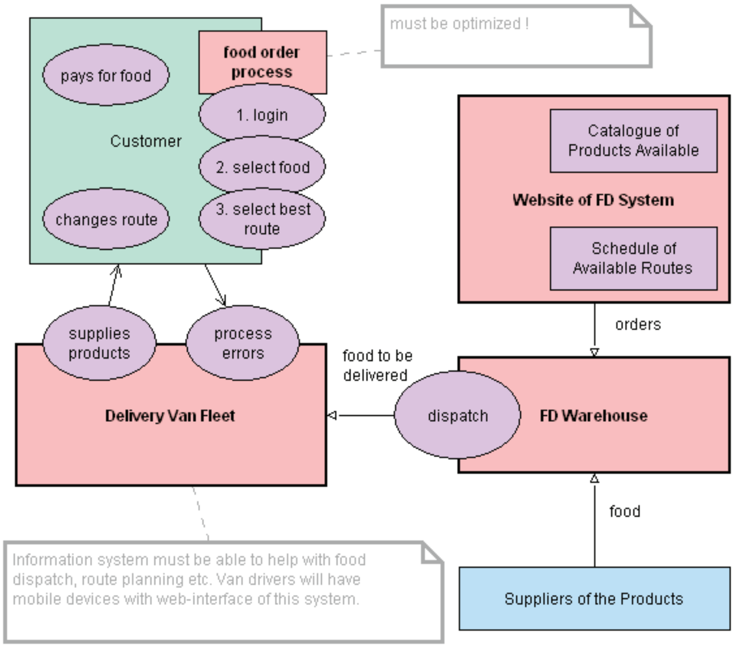

Fig. 2. Record of kick-off meeting interview – free drawing in Craft.CASE

Architecture. Five functional areas were developed, see Fig. 4. It was then decided that three functions (marked as external functions) related to care, and that virtual food products warehouse, advertising activity and web page maintenance will not be included in the concrete scope of our project. The project was limited to activities related to ordering and delivering (marked as internal functions).

Relationships. Several bindings were modelled between elements of the product hierarchy and the product supplier hierarchy to finalize system requirements. These data relationships (elements of the Cartesian product, more formally) show a concrete supplier of concrete products that can be used later for instance-level testing.

Table 2. Participants taking part in the modelled process

Customer	Person who orders food or a person who wants to be a customer.
FD Database	Database of products, orders, routes, customers and suppliers.
Logistics manager	Person responsible for the distribution of orders. This person uses information and scheduling system.
Supermarket Supplier	Department of Supermarket responsible for food delivery.
Supplier	Small company or individual farmer.
Supplier Ordering System	Automatic system for requesting suppliers for food.
Van Driver	Person who delivers food.
Website of FD	Communication software enabling orders.

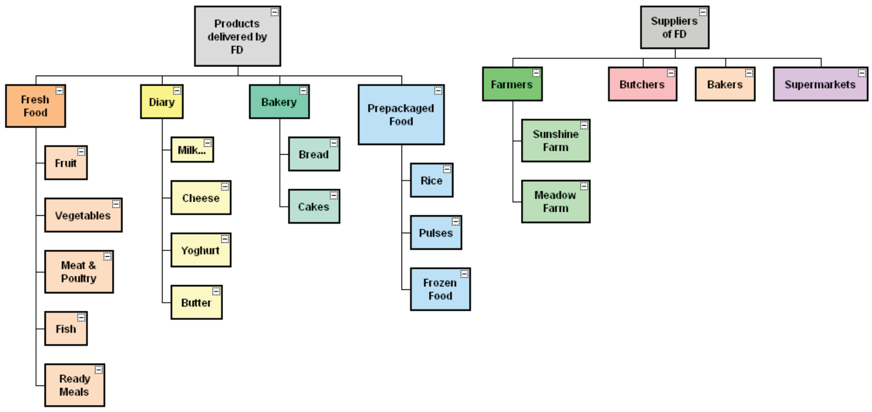

Fig. 3. Products and supplier hierarchies

4.3. Business Process Simulation

Processes. First, the ordering process was modelled, see Fig. 5. Based on the project focus, the food delivery company, a new property for communications was created, with „manual" or „automatic" values, with a relation to the graphic interface where, for automatic communication -- expecting a software realization -- a thick arrowed line between oval activities is drawn. Manual communications are visualized in the standard way of thin arrowed line between oval activities.

Modelling cards. The modelling cards of selected objects were presented during a meeting with the client. Based on these cards (generated as a result of the simulation) a process model was finally confirmed by clients/stakeholders for subsequent software implementation.

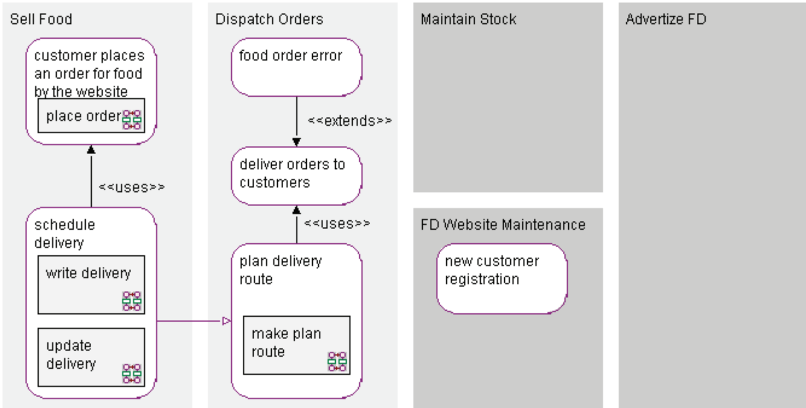

Fig. 4. Business architecture diagram (Use-Case clone) of the whole system requirements

Process testing. Detailed modelling cards and a simulator were used for testing in this step. Craft.CASE simulator displays particular simulation steps and allows dialogue with the user, see Fig. 6. Especially in the area of agriculture, food and environment, where our customers are mostly not technically educated persons, it turned out that fair testing of correctness of all discovered business processes is an absolutely must.

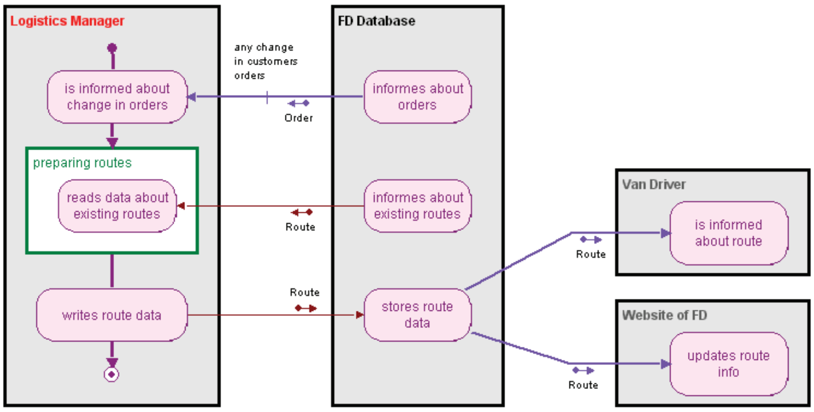

Fig. 5. Delivery planning business process

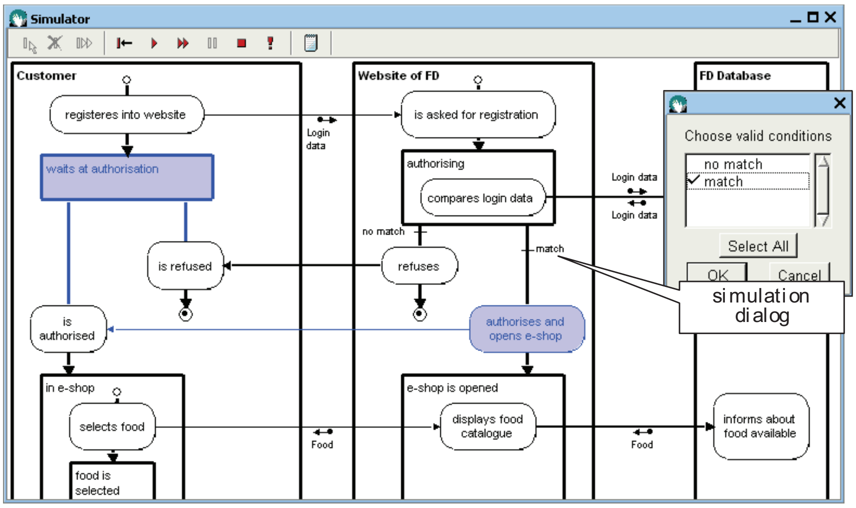

Fig. 6. One concrete simulation step in Craft.CASE

5. CONCLUSION

Currently there is not a 'standard solution' to the problem of gathering and representing business knowledge. The agricultural, food and environmental business systems are specially such cases, when we need ‘smart’ modelling tools, because processes and data are instantly changing and modifying through the whole life cycle of such systems. Our approach, described here, developed out of business experience and enhanced by graphic models with clear connection towards system development seems to be a promising candidate for such a standard. The approach we propose may serve not only as a tool for formal representation of modelled information, but also as we have demonstrated as a useful tool for communicating with developers and experts from the problem domain (managers, employees, etc.). The key advantages of BORM are its graphic models of knowledge representation, which provides easy and effective feedback. There are also clear rules how to progress through the system development process using this knowledge representation.

The number of projects, including projects in agriculture, food and environment area, executed in past 10 years gives us an important feedback. The clients say our analysis gives them a complex and context view of issues they did not see before. Clients appreciate BORM models having collection of business elements and their relationships being visualized and simulated together. Moreover, several clients use miscellaneous legacy Process Modelling Systems for historical reasons (e.g. EPC-based ARIS, for example). (EPC, 2009) However they prefer to analyse and design processes using BORM as well.

Our next work will concentrate on elaboration of the BORM, its concept transformations from and to other methodologies and research in the area of business process patterns.

Acknowledgments. This research (work) has been supported by Ministry of Education, Youth and Sports of Czech Republic under research programs number MSM6840770017 and MSM6046070904, and SGS projects number SGS10/267/OHK3/3T/13, SGS10/162/OHK3/2T/13 and SGS11/166/OHK4/3T/14.

References

1. Ambler, S. (1997) *Building Object Applications That Work, Your Step–By–Step Handbook for Developing Robust Systems Using Object Technology*, Cambridge University Press/SIGS Books, ISBN 0521648262.
2. Barjis, J., & Reichgelt, H. (2006). A Petri Net Based Methodology for Business Process Modeling and Simulation. In the proceedings of the *Fourth International Workshop on Modeling, Simulation, Verification and Validation of Enterprise Information Systems* (MSVVEIS), Paphos, Cyprus, May 23-24, 2006. ISBN: 972-8865-49-X
3. Craft.CASE (2009), *Business Process Modelling,* online: <http://www.craftcase.com>
4. Eriksson, H. and Penker, M. (2000) *Business Modeling with UML*, John Wiley and Sons, ISBN 0-471-29551-5.
5. EPC (2009), *Event-driven Process Chain,* online: <http://en.wikipedia.org/wiki/Event-driven_process_chain>
6. Goldberg, A. and Rubin, K. S. (1995) *Succeeding with Objects – Decision Frameworks for Project Management*, Addison Wesley, ISBN 0-201-62878-3.
7. Jacobson, I. (1992) *Object–Oriented Software Engineering – A Use Case Driven Approach*, Addison Wesley, ISBN 0-201-54435-0.
8. Knott, R. P., Merunka, V., Polak, J. (2000) Process Modeling for Object Oriented Analysis using BORM Object Behavioral Analysis, *Proceedings of Fourth International Conference on Requirements Engineering* ICRE 2000, Chicago, USA, IEEE Computer Society Press, ISBN 0-7695-0565-1.
9. Liu, L. and Roussev, B. (2006) *Management of the Object–oriented Development Process*, Idea Group Inc, ISBN 1591406048.
10. MDA (2009), *The Model Driven Architecture,* OMG – The Object Management Group, online: <http://www.omg.org>.
11. Merunka, V., Brozek, J., Nouza, O. (2008) Automated Model Transformations Using the C.C Language, Proceedings of the *International conference EOMAS 2008*, June 16 – 17, Montpellier, France, Springer LNBIP 2008, ISSN 1865-1348.
12. Simone A. J. H. and Graham, I. (1999) 30 Things that go wrong in Object Modeling with UML, chapter 17 in: *Behavioral Specifications of Businesses and Systems*, eds. H Kilov, B Rumpe, I Simmonds (Kluwer Academic Publishers), 237 – 257.