MOVING BUSINESS RULES FROM SYSTEM MODELS TO BUSINESS RULES REPOSITORY

Olegas Vasilecas[1], Evaldas lebedys[2]

Departament of Information Systems, Vilnius Gediminas Technical University, Vilnius, Lithuania

olegas@fm.vtu.lt[1], evaldas@isl.vtu.lt[2]

Abstract. The paper discusses business systems modelling and information systems development from business rules perspective. The way business rules influence different system models is analysed in the paper. The paper presents a method to create a whole business rules system model without separating business rules from models representing other aspects of the system: system structure, behaviour, actor's roles etc. The paper briefly discusses business rules represented in UML use case diagrams. A software system implementing the proposed method is presented in the paper. The paper shortly presents the results of the experiment performed on a use case model of a textile factory.

Keywords: business rules modelling, business rules repository, information systems, UML, use case

Received January 27, 2006 / Accepted April 10, 2006

1 Introduction

Business rules approach is important to systems modelling because business rules are the essential components of every business system. Business rules are the integral part of every business system. Business rules influence all aspects of the business systems [10]. Business rules approach is also important in automated information systems implementation. Every system can be represented in a single model or several models that represent different aspects of the system. In most cases, full and detailed system model includes several models that can be manipulated separately, because business systems are complex and are modelled from different points of view. Modelling a system from several separate but related viewpoints permits it to be understood for different purposes [7]. Complete business system model should also include business rules model. There is no generally accepted approach for defining or representing business rules [6]. Different modelling languages can be used to represent business rules: UML, structured English sentences, specialised rule modelling languages, program code and others [5]. UML became a widely used modelling language for systems modelling [8]. UML was created for software systems analysis and design. However, later UML was used to model information systems and business systems. It may be difficult to create full and detailed business system model using UML because UML is not expressive enough to model all the perspectives of the system of interest. UML does not have facilities to create full business rule model using one notation but business rules represented in different UML diagrams may compose a business rules system. The problem appears when a whole business rules system has to be represented and analysed as one, because business rules are spread all over the system models. Our proposal is to move business rules to business rules repository and use it for representing business rules formally for automated implementation and informally for business audience. The relationship between models and rules in repository ensures traceability. The paper discusses only a part of the business rules system model represented in UML use case diagrams. The software system prototype realising the proposed method is presented further.

The rest of the paper is organised as follows. Section 2 briefly describes business systems modelling using UML and the way business rules appear in different UML models. Section 3 shortly discusses business rules in UML use case diagrams. Section 4 analyses the structure of the business rules repository for rules represented in use case models. Section 5 shortly presents the created software system prototype realising the proposed method. Section 6 concludes the paper.

2 Business systems modelling from business rules perspective

Generally, a system can be described in a single model or using different models that represent different aspects of the system. In most cases, full and detail system model includes distinct models that can be manipulated separately, because business systems are complex and are modelled from different points of view. Concept “model” is used here as a representation of a part of the function, structure and/or behaviour of system of interest [6]. A model is generally regarded as a representation of reality [14]. Different modelling languages can be used to represent specific aspects of the systems being modelled. Modelling a system from several separate but related viewpoints permits it to be understood for different purposes [7]. System structure, behaviour, functions, actor obligations and competence spheres have to be modelled to create full business system model. Complete business system model should also include business rules model. Business systems structure, behaviour, organisation of functions and other aspects were modelled for a long time, but rules that constrain different aspects of business systems were neglected mostly [1]. These constraints are expressed as business rules [3]. There were many concepts of business rules proposed recently. In this paper we use the concept of business rule as stated in [5]: business rule means a statement that defines or constrains some aspect of the business. Concept “business” in this context is used as an abstraction that refers to any subject of any scope to which model driven architecture or UML modelling is or could be applied [5]. This means that concept business in this paper is used to specify any type of activity having its goals, using its resources, processing some processes and achieving some results.

Business rules appear both in static and dynamic business systems models. Business rules represented in static and dynamic system models are of different classes. There are many different classes of business rules. Business rules of different classes can be represented differently [4]. In principle, all business rules can be represented using natural language, but in this case business rules model becomes informal and ambiguous [2]. Many graphical modelling languages and diagrams can be used to model business rules: UML diagrams, ERA diagrams, IDEF methods diagrams, conceptual graphs [11]. Non-graphical techniques can also be used to represent business rules: decision trees, decision tables, first order predicate logic, rule templates. We have chosen UML diagrams for analysis in this paper because UML is the most popular modelling language used to model business systems, information systems and software systems [8]. The Unified Modelling Language is a general-purpose visual modelling language that is used to specify, visualize, construct and document the artefacts of a software system [7].

All UML diagrams can be classified into three different classes [12]:

- diagrams describing the roles and obligations of system users (Use Case diagrams);
- diagrams describing structural system aspects (class and object diagrams);
- diagrams describing the internal and external behaviour of system (state transition diagrams, sequence and collaboration diagrams).

Business rules in Use Case diagrams mostly appear as statements describing system actor’s competence boundaries and obligations. Business rules in Use Case diagrams are represented relating system functions with domain actors. It is important to declare how tasks are assigned to actors: are they mandatory or non mandatory. It influences how business rules are formulated. Obligatory mandates are assigned using rules of form “… has to do …” and non obligatory mandates are assigned using rules of form “… can/may do…”.

Sequence and collaboration diagrams are used to describe how system users perform their tasks. Sequence and collaboration diagrams contain business rules describing the internal and external behaviour of business system [9]. Sequence and collaboration diagrams include business rules describing the exact order of actions for a user to be taken to perform specific task.

State transition diagrams are used to specify the sequences of changes of states of business objects. State transition diagrams represent the changes of system objects states according to different conditions. Event-Condition-Action (ECA) rules are mostly represented in state transition diagrams.

Class diagrams are used to represent classes of objects. Classes are aggregates of objects, which are used as variables in predicates when specifying business rules. In other words, classes are sets of business objects. Class model also include specific business rules. These are constraints of business objects, properties of relationships between business objects.

UML activity diagrams can be used to model the logic of the operations captured by a use case or a few use cases. Activity diagrams are similar to flow charts and data flow diagrams in structured development. Activity diagrams represent both the basic sequence of actions as well as the alternate sequence of actions. As activity diagrams are special kind of state transitions diagrams they use some of the same modelling conventions. The notation of activity diagrams is analogous to state transition diagrams notation. ECA rules are mostly represented in activity diagrams (Fig. 1).

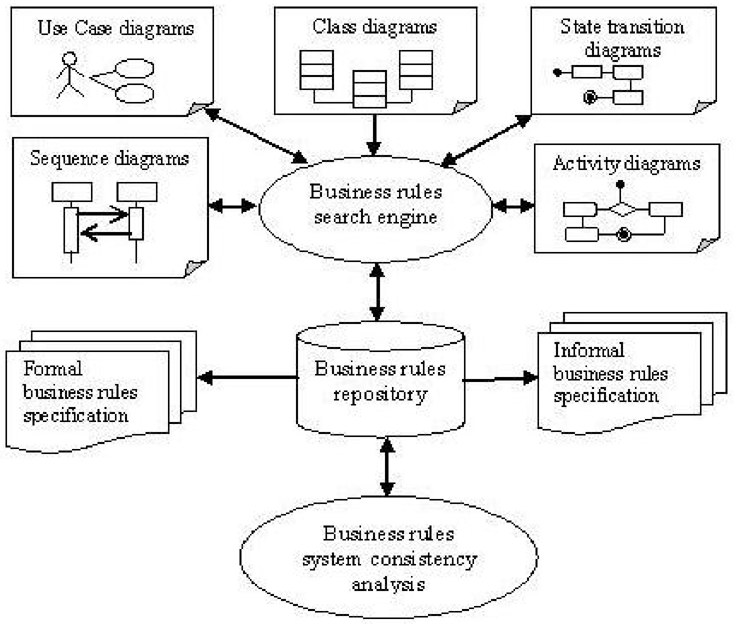

Use Case diagrams
Class diagrams
State transition diagrams
Sequence diagrams
Business rules search engine
Activity diagrams
Formal business rules specification
Business rules repository
Informal business rules specification
Business rules system consistency analysis

Figure 1. The transformation of business rules

3 Business rules in use case models

The way to get business rules from business systems use case model is proposed further. Sybase® PowerDesigner® 9.0 was chosen as a tool for analysing the rules specified in the created use case model. The created use case model is stored in XML file. Data from XML file is copied to temporal storage using the created software system prototype (Fig. 2). The search of business rules represented in use case model can be performed directly in XML file, but in order to accelerate the process of business rules search, the data is copied to temporal storage. First of all the following components of business rules are looked for and copied to temporal storage:

- Actors;
- Use cases;
- Relationships between actors and use cases and,
- Stereotypes of relations;

Stereotypes of relationships are used to specify the roles of actors. Business rules components are joined and business rules are composed in the following step. The roles of business actors are used to form predicates of a form ROLE(ACTOR X, USE CASE Y).

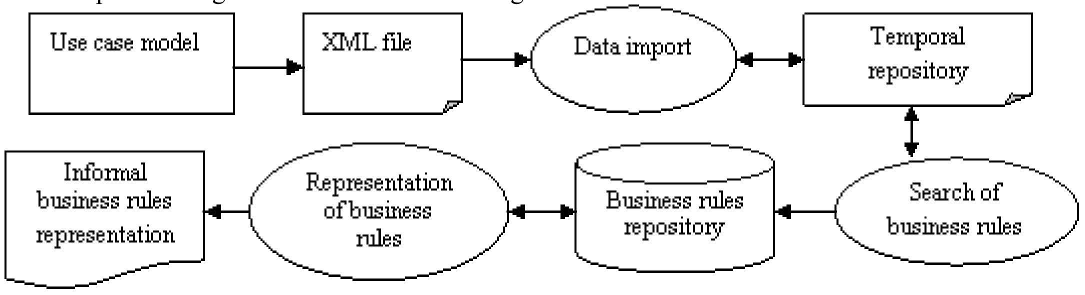

Use case model
XML file
Data import
Temporal repository
Informal business rules representation
Representation of business rules
Business rules repository
Search of business rules

Figure 2. The import of business rules from use case model to business rules repository

Business systems are dynamic and react to the changes in its environment and other business systems. Business systems models are being changed according to the changes in business systems. Business rules express the external and internal business systems behaviour and are changing also: new business rules become a part of business rules system, some business rules become invalid and are eliminated from rule system. Respectively changes to the business rules system are implemented in information system.

Changes in the business systems impact all business systems models because business rules influence all aspects of business systems. It is important to ensure automatic realisation of business systems changes in information systems. The changes made in use case model are implemented in the business rules repository automatically when the data import is repeated [13].

4 Repository for business rules represented in UML use case diagrams

As mentioned above business rules influence all aspects of business systems and appear in almost all business systems models. It is obvious that business rules differ and business rules are classified according to the different classifiers. Only business rules represented in UML use case models are discussed in this paper. Business rules represented in UML use case diagrams can be expressed formally using predicates of a form ROLE(ACCTOR X, USE CASE Y) [11]. Business rules repository is used to store the information of components of business rules. Both formal and informal business rules expressions expressed in natural language are composed of components of business rules stored in the repository. Figure 3 shows the structure of the created business rules repository.

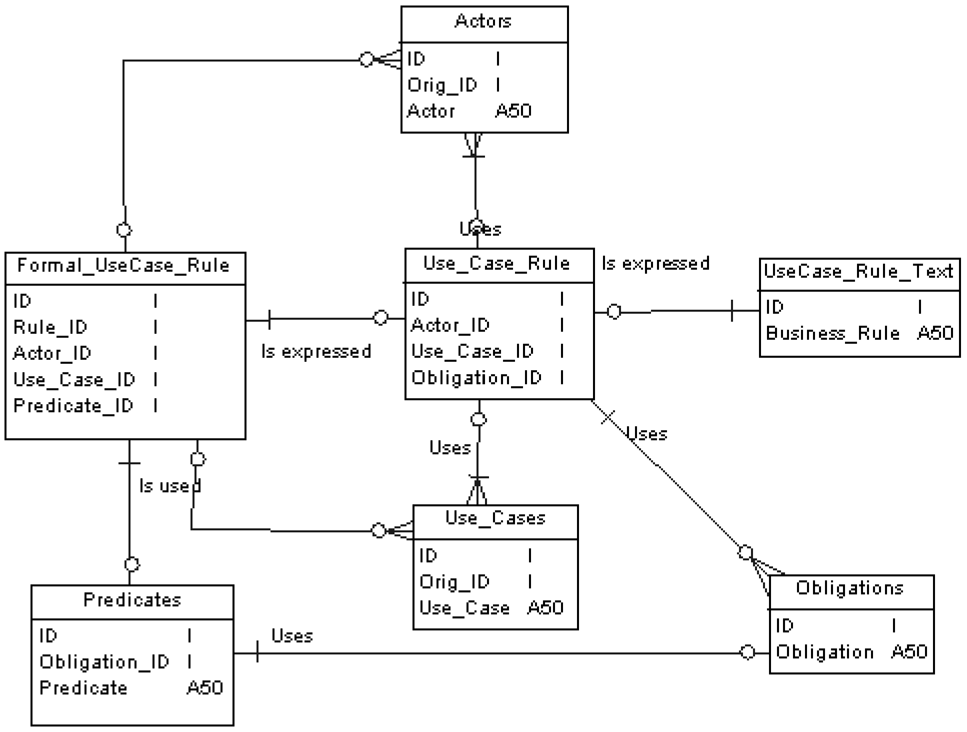


Figure 3. The structure of the business rules repository

Table “Actors” is used to store the information of actors represented in use case model. The purpose of table “Use Cases” is to store the information of use cases represented in the use case model of the business system of interest. The information of business systems actor’s roles and obligations is stored in the table “Obligations”. Table “Predicates” contains the information of predicates, which are formed, on the basis of actor’s roles and obligations data. The relationships between business rules components are stored in table “Use_Case_Rule”. The information stored in this table is used to express business rules formally and informally. Formal business rules expressions are stored in the table “Formal_UseCase_Rule”. Business rules expressed in natural language are stored in table “UseCase_Rule_Text”.

Figure 3 represents only a part of the business rules repository used to store the information of business rules represented in UML use case diagrams. As business rules represented in other business systems models are out the scope of this paper, a full business rules repository is not discussed in detail further.

5 Transferring business rules from models to rules repository

The experiment was performed to test the created software system prototype. Textile factory domain was chosen as a business system for analysis. Figure 3 represents a fragment of a whole use case model. Only use cases related to the process of manufacturing a particular item are represented in Figure 3. The created software system prototype was used to copy business rules, represented in Figure 4, from use case model to business rules repository. Business rules were expressed in natural language after the business rules import was performed.

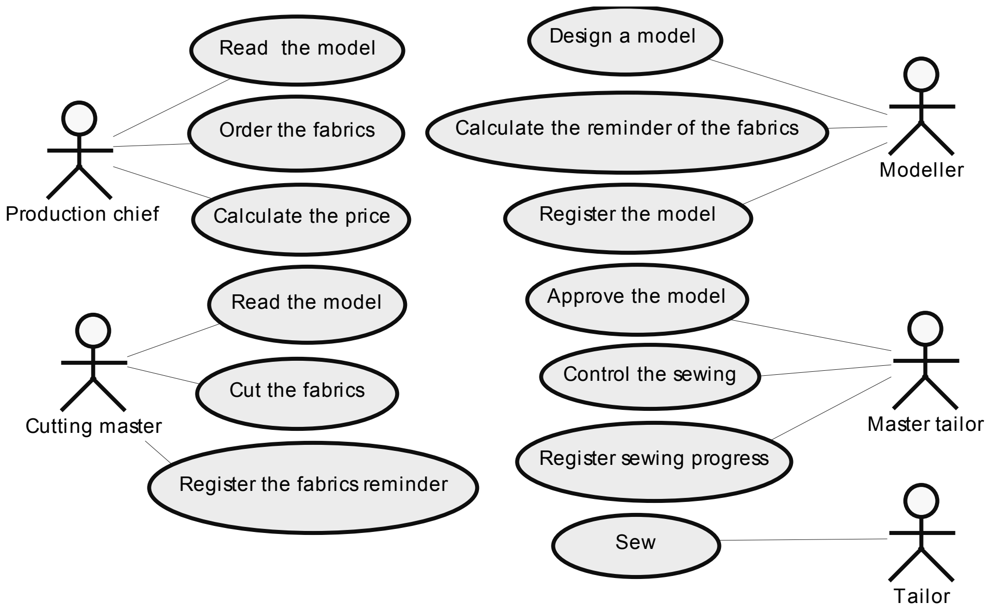
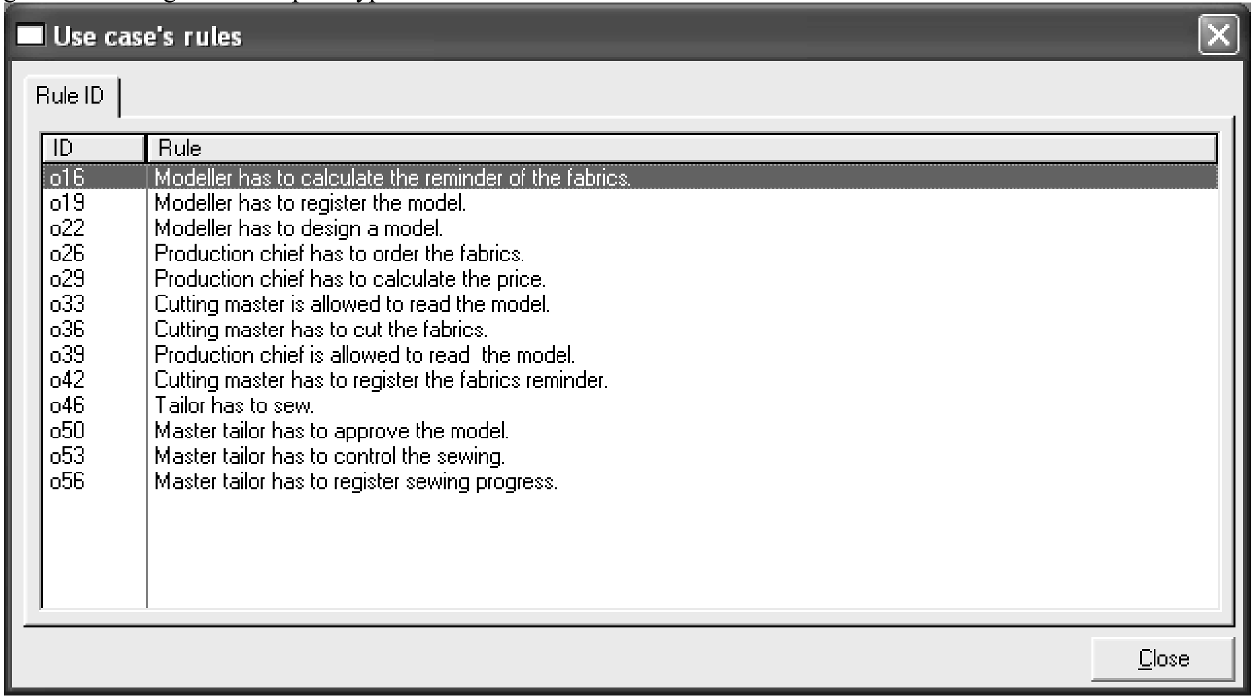


Figure 4. A fragment of the use case model

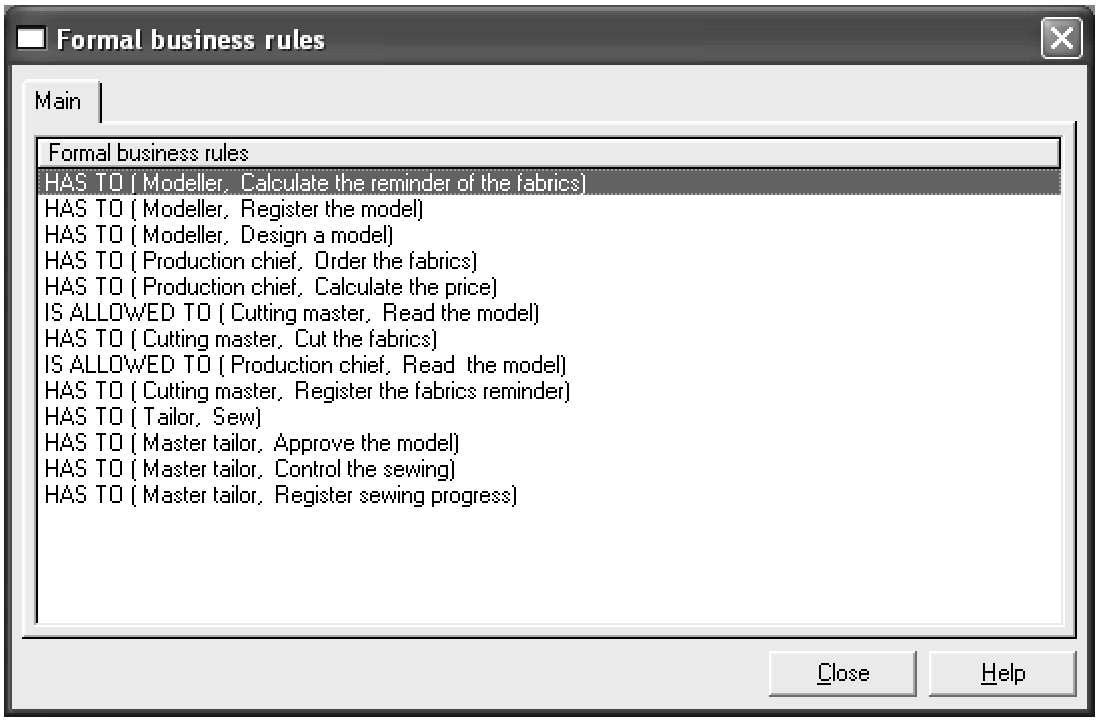
Business rules stored in the rules repository are represented in natural language. Figure 5 represents a list of the business rules expressed in natural language generated using software prototype. Informal business rules expressions are intended for business audience because this is the most understandable way of representation of business rules.



ID	Rule
o16	Modeller has to calculate the reminder of the fabrics.
o19	Modeller has to register the model.
o22	Modeller has to design a model.
o26	Production chief has to order the fabrics.
o29	Production chief has to calculate the price.
o33	Cutting master is allowed to read the model.
o36	Cutting master has to cut the fabrics.
o39	Production chief is allowed to read the model.
o42	Cutting master has to register the fabrics reminder.
o46	Tailor has to sew.
o50	Master tailor has to approve the model.
o53	Master tailor has to control the sewing.
o56	Master tailor has to register sewing progress.

Figure 5. A list of the business rules represented in use case model

Different classes of business actor's roles and obligations are distinguished on business rules import to business rules repository. These classes of roles and obligations are used to form predicates to express business rules formally. Two types of business actors roles are distinguished on import of business rules represented in the use case model presented above: "has to" and "is allowed to". Although the roles are not represented in use case model, the chosen modelling tool allows specifying actor roles. A list of the business rules expressed using predicates is generated using software system prototype (Fig. 6).



```
Formal business rules
HAS TO ( Modeller,  Calculate the reminder of the fabrics)
HAS TO ( Modeller,  Register the model)
HAS TO ( Modeller,  Design a model)
HAS TO ( Production chief,  Order the fabrics)
HAS TO ( Production chief,  Calculate the price)
IS ALLOWED TO ( Cutting master,  Read the model)
HAS TO ( Cutting master,  Cut the fabrics)
IS ALLOWED TO ( Production chief,  Read  the model)
HAS TO ( Cutting master,  Register the fabrics reminder)
HAS TO ( Tailor,  Sew)
HAS TO ( Master tailor,  Approve the model)
HAS TO ( Master tailor,  Control the sewing)
HAS TO ( Master tailor,  Register sewing progress)
```

Figure 6. A list of the business rules expressed formally

6 Conclusions

Business rules are the integral components of every system and influence all aspects of business systems. There are different classes of business rules and different modelling techniques can be used to represent business rules of different classes. There is no universal modelling language suitable to represent all aspects of the systems. It is hard to create a full business rules system model because business rules appear in all systems models and cannot be separated from these models. The paper presented a method for moving business rules from systems models to business rules repository. A software system prototype and business rules repository was created to implement and test the proposed method. The paper presents a part of the software systems used to manage business rules represented in UML use case diagrams. An experiment was performed and presented in the paper.

References

[1] Business Rules Group. Defining Business Rules ~What Are They Really?, 1995. URL: http://www.businessrulesgroup.org/first_paper/BRG-whatisBR_3ed.pdf.

[2] Date, C. J. Business Rules and Logic ~ What Exactly Is a Business Rule?. // Business Rules Journal. 4 (12), 2003. URL: http://www.BRCommunity.com/a2003/b170.html.

[3] Liu, Y., Pluempitiwiriyawej, C. and others. A Rule Warehouse System for Knowledge Sharing and Business Collaboration. Technical Report, TR01-006, University of Florida, 2001, URL http://www.cise.ufl.edu/tech-reports/tech-reports/tr01-abstracts.shtml.

[4] Morgan, T. Business Rules and Information Systems: Aligning IT with Business Goals, Addison-Wesley Pub Co, 2002, p. 30-97.

[5] Object Management Group (OMG). Business Rules in Models - Request for Information (BRWG RFI1), 2002. URL: http://www.omg.org/docs/ad/02-09-13.pdf, [retrieved on 2006.01.15].

[6] Object Management Group (OMG). Business Semantics of Business Rules - Request For Proposal, 2003. URL:

http://www.omg.org/docs/br/03-06-03.pdf, [retrieved on 2006.01.09].

[7] Rumbaugh, J., Jacobson, I., Booch G. The UML Reference Manual, Addison-Wesley, p. 13-38, 1999.

[8] Shen, W., Compton, K., Huggins, J. K. A Toolset for Supporting UML Static and Dynamic Model Checking. // In proc. of the 26th International Computer Software and Applications Conference (COMPSAC 2002), Prolonging Software Life: Development and Redevelopment, Oxford, England, IEEE Computer Society, p. 147-152, 2002.

[9] Sparks, G. An introduction to modelling software systems using the Unified Modelling Language: The Dynamic Model, 2001. URL: http://www.sparxsystems.com.au/WhitePapers/The_Dynamic_Model.pdf, [retrieved on 2005-12-15].

[10] Vasilecas, O., Lebedys, E., Laucius, J. Formal methods for representation of business rules specified using UML. // In: R. Simutis (eds.). Proceedings of the International Conference "Information Technologies for business - 2005", Kaunas: Technologija, p. 41-47, 2005.

[11] Vasilecas, O., Lebedys, E. Business rules repository for business rules represented using UML // In R. Rachev, A. Smrikarov (Eds.). Proc. of the International Conference on Computer Systems and Technologies "CompSysTech ‚05", Varna, Bulgaria, p. II.5-1 – II.5-6, 2004.

[12] Vasilecas, O., Lebedys, E. Employment of formal business rule specification in IS development process. // In Proc. of the Conference "Information Technologies‘2005“, Kaunas: Technologija, Lithuania, XIVi, p. 636-643, 2005.

[13] Vasilecas, O., Lebedys, E. Repository for Business Rules Represented in UML Diagrams // Izvestia of the Belarusian Engineering Academy, Vol. 1 (19)/2, p. 187-192, 2005.

[14] Whitman, L., Huff, B. Issues Encountered Between Model Views. Flexible Automation and Intelligent Manufacturing 1998. Portland, OR, p. 117-130, 1998.