ISSN 1392 – 124X INFORMACINĖS TECHNOLOGIJOS IR VALDYMAS, 2004, Nr. 2(31)

AUTOMATIC ENFORCEMENT OF BUSINESS RULES AS ADBMS TRIGGERS FROM CONCEPTUAL GRAPHS MODEL

Irma Valatkaite, Olegas Vasilecas

Information Systems Scientific Laboratory, Vilnius Gediminas Technical University
Sauletekio al. 11, Vilnius, Lithuania

Abstract. The business rules approach to business information systems development has already gained a lot of attention from both research communities and commercial organizations. In this paper we survey the current situation and point out the lack of standards in business rules technologies. We elaborate on our approach to business rules modeling by conceptual graphs and enforcement tasks which is presented in our earlier publications, now focusing on trigger generation component, which employs active database management systems as the rules enforcement technology, also allowing the possibility of rules sharing or exchange through the business rules repository in XML format.

1. Indroduction

A business rules approach to the business information systems (IS) development has achieved a lot of attention and already has a steady niche with a strong motivation behind. Since business rules are at the heart of the business giving the business rules the proper attention at organization management level (as a part of organization knowledge management activity) enables business organization to manage itself by making visible the individual policies that the organization puts in place as a guide to accomplish its goals. By manipulating the business rules the organization enables to fine-tune itself to the ever changing business environment thus becoming faster and more responsive than its competitors.

The underlying business IS should be there to support this ultimate goal of business organizations to be fast and adaptive. The business rules approach to IS development is intended to solve this task and enable business IS developers to shorten the path from business requirements to the actual implementation and also give the possibility to business users to maintain their business rules set by themselves thus achieving the intended speed of responsiveness and adaptability. Since the business users are the owners of the business rules it is a rather straightforward solution 0, 0. However, the solution requires a good deal of technology behind.

The research on business rules approach and the resulting technology (business rules tools) may be characterized by their intended purpose 0:

- Organization management: to provide an environment where business rules are captured, documented, secured, distributed to the relevant parties within both the business and technology organizations, and modified as the business evolves.
- Enforcement: to integrate business rules into business application's architecture, but to hold business rules separately from the database and processing code.
- Excavation: mine through existing IS to uncover business rules held within application code or textual documents with the purpose to transform business rules uncovered from code into something that is meaningful to people.

The three purposes (or characteristics) of technologies and tools are separate research topics, however, as technologies and tools evolve they are intermingled and tools emerge that serve for two purposes simultaneously. Most often technologies and tools are created that are suitable for the first two tasks: organization management and enforcement. This is because these two require the same basic component – the business rules repository which serves as the central location of storing business rules metadata. The excavation of business rules is a different activity: (1) organization management and enforcement are the activities that support daily business while excavation is a rather one-time activity whose results should be used as the input to the management and enforcement; (2) excavation requires completely different algo-

rithms and implementation architectures than management and / or enforcement technology. Thus it is not reasonable to combine excavation with the other two. However, some tools successfully combine all three (e.g., Infrex 0; the survey of tools is given below in Section 0).

The business rules repository is based on the conceptual business rules model. The uniform and consistent conceptual model of the domain at hand is the commonly agreed necessity of all software engineering disciplines. The conceptual business rules model characteristics (usability, consistency, etc.) have a straight reference to the modeling language used. The main characteristics of the modeling languages intended for business rules modeling is the ability to represent business rules precisely and without the possibility of ambiguity, understandable by business users but also usable by IS developers, suitable for automatic business rules engines thus extending the usability of the model 0.

The number of modeling languages were proposed to use for business rules modeling (the short survey is given in 0), the commercial products employ their own languages, but there is no consensus yet on the technology standard. We propose to use conceptual graphs 0 for this purpose and argue that conceptual graphs meet the above stated requirements. However, the lack of supporting technology and tools is an obstacle for the wide spread usage of the language.

In our research focusing on organization management and business rules enforcement tasks we present a framework for the business IS development. We propose to use the conceptual graphs as a modeling language for the conceptual business rules model which is then employed at two different business IS development levels – conceptual and implementation, and which enables automatic enforcement of business rules by the means of active databases triggers. In this paper we are focusing on the implementation level of the framework – the automatic business rules enforcement code generation.

The rest of the paper is organized as follows. The overview of the related work is given in Section 0. The basic idea of the framework and the modeling technique is briefly covered in Section 0 (the ideas being introduced in 0 and later elaborated in 0, 0, 0). In Section 0 the description of the implementation component of the framework is given with the example business rule. Finally, the conclusions are drawn and the further research directions are given in Section 0.

2. Related work

There are a number of different business rule management and enforcement tools available today. The common component of the majority of tools is the business rules repository which is later used for different tasks. However, as it was mentioned above, they employ different, sometimes very specific modeling languages. As for the functionality that the tools offer – it is mainly centered on the following tasks:

- Input and change rules – rule editors
- Store rules – rule repository
- Enforce rules – rules engines or similar mechanisms

The rules enforcement component of the tools offers the invisible for the business user reference to the actual enforcement mechanism achieving the required automatic dependency of the business rules implementation on declarative business rules statements.

Blaze Advisor offers the complete process for designing, running, and maintaining e-business applications. Blaze Advisor consists of the following 5 components:

- Builder – a rule creating tool targeted for developers;
- Innovator – a rule management and maintenance tool targeted for business users;
- Rule Engine – a scalable processing engine that determines and executes the control flow of rules, works together with the Rule Server;
- Rule Server – a dedicated rule server which supports rule execution, session management, scheduling, and dynamic load balancing.

Blaze Advisor uses its unique Structured Rule Language for input of business rules; decision trees and decision tables can be employed as well.

Infrex 0 is another type of tool – while Blaze Advisor is a totally stand alone application, Infrex can be embedded into applications written in C/C++/Java/C#. Infrex uses classes and variables of the application with the support for high level operators for rules specification. Rule Translator component generates C/C++/Java/C# code which is compiled and linked with the application to create an executable. The executable has the rules to be called at run-time, through the engine. Thus the adaptivity feature is achieved by the ability of the tool automatically to create executable code from the rules specification. However, the language for rules specification is not suitable for business users because it directly operates with classes and variables which are not exactly the business terms.

QuickRules 0 is a business rules management system which allows inserting and editing business rules using the specific QuickRules Rules Markup Language. Rules are stored in XML format and may be executed by Business Rules Engine component.

ILOG rules are of ECA (Event, Condition, Action) type and consist of three parts: header, condition and action specifications. The rules can be defined using BAL (Business Action Language), IF-THEN-

ELSE rule format and TRL (Technical Rule Language).

The above mentioned tools have their own business rules engine, therefore, it is a must for the business organizations to buy the respective product suit in order to be able to use their offerings. The opposite solution would be to use a wide spread technology of active database management systems (ADBMS) for rules repository and as an enforcement engine. The big advantage of this approach is that the majority of business organizations are already using various ADBMS for storing their business data.

The Dulcian company (an Oracle consulting firm that specializes in data warehousing, systems development and products that support the Oracle environment) is working in this direction offering BRIM – Business Rules Information Manager which offers the rules editing, designing and implementation functionality. The rules editing is done over two steps – analysis rules are entered using weakly structured RuleSpeak language (by Business Rules Solutions 0) and implementation rules are specified using UML 0 class and activity diagrams. The mapping between analysis and implementation rules may be done by business users manually associating rules with classes, states and diagrams 0. The automatic code generation component creates text files that either creates triggers on tables or alters the existing triggers on tables. The technology is completely Oracle-centered.

The survey given above is by no means complete but it highlights the trends within the field of interest. The lack of standards is clear – for business rules modeling languages, repositories format, architectures. It is not possible to exchange business rules among different products or present to the business users more or less standard rules language. However, the promising step in business rules repository is the usage of XML as the business rules storage format which would enable the business rules sharing and exchange.

3. The proposed business IS development framework

In this paper we present the framework for business IS development which makes use of the conceptual graphs as a conceptual modeling language and employs active databases triggering mechanism for the rules enforcement.

As presented in 0, the major components of the proposed framework are the following:

1. *Conceptual model of the business rules and business domain*

The uniform and consistent conceptual model of the business domain is the commonly agreed necessity of all software engineering disciplines. We propose to model business domain with explicit attention to business rules using conceptual graphs (CG) 0 as the modeling language. The advantages of CG as a business rules modeling language are thoroughly discussed in 0, 0 and 0, the main of which is the unique characteristic of CG to be used as a formal and informal language simultaneously and appropriate visual and textual notations.

The business rules modeling template using CG is presented in 0. It is proposed to model business IS using the CG element Knowledge Base that consists of the following elements:

1. Concept Hierarchy – to represent entities (or terms)
2. Type Hierarchy – to represent relations
3. Catalog of Individuals – to represent facts
4. Outermost Context – to represent system boundaries
5. Rule Base – a proposed extension to Knowledge Base to represent business rules

Rule Base element is designed specifically for business rules modeling. The approach presented covers the business rules that fall under dynamic assertions type (a survey on business rules types and taxonomies is given in 0) by employing the rule metamodel which is based on ECA (Event, Condition, Action) rule paradigm from active databases research field 0. The other type of business rules – structural assertions must be modeled using other CG Knowledge Base elements – Concept Hierarchy, Relation Hierarchy, and Catalog of Individuals thus forming business IS ontology.

2. *Active databases for business rules implementation*

Business rules are implemented within active database, using triggers and stored procedures. The ability of active databases (triggers) to demonstrate reactive behavior makes active databases convenient for implementation of business rules. Since the implementation of the business rules resides in the database itself, modifications are application independent and can be performed without accessing the application logic.

3. *Trigger generation component*

The automatic implementation is at the heart of the proposed framework. Implementation of business rules (as triggers) is performed and managed automatically by the special trigger generation component thus achieving the required flexibility and automated business rules implementation.

The detailed description of the trigger generation component is given in Section 0.

4. *Business rules repository*

A repository must adhere to the principles of component architecture. The goal is to have a repository with the following characteristics:

- strictly defined visual interface for rules inserting / editing;
- automatic trigger generation component to ensure the consistency between the rules in repository and their implementation in active database management system;

In this paper we do not address the first characteristic but rather focus on the automatic implementation issue. Although it is not a mandatory part of repository itself, such a component facilitates the usage of the repository. Later having underlying technology and implementation in place the focus could be shifted to the rules editing interface thus enabling the business rules owners to use the repository to its maximum extent.

4. Automatic implementation: trigger generation component

In **Figure 1** the flow of execution of the trigger generation component is presented: the business rules model serves as the input for creating the repository which again serves as the input for ADBMS triggers generation.

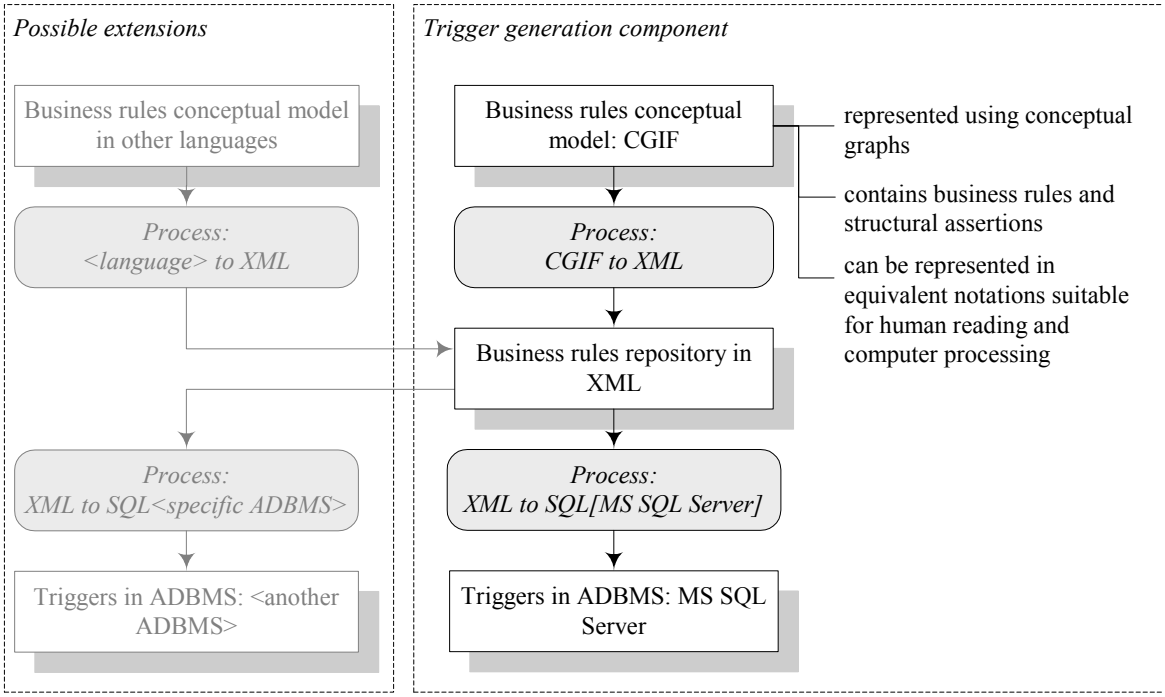

Figure 1. The trigger generation component: from CGIF to SQL

In the current research as shown in **Figure 1** we focus on the business rules model represented in conceptual graphs CGIF notation (Conceptual Graphs Interchange Format, please refer to 0 for the draft standard of conceptual graphs and its notations including CGIF). However, the architecture is not limited in that sense and the extensions are possible (shown in **Figure 1** to the left):

- Usage of another modeling language for business rules conceptual model;
- Usage of another underlying active database management system (because of syntactic differences of trigger definition in different ADBMS).

The XML based business rules representation thus is the central element allowing adding flexible extensions and serving as the business rules repository.

The component consists of the two processes:

1. *Process CGIF2XML*

The scope of the process is a transformation of business rules representation from conceptual graphs model in CGIF format to the intermediate business rules representation in XML. The latter can be used as the basis for the business rules repository.

In current implementation the input format is limited to CGIF; the output format adheres to the rules representation using XML structure.

2. *Process XML2SQL*

The scope of the second process is a transformation of rules contained in intermediate format of XML based structure to the actual implementation representation – namely, active databases triggers.

In the current implementation the Microsoft SQL Server ADBMS trigger definition syntax is used.

4.1. Example business rule and the resulting trigger

To illustrate the flow of execution we present the example business rule, its intermediate representation in XML, and the resulting MS SQL Server ADBMS trigger:

1. *Business rule in natural language*

If manufacturing order is completed and it is not an internal order (the identification number is greater than 17 which marks the limit of the internal orders), its finishing date is set to today.

2. *In conceptual graphs visual notation*

The diagram in Figure 2 represents the business rule using the standard conceptual graphs nodes – concepts (drawn as rectangles), conceptual relations (as ovals), actors (as diamonds) and links (as arrows connecting concepts with actors or conceptual relations).

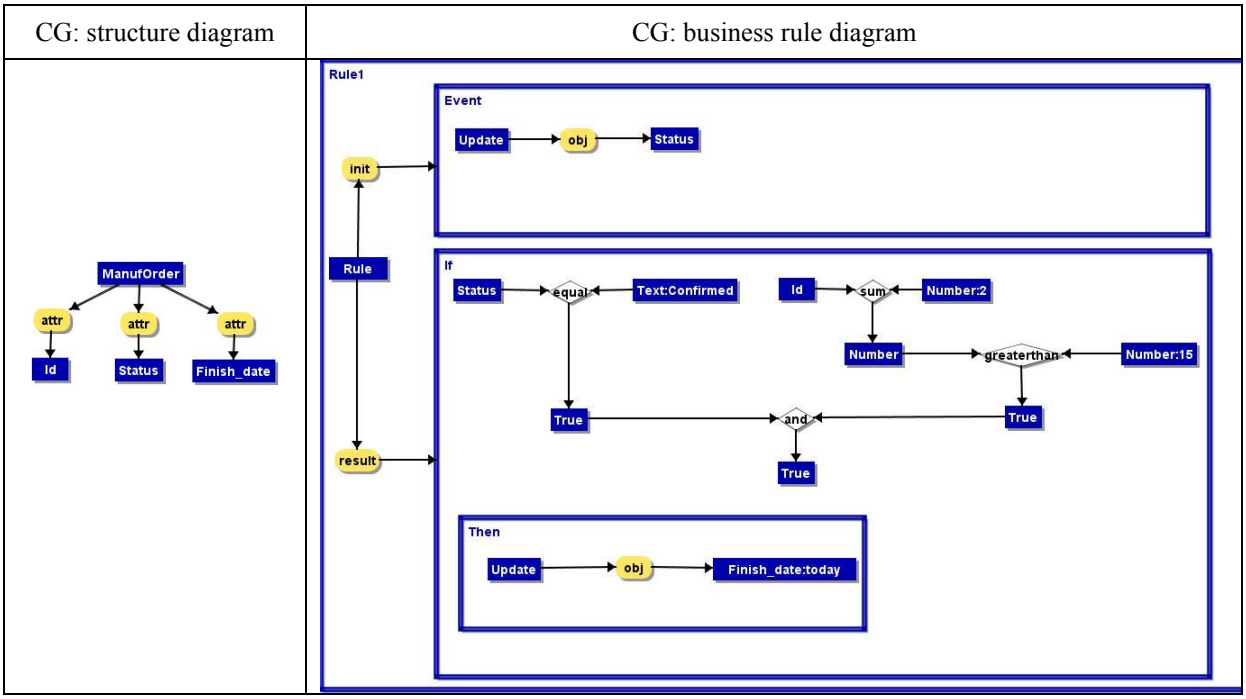

Figure 2. Business rule modeled in CGIF using business rule template

The diagram is created using the conceptual graphs editor CharGer v3.0b 0. The current version offers functionality to draw conceptual graphs diagrams, get the respective representation of the diagrams in CGIF and near natural English language, export diagrams to various picture formats, etc. In our research we rely on CharGer for generating CGIF representation of the diagrams.

While modeling the business rule we employ the following rule template:

```
Type Rule(*x) is T(?x) –
    <-(INIT)<-[Event:*y]
      ->(RESULT)->[If:*z[Then:*w]]
```

We have proposed the template to model business rules in 0 with the goal to standardize the business rule representation in order to make the automatic generation of the triggers possible.

3. *In CGIF:*

The below presented CGIF representation of the business rule is generated by CharGer tool. Comments are added manually in order to increase the readability of the representation (comments begin by double slash '//').

```
//Rule name:
[Rule1:'']
//Template begins:
[Rule*a:'']
//Event part:
[Event*n:'']
[Status*o:''][Update*p:''](obj?p?o)]
(result?a?b)(init?a?n)]
//Conditional action part:
[If*b:'']
[True*c:''][True*d:''][True*e:'']
[Number*f:''][Number*g:'15'][Status*h:'']
[Text*i:'Confirmed'][Id*j:''][Number*k:'2']
[Then:'']
[Update*l:'']
[Finish_date*m:'today'](obj?l?m)]
<equal?h?i|?e>
<sum?j?k|?f>
<greaterthan?g?f|?d>
<and?d?e|?c>]
```

4. *In XML:*

The XML representation generated using the above presented CGIF representation is given below. Currently the unique XML structure created specifically for this project is used. However, it is reasonable to adapt the XML structure to the emerging standards, for example, rules markup language RuleML by Rule Markup Initiative 0 or to provide transformation to and from RuleML (the Initiative develops a modular RuleML specification and transformations from and to other rule standards/systems).

```
<?xml version="1.0" encoding="iso-8859-1"?>
<CGIF>
//Entity definition (from structural part of the model):
    <Tables>
        <Table Name="ManufOrder">
            <Column Name="Status"/>
            <Column Name="Id"/>
            <Column Name="Finish_date"/>
        </Table>
    </Tables>

//Rule name:
    <Rule Name="Rule1">

//Event part:
        <Event Column="Status" MainTable="ManufOrder" Operation="Update">
            <Condition>
                <Actor Name="jkf" Type="sum">
                    <In1 Name="" Type="Id" Value=""/>
                    <In2 Name="" Type="Number" Value="2"/>
                    <Out Name="" Type="Number" Value=""/>
                </Actor>
                <Actor Name="hie" Type="equal">
                    <In1 Name="" Type="Status" Value=""/>
                    <In2 Name="" Type="Text" Value="Confirmed"/>
                    <Out Name="" Type="True" Value=""/>
                </Actor>
                <Actor Name="gfd" Type="greaterthan">
                    <In1 Name="" Type="Number" Value="15"/>
                    <In2 Name="jkf" Type="Number" Value=""/>
                    <Out Name="" Type="True" Value=""/>
                </Actor>
                <Actor Name="dec" Type="and">
                    <In1 Name="gfd" Type="True" Value=""/>
                    <In2 Name="hie" Type="True" Value=""/>
                    <Out Name="" Type="True" Value=""/>
                </Actor>
            </Condition>
        </Event>

//Conditional action part:
        <Action>
            <Block Column="Finish_date" Operation="Update"    Value="today"/>
        </Action>
    </Rule>
</CGIF>
```

5. *Resulting trigger:*

The resulting trigger implementing the business rule is shown below. The trigger is generated using MS SQL Server trigger definition syntax.

```
CREATE trigger Rule1 ON ManufOrder
FOR UPDATE
AS
IF UPDATE(status)
BEGIN
  if((SELECT status FROM INSERTED)='Confirmed' and (SELECT id FROM INSERTED)+2>15 )
    BEGIN
      UPDATE ManufOrder
      SET Finish_date=getdate()
      WHERE id=(SELECT id FROM INSERTED)
  END
END
```

5. Conclusions and future research

In the research of the business information systems development the business rules approach has achieved a lot of attention and already has a steady niche with a strong motivation behind. The usefulness of the approach and its advantages over the traditional IS development approaches call for the technology standards. Despite the work done towards standardization there is still a way to go – the commercial products use their unique modeling languages for business rules, most of their rules processing and enforcement engines are stand alone, even the embeddable ones use their unique rules representation format. There are initiatives such as Rule Markup Initiative, OMG, etc which take the steps in unifying business rules and technologies around business rules.

In our research focusing on the business rules that fall under ECA paradigm we aim towards employing the widely spread technology of active databases and in the series of papers we show that it is feasible and possible to model business rules using conceptual graphs. Such a model having representations in visual form, textual form – linear form and CGIF, and the possibility to be translated to the near natural English language is useful both at conceptual and implementation levels.

In this paper we presented the trigger generation component and gave an example of the business rule which is transformed from its visual notation to CGIF, from CGIF to XML, from XML to MS SQL Server trigger. The advantages of our approach are several:

Business rules modeling language – conceptual graphs – is flexible and usable at different levels of IS development activities;

Business rules repository in XML format offers extension points, rules exchange and sharing possibilities;

The wide spread technology of active databases is used for rules enforcement instead of dedicated technologies and tools that easing incorporation of the approach into the real business applications.

The next steps would include the case study using the presented approach where the set of interrelated business rules and resulting triggers are necessary for the selected business domain in order to find out the premises and constraints that apply depending on the number of the business rules and the type of their interrelations.

References

[1] ACT-NET Consortium. The Active Database Management Systems Manifesto: A Rulebase of ADBMS Features. *ACM Sigmod Record*, *Vol*.25(30), 1996, 40-49.

[2] **G. Booch, J. Rumbaugh, I. Jacobson.** The Unified Modeling Language User Guide. *Addison-Wesley,* 2000.

[3] Conceptual Graphs Standard. Document type: International standard (Draft). *Document stage:* (20) Preparation, reference number of working document: ISO/JTC1/SC 32/WG2 N 000. *http://www.jfsowa.com/cg/cgstand.htm,* 2001.

[4] FairIsaac BlazeAdvisor: How it works? *http://www.fairisaac.com/NR/rdonlyres/C3817720-3C36-4B43-9F65-3300B0B9AA29/0/advisor_how.pdf,* 2003.

[5] Business Rules Solutions homepage. *http://www.brsolutions.com, accessed* 2004 05 31.

[6] Infrex. Product overview. *http://www.tcs.com/0_products/infrex/index.htm,* 2002.

[7] **T. Morgan.** Business Rules and Information Systems: Aligning IT with Business Goals. *Addison-Wesley,* 2002.

[8] **P. Dorsey.** Business Rules Analysis in the Real World. *Electronic Proceedings of Oracle Development Tools User Group ODTUG* 2003, *http://www.odtug.com/2003_papers.htm* , 2003.

[9] **R.G. Ross.** Principles of the Business Rule Approach. *Addison-Wesley*, 2003.

[10] **J.F. Sowa.** Knowledge Representation: Logical, Philosophical, and Computational Foundations. *Brooks/ Cole, Pasific Grove et al*., 2000.

[11] **T. Moriarty.** Business-Rule Stuff or Marketing Fluff? *Intelligent Enterprise, Vol*.3, *No*.3. *http://www.intelligententerprise.com/000209/metaprise.jhtml, February* 9, 2000.

[12] **I. Valatkaite, O.Vasilecas.** On Business Rules Approach to the Information Systems Development. *In: H. Linger at al (Eds.). Proc. of Twelfth international conference on Information Systems Development: Constructing the Infrastructure for the Knowledge Economy. Melbourne, Australia, August* 25-27, 2003, *to be published by Kluwer Academic/Plenum Publishers*, 2004.

[13] **I. Valatkaite, O.Vasilecas.** A Conceptual Graphs Approach for Business Rules Modeling. *Advance in Databases and Information Systems, Lecture Notes in Computer Science,* 2798*, Springer-Verlag*, 2003, 178-189.

[14] **I. Valatkaite, O.Vasilecas.** Application Domain Knowledge Modeling Using Conceptual Graphs. *In: Kirikova, M. at al (eds.): Information Systems Development: Advances in Methodologies, Components and Management*. *Kluwer Academic/Plenum Publishers*, 2002, 193-202.

[15] **I. Valatkaite, O.Vasilecas.** Deriving active database triggers from business rules model with conceptual graphs. *Lietuvos matematikos rinkinys, Vilnius, MII, t.* 42, *special issue "Proceedings of the XLIII conference of Lithuanian Mathematical Society"*, 2002, 289-293 *(in Lithuanian).*

[16] Yasu Technologies. QuickRules Discovery Guide. *http://www.yasutech.com/products/quickrules/datasheet.pdf,* 2003.

[17] **H. Delugach.** CharGer Manual. *http://www.cs.uah.edu/~delugach/CharGer/,* 2003.

[18] The Rule Markup Initiative. *http://www.ruleml.org*, accessed *on* 2004 05 31.