Liet. matem. rink, **45**, *spec. nr.*, 2005, 167–173

Events modelling in the process of business rules based information systems development

Olegas VASILECAS, Diana BŪGAITĖ (VGTU)
e-mail: olegas@fm.vtu.lt, diana@isl.vtu.lt

1. Introduction

Information systems (IS) are central for any business because they capture and store the information required to support business processes and ensure the execution of these business processes. The database management system (DBMS) is a vital part of the IS in an organization.

Traditional DBMS are passive. They can only execute a query or a transaction when a users or an application explicitly requests it.

Active DBMS (ADBMS) extend traditional DBMS to enhance their functionality so that they can automatically react to events in a timely and efficient manner.

The ADBMS can perform operations *automatically*, in response to situations that have occurred *inside* or *outside* the database (DB). Those systems are much more complex in every dimension (human, organizational and technical) than the passive ones. It is puzzle to manage the components of those systems and these systems as a whole.

ADBMS are centred on the notions of ECA (*event-condition-action*) rules. When the event of a related rule has been *detected* the system *notifies* the rule. All rules that respond to the event are *triggered* or *fired* and must be *executed*.

It is necessary to determine and elicit rules from the application domain and develop ECA rules to implement them into ADBMS. An event is an important component of ECA rule, since it enables to automate triggering of correspondent rule. The objective of this paper is to investigate how events, as part of ECA rules, are modelling in the process of business rules (BR) based IS development.

2. Related work

The development of an enterprise system requires that systems operating at all enterprise levels (business, information, and software) shared a common conceptualisation first of all. Business system (BS) of an enterprise to the large extent depends on the supporting IS and software system (SS). Therefore the change in the BS results in the changes of the IS and SS (Caplinskas, 2002).

BR are important part of business and exist at the BS level. Any changes of BR from this level must be propagated to lower level systems of enterprise system.

Definition of a BR depends on the context in which it is used. From the BS perspective, a BR is a statement that defines or constrains some aspects of a particular business. (Zachman, 1992; RMC, 2003). These BR are actually derived from business policies. They are created to constrain the actions in the enterprise. (Hay, 1999). At the BS level, BR are expressed in a declarative manner (Perrin, 2004; Hay, 2003). For example: *A customer could not buy more than credit limit permits* (Hay, 1999).

From the perspective of IS, a BR is a statement, which defines the major rules of information processing using a rule-based language (Lebedys, 2004). By (Hay, 1999) these BR constrain data. They do not directly control processes. Processes may be used to implement rules. Expressions of information processing rules are very precise. Terms of the data model are used to define them (Hay, 2003). For example: *"Total Value" of an ORDER could not be greater than the "Credit Limit" of a CUSTOMER* (Hay, 1999).

At the execution level (or SS level), rules are statements that are transferred to the executable rules, like ADBMS triggers.

BR, which are directly relatedto the reactive behaviour of ADBMS, fall under ECA paradigm (when *event* occurs, if *condition* is true, then *action*). They are called dynamic assertions. (Valatkaite, 2004)

As was mentioned above, an event is an important component of ECA rule as a condition and an action. The objective of this paper is to investigate how events are modelling in the process of BR based IS development.

Almost all BR, which belong to dynamic assertions, have explicit or not explicit condition and action parts. The missing condition can always be substituted with a default condition state as *TRUE*. Dynamic assertions may have no explicit action since they can state what kind of transition from one data state to another is not admissible.

But the majority of BR have not explicitly or have not absolutely defined the *event* part. Therefore, it isn't obvious, when a BR and the consequently information processing and the executable rules have been triggered.

There are the following ways to trigger rules:

- Trigger all rules every time, when any related event occurs.
- Trigger rules by user, when he/she decided it are necessary.
- Define events, which triggers rules.

The third one was chosen for rules triggering, because as was mentioned above, events allows the specification and implementation of reactive behaviour of a system. Events specification and linking them to exact rules enable automatically react to the defined events and perform defined operations, e.g., to automate rules triggering. System is not overloading by rules executing every time then any event occurs.

Therefore the events and their modelling possibilities are analysed below.

2.1. Events and their modelling

In IS development, the concept of event means occurrence of happening of interest in the application domain (Adaikkalavan, 2003; Michiels, 2003; Cilia, 2002).

Events can be either primitive (e.g., depositing cash in bank) or composite (e.g., depositing cash in bank, followed by with-drawl of cash from bank). Primitive events occur at a point in time (i.e., time of depositing). They are a finite set of events that are pre-defined in the domain of interest. Composite events occur over an interval (i.e., interval starts at the time cash is deposited and ends when cash is withdrawn). They are consist of several primitive events. (Adaikkalavan, 2003).

Events reflect how information and objects come into existence, how they are modified, and how they disappear from our universe of discourse (Michiels, 2003).

An event driven system is a system in which actions result from business events. BR determines what event under witch conditions leads to what action. Any action may constitute a new business event. (Johnston, 2004; OMG, 2005)

Software events are messages in the IS that describe business events. Software events are generated by an application program or some other software. (Weigand, 2005). Event allows an application to signal that something of importance has happened (Lockhart, 2005).

The main concepts in event driven business models are the business entity, business event, business process, business activity and BR. So the basic building blocks are the business process and the business entity. The two are 'wired together' by a flow of actions from process to entity, and by a flow of events from entity to process. In a component framework, therefore, business processes have event inflow and action outflow, and entities have action inflow and event outflow. (OMG, 2005)

In (Cilia, 2005) the shared terms defined in common vocabularies, or ontologies, are used as the basis for the correct interpretation of events coming from different sources. An event is represented as triplet of the form $<\psi C, \psi v, \psi S>$ with C referring to a concept from the underlying ontology or vocabulary, v standing for the actual data value, and S representing the *semantic context* of v. This semantic context consists of a variable set that explicitly describe implicit modelling assumptions. For example, the event `PostNewPackageOffer` can be described by attributes `FromDate`, `UntilDate`, `Accommodation`, `PackagePrice`, e.g., (Cilia, 2005; Siorpaes, 2004).

In (Cilia, 2002) events are classified as follows. *DB events* refer to the data modification (like SQL operations insert, delete and update) and data retrieval (like selection in a relational DB, the fetch of an object in an OODB) in the DB. *Transaction events* refer to the different stages of transaction execution, e.g., begin transaction, commit, rollback, etc. *Temporal events* refer to the time. *Abstract events* or application-defined events are signalled explicitly by the application, e.g., AuctionCancelled.

Entity life cycle is widely used in a variety of methodologies to represent changes that happen over time (most of the other techniques represent static views of a system). The objective of entity life cycle analysis is to identify the various possible states that entity can legitimately be in. An event is always a starting point, which sets the entity into its initial state. (Avison, 2003)

3. An approach on events modelling

The structure of an enterprise system from (Caplinskas, 2002) was extended with rules and events facilities to determine the propagation of BR and events to lower levels of enterprise system and to show the formation of rules and events in different levels of abstraction (Fig. 1).

BR and business events are important part of business and exist at the BS level. Any changes of BR and business event from this level must be propagated to lower level systems of enterprise system.

Executable rules implement information processing rules, information processing rules implement BR. Software events implement events in IS, events in IS implement business events. Therefore, the BR can be mapped into information processing rules, while the information processing rules can be mapped or transformed into executable rules (like SQL triggers in ADBMS). The business event can be mapped into events in IS, while the events in IS can be mapped or transformed into software events.

The structure of an enterprise system in Fig. 1 is going to be used in further research of events modelling in the process of BR based IS development.

In the related work analysing events authors not explicitly define the level of abstraction they talk about. Usually they talk about one level of abstraction.

According to the levels of abstraction all events can be classified into business events, events in IS and software events. The main objective of this paper is that it is necessary to distinct business events, events in IS and software events to ensure consistent and complete modelling and implementation of these events. It is necessary to start from analysis and modelling of events in the BS level and consequently follow to the analysis and modelling of events in IS and then of the software event.

For this according to the related works business events, events in IS and software events can be defined in the following way:

A business event – is a significant occurrence of a happening of interest in the application domain. An example of a business event can be customer order, the arrival of a shipment at a loading dock, or a truck breakdown. A business event activates a business process. In particular situations the ending of a business process can cause a new business event.

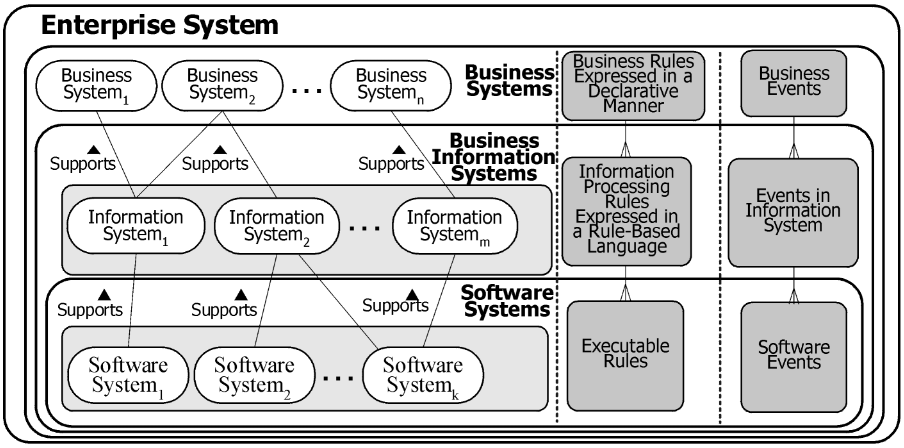

Fig. 1. The structure of an enterprise system.

An event in IS is generated by an IS. It is the implementation of the business event. One business event can be implemented by one or several events in IS. An example of an event in IS can be *a method execution* by an object.

A software event is generated by a SS. An example of a software event can be SELECT, CREATE, UPDATE, DELETE and other.

However, not all information processing rules and events in IS are come from a BS. Some of them came from an IS.

According to the above made definitions, the modelling abstraction levels from (Caplinskas, 2002) were extended with rules and events modelling facilities and their propagation into lower levels of enterprise system (Fig. 2).

A software event activates a particular action processing. In some cases it can trigger particular rules (ECA rules). Rule execution incorporates condition evaluation and action execution. First, the condition is evaluated (checked) and if it satisfies, the action or some actions are executed. These actions can cause another software event and so on. The possible event-driven process chain in the SS is shown in Fig. 3 (Nüttgens, 1997; Cilia, 2002).

In this work we make an assumption that digital process consist of one or several actions in SS.

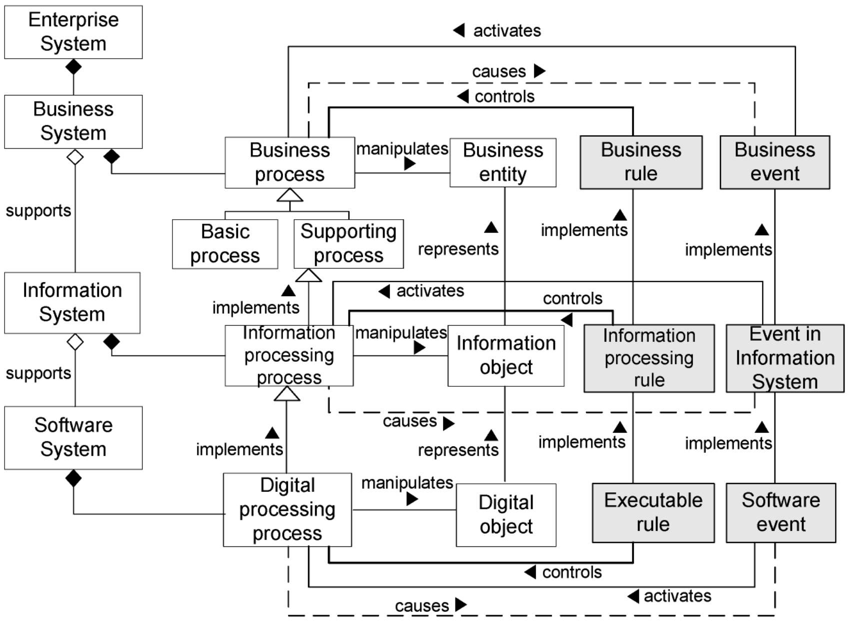

Fig. 2. Events and rules in BS, IS and SS.

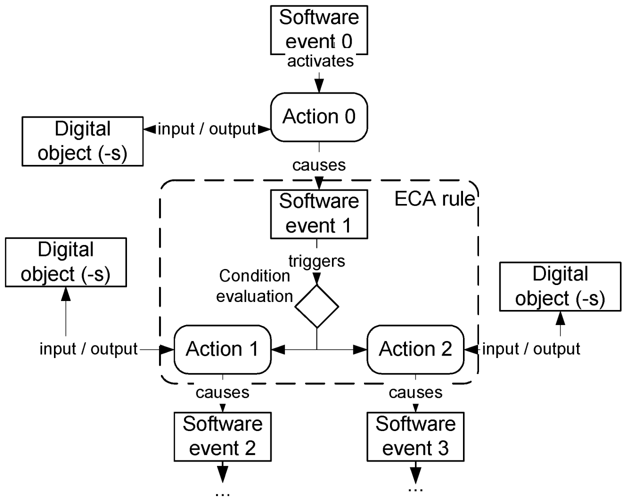

Fig. 3. Event-driven process chain in the SS.

4. Conclusion

The analysis of the related works on events modelling in the process of business rules based information systems development shows that it is necessary to distinct business events, events in information systems and software events to ensure consistent and complete modelling and implementation of these events in information systems.

The business events can be mapped into events in information system, while the events in information system can be mapped or transformed into software events.

According to the made definitions of business event, event in information system and software event the modelling abstraction levels (business system, information system and software system) were extended with rules and event modelling facilities and their propagation into lower levels of enterprise system.

References

1. R. Adaikkalavan, S. Chakravarthy, SnoopIB: interval-based event specification and detection for active databases, in: L. Kalinichenko, R. Manthey, B. Thalheim, U. Wloka (Eds.), *Proc. of Seventh East-European Conference on Advance in Databases and Information Systems (ADBIS)*, Dresden, Germany. *LNCS*, **2798**, Springer-Verlag (2003), pp. 190–204.
2. D. Avison, G. Fitzgerald, *Information Systems Development. Methodologies, Techniques and Tools*. Third Edition. Mc Graw Hill (2003).
3. A. Caplinskas, A. Lupeikiene, O. Vasilecas, Shared conceptualisation of business systems, information systems and supporting software, D*atabases and Information Systems*, I, Kluwer Academic Publishers (2002), pp. 109–120.
4. M. Cilia, *An Active Functionality Service for Open Distributed Heterogeneous Environments*, PhD Thesis, Technische Universit¨at Darmstadt genehmigte (2002).
`http://www.dvs1.informatik.tu-darmstadt.de/publications/pdf/active-functionality`

5. M. Cilia, C. Bornhovd, A.P. Buchmann, Event handling for the universal enterprise (to appear), *Information Technology and Management (ITM)*, Special Issue on Universal Global Integration (2005). `http://www.dvs1.informatik.tu-darmstadt.de/staff/bornhoevd/ITM-journal.pdf`
6. D.C. Hay, *Requirement Analysis. From Business Views to Architecture*, Prentice Hall PTR, New Jersey (2003).
7. D.C. Hay, *Managing Business by the Rules* (1999). http://essentialstrategies.com/documents/bruleseco99.pdf.
8. S. Johnston, *Rational UML Profile for Business Modelling*, IBM (2004). `http://www-128.ibm.com/developerworks/rational/library/5167.html#author1`
9. E. Lebedys, O. Vasilecas, Analysis of business rules modelling languages, in: *Proc. of the "Information Technologies 2004"*, Kaunas, Lithuania, Technologija (2004), pp. 487–494.
10. J. Lockhart, *The Big Event: Oracle Workflow Business Event System*, IMPAC (2005). `http://bcoaug.oaug.org/downloads/Workflow%20BES.ppt`
11. C. Michiels, M. Snoeck, W. Lemahieu, F. Goethals, G. Dedene, A layered architecture sustaining model driven and event driven software development, in: M. Broy, A.V. Zamulin (Eds.), *Proc. of the Perspectives of System Informatics*, Novosibirsk, Akademgorodok, Russia (2203), *LNCS*, **2890**, pp. 58–65.
12. M. Nüttgens, T. Feld, V. Zimmermann, Business process modeling with EPC and UML transformation or integration? in: M. Schader, A. Korthaus (Eds.), *Proc. of the Unified Modeling Language – Technical Aspects and Applications*, Mannheim, Heidelberg (1997), pp. 250–261.
13. OMG, EDOC Vision (Extracted from the EDOC specification). Object Management Group (OMG), (2005). `http://www.enterprise-component.com/docs/EDOC%20Vision.pdf`
14. O. Perrin, C. Gobart, An approach to implement contracts as trusted intermediaries, in: *Proceedings of the First International Workshop on Electronic Contracting (WEC'04)* (2004), pp. 71–78.
15. RMC, Why Use Rules? Role Machines Corporation (RMC) (2003). `http://www.rulemachines.com/VRS/whyrules.htm`
16. K. Siorpaes, K. Prantner, D. Bachlechner, Project: e-tourism-v8. Class Event, (2004). `http://e-tourism.deri.at/ont/Event.html`
17. I. Valatkaite, O. Vasilecas, On business rules approach to the information systems development, in: H. Linger *et al.* (Eds.), *Proc. of Twelfth International Conference on Information Systems Development, Constructing the Infrastructure for the Knowledge Economy*, Melbourne, Australia, Kluwer Academic/Plenum Publishers (2004), pp. 199–208.
18. H. Weigand, A. de Moor, Linking event-driven and communication-oriented business modeling, in: *The Language Action Perspective on Communication Modelling*, Kiruna, Sweden (2005), pp. 107–118.
19. J.A. Zachman, J.F. Sowa, Extending and formalizing the framework for information systems architecture, *IBM Systems Journal*, **31** (3), 590–616, IBM Publication (1992).

REZIUMĖ

O. Vasilecas, D. Būgaitė. Įvykių modeliavimas verslo taisyklėmis grindžamų informacinių sistemų kūrimo procese

Straipsnyje yra aptariamas įvykių modeliavimas įvairiuose įmonės modeliavimo lygmenyse (verslo, informacinės sistemos ir programų sistemos). Darbe yra pateikiamos ir aptariamos verslo taisyklės ir įvykio sampratos priklausomai nuo įmonės modeliavimo lygmens.

Priklausomai nuo pateiktų sampratų ir jų tarpusavio sąryšių, įmonės modeliavimo lygmenys yra praplečiami taisyklėmis ir įvykiais.