BUSINESS RULES ELICITATION IN THE PROCESS OF ENTERPRISE INFORMATION SYSTEM DEVELOPMENT

Olegas Vasilecas, Diana Bugaite
Vilnius Gediminas Techical University, Saulėtekio al.11, Vilnius – 40, LT-10223 Lithuania

Keywords: Business rule, event, ECA rule, ontology, business system, information system, software system.

Abstract: The paper considers the problems of events modelling in the process of developing business rule-based information systems and determines their significance. The concepts of a rule and an event are defined at different levels of abstraction (business system, information system and software system). According to the suggested definitions of business event, event in the information system and software event, the abstraction levels of modelling are extended by the rule and event modelling facilities and their propagation into the lower levels of an enterprise system. Since ontology represents the real-world domain knowledge and events as well as business rules, making a specific part of all domain knowledge, it is suggested to use ontology as an extra source to improve elicitation of business rules and events.

1 INTRODUCTION

Enterprise information systems (IS) are central for any enterprise because they capture and store the information required to support business processes and ensure the execution of these business processes at an enterprise. Information processing rules are used in IS to process the required information correctly. These information processing rules are derived from business rules (BR).

In practice, information processing rules are implemented by active DBMS SQL triggers.

Information processing rules should be expressed as ECA (*event-condition-action*) rules to be implemented by active DBMS SQL triggers. Therefore it is necessary to determine and elicit rules from the application domain and develop ECA rules.

An event is an important component of an ECA rule, since event specification and linking them to corresponding rules enable us to automate rules triggering. Moreover, system is not overloaded during the rule execution every time when an event occurs.

IS, which can perform operations automatically, e.g., respond to events inside or outside a system, are called either a reactive IS or active IS.

The objective of this paper is to investigate how events, making an important part of ECA rules, are modelled in the process of BR-based IS development.

Since ontology represents the real-world domain knowledge, events and rules, making a specific part of all domain knowledge, it can be used to form a set of ECA rules.

2 RELATED WORK

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

BRs make an important part of business and exist at the BS level. Any changes in BRs at this level should be reflected at the lower level systems of an enterprise.

A 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. These BRs are actually derived from business policies. They are created to constrain the actions at the enterprise. At the BS level, BRs are expressed in a declarative manner. (Bugaite, 2005). For example: *A customer could not buy more than credit limit permits*.

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). 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 active DBMS triggers.

In this paper, the emphasis is placed on the BRs, which are directly related to the reactive behaviour of active ISs. These BRs fall under the ECA paradigm (when *event* occurs, if *condition* is true, then *action*). Note, that the BRs of only this category are discussed.

However, in the application domain or ontology, to which the BRs belong (see Figure 1), they are not always expressed in terms of ECA rules.

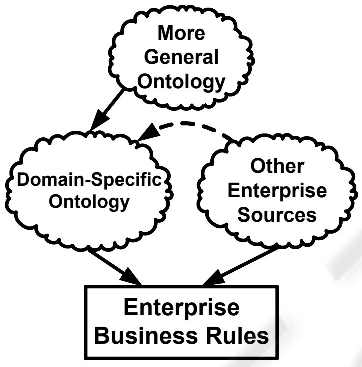

Figure 1: Sources of Business Rules.

Some of these BRs have explicit or implicit condition and action parts. The missing condition can always be substituted with a default condition state as *TRUE*. Some BR may have no explicit action since they can state what kind of transition from one data state to another is not admissible. (Valatkaite, 2004)

But the majority of these BRs do not define explicitly or implicitly the *event*. Therefore, it is not clear, when a BR, the consequent information processing and executable rules have been triggered.

There are the following ways to trigger rules:

▪ A system triggers all rules every time, when any related event occurs.

▪ A user triggers rules, when he/she decides that it is necessary.

▪ A user defines the events, which trigger rules.

In this paper the third way was used for rule triggering, because, as mentioned above, events allow the specification and implementation of the reactive behaviour of a system. The specification of the events and their linking to actual rules enabling the system automatically react to the defined events and perform the defined operations, e.g. automatic rule triggering. System is not overloaded during the rule execution every time when an event occurs.

A method of the BRs elicitation from the application domain problems is described in (Bugaite, 2005).

The one possible solution proposed to solve the defined problems was using of the domain ontology.

Since ontology represents the real-world domain knowledge, events and BRs, making a specific part of all domain knowledge, it can be used to form a set of BRs (Bugaite, 2005).

The analysis of the ontology concept was made in (Bugaite, 2005). Therefore, only a definition of the ontology suggested by these authors is presented in this paper.

Ontology is a specification of a conceptualisation. Ontology defines the basic terms and their relationships, comprising the vocabulary of an application domain and the axioms.

Ontology axioms (and ontology as a whole) represented in a formal way can be used to elicit BRs. The ontology axioms can be transformed into BRs, while the BRs can be transformed into information processing rules and latter can be transformed into executable rules. (Bugaite, 2005)

The analysis of ontology axioms shows, that axioms have clearly defined action(s) and, sometimes, condition(s). Event is not defined because axioms define the state in which an application domain should be. However, no information is provided about what should be done to implement a desirable state. (Bugaite, 2005). Therefore, it is necessary to investigate events and ways of their modelling.

2.1 Events and Their Modelling in IS Development

In IS development, the concept of event means the 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 withdrawal 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 consist of several primitive events. (Adaikkalavan, 2003).

Events reflect how information and objects come into existence, how information and objects 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. BRs determine what these events are and under what conditions they can lead to some particular actions. Any action may constitute a new business event. (Johnston, 2004; OMG, 2005).

Software events are messages in the IS that describe a business event. 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 < *C; v;* 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 the attributes FromDate, UntilDate, Accommodation, PackagePrice, e.g. (Cilia, 2005; Siorpaes, 2004).

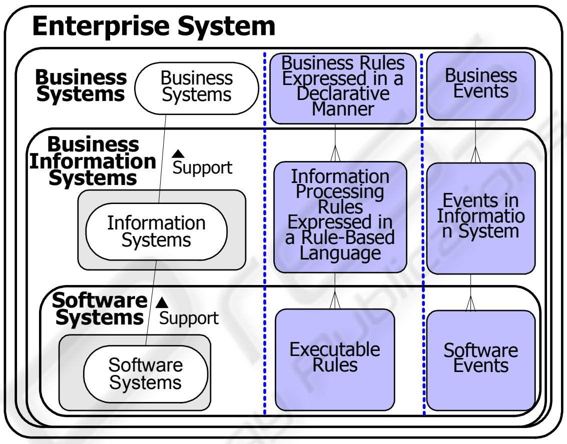
Enterprise System
Business Systems
Business Systems
Business Rules Expressed in a Declarative Manner
Business Events
Business Information Systems
Support
Information Systems
Information Processing Rules Expressed in a Rule-Based Language
Events in Information System
Software Systems
Support
Software Systems
Executable Rules
Software Events

Figure 2: The structure of an enterprise system.

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 in the DB (like selection in a relational DB, the fetch of an object in an OODB). *Transaction events* refer to the different stages of transaction execution, e.g. begin transaction,

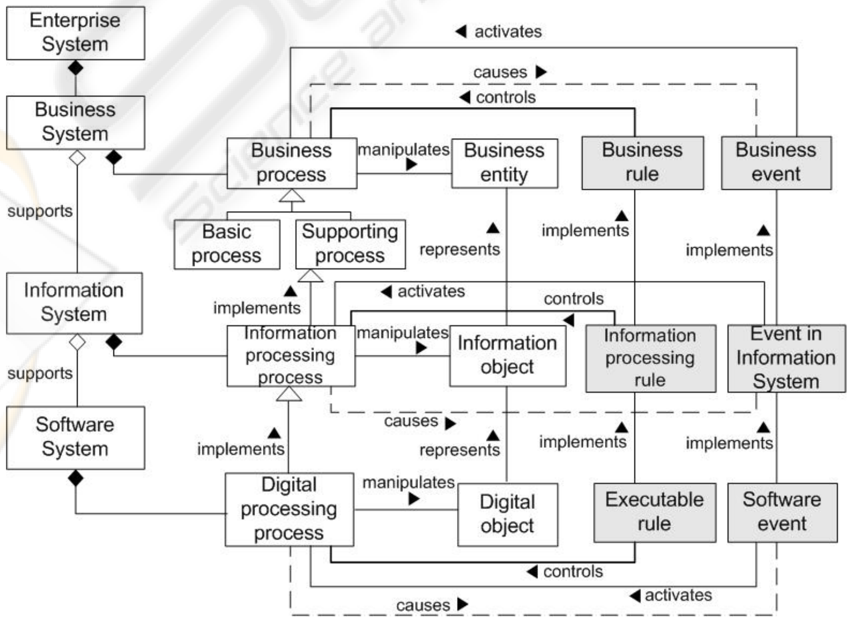

Figure 3: Events and rules in business, information and software systems.

commit, rollback, etc. *Temporal events* refer to time. *Abstract events* or application-defined events are signalled explicitly by the application, e.g. AuctionCancelled.

According to (Mahesh, 1996; Humphreys, 1997) knowledge about events is represented by ontology as well. Events are modelled by the terms defined in the domain ontology. Sometimes, special event ontologies are created for events in a certain application domain modelling. But these so-called event ontologies do not exist alone. They usually are related to term (or so-called context) ontologies of the same domain. (Yu, 2001). For the sake of simplicity, the first case is used in this paper.

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 TO EVENT MODELLING IN IS DEVELOPMENT

The structure of an enterprise system described by (Caplinskas, 2002) was extended by rule and event facilities to determine the propagation of BRs and events to lower levels of an enterprise system and to show the formation of rules and events at different levels of abstraction (Figure 2).

BRs and business events make an important part of business and exist at the BS level. Any changes in the BRs and business events at this level should be reflected at the lower level of an enterprise system.

Executable rules implement information processing rules, while information processing rules implement the BRs. Software events implement the events in IS, whereas the events in IS implement business events. Therefore, the BRs 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 Figure 2 is going to be used in further research of events modelling in the process of developing BR-based IS.

In the analysis of the related work the authors do not explicitly define a particular level of abstraction because only one level of abstraction is considered.

According to the levels of abstraction, all events can be classified into business events, events in IS and software events. It is necessary to differentiate between business events, events in IS and software events to ensure consistent and complete modelling and implementation of these events. First, events at the BS level have to be analysed and modelled, second – events in IS and finally – software events should be handled.

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

A business event is a significant occurrence or a happening of interest in the application domain. An example of a business event can be a 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.

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 others.

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

According to the above definitions, the modelling abstraction levels described by (Caplinskas, 2002) were extended by rule and event modelling facilities and their propagation into lower levels of enterprise system (Figure 3).

A software event activates the 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 is true, the action or some actions are executed. These actions can cause another software event, and so on. A possible event-driven process chain in the SS is shown in Fig. 4 (Nüttgens, 1997; Cilia, 2002).

In this work, an assumption is made that a digital process consists of one or several actions in SS.

Ontology of an enterprise (or, in general, a domain ontology) is above the enterprise system.

Some propositions presented above will be repeated below to draw some inferences.

Ontology axioms can be transformed into the business rules, while the BR can be transformed into information processing rules and the information processing rules can be transformed into executable rules.

Events in domain ontology are modelled by the terms defined in this domain ontology. Ontology events are more general or the same as business events, for example, a customer order, a particular meeting…

3.1 A Case Study of Ontology Axioms and Events Transformation into the ECA Rules

The ontology for a particular business enterprise was created using Protégé-2000 ontology development tool to support the statement of the authors that ontology axioms and events can be transformed into BRs. Protégé-2000 was chosen to develop the ontology because it allows the open source software to be installed locally. A free version of the software provides all features and capabilities required for the present research as well as being user-friendly. It also maintains multiple inheritances, provides exhaustive decomposition, disjoint decomposition and constraints writing as well as being Java-based (Jakkilinki, 2005).

The axioms are implemented in Protégé-2000 ontology by the Protégé Axiom Language (PAL) constraints. PAL is a superset of the first-order logic, which is used for writing strong logical constraints (Crubézy, 2002).

The schema of PAL constraints (and ontology as a whole) was analysed to enable their transformation into BRs.

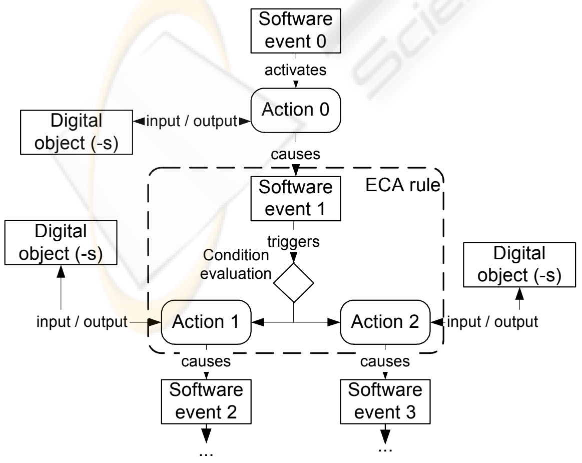

Figure 4: Event-driven process chain in the SS.

The core of the PAL constraint is the *PAL-statement*, which can be mapped to the BR and consequently to the ECA rule. The PAL-statement *has* clearly defined *action* and, sometimes, *condition*. The *event* is not defined because the user triggers constraints represented by PAL manually, when required. All constraints written by PAL define the state in which the domain should be. However, no information is provided about what should be done to implement a desirable state.

An example of the PAL constraint is as follows:

```
(defrange ?contract_product :FRAME
       Contract_Product)
(forall ?contract_product
(and (=> (and (> (quantity
            ?contract_product) 10.0)
(< (quantity ?contract_product) 19.0))
(= (discount ?contract_product) 0.03))
      (=> (and (> (quantity
            ?contract_product) 20.0)
(< (quantity ?contract_product) 49.0))
(= (discount ?contract_product) 0.05))…))
```

Some events of interest were defined in terms of ontology.

There is no distinct relationship between ontology axioms and the defined events. Therefore, it is necessary to extend ontology by adding a class, describing the relationship between particular events and axioms, to enable an automatic transformation of ontology axioms and events into BRs.

4 CONCLUSION

The analysis of the related works on information systems development using the domain ontology shows that the business rules and events are part of knowledge represented by the ontology. Ontology axioms and events, describing terms of the same ontology, can be transformed into the business rules, while the business rules can be transformed into information processing rules and, finally, the information processing rules can be transformed into executable rules. Ontology events can be transformed into business events, which can be mapped into events in the information system, while the events in the information system can be mapped or transformed into software events.

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

The example provided shows that the suggested method could be used to elicit business rules ontology. For this transformation, a suitable tool was needed and Protégé-2000 was chosen.

The analysis has shown that this tool can be used for the purposes pursued in this study. However, it is not provided with a suitable plug-in for PAL constraints and events transformation into ECA rules.

REFERENCES

Adaikkalavan, R., S. Chakravarthy (2003). SnoopIB: Interval-Based Event Specification and Detection for Active Databases. LNCS 2798, Springer-Verlag, 2003, p.190-204.

Avison, D., G. Fitzgerald (2003). *Information Systems Development. Methodologies, Techniques and Tools*. Third Edition. Mc Graw Hill. 2003. p. 592.

Bugaite, D., O. Vasilecas (2005). Ontology-Based Elicitation of Business Rules (to appear). In A.G.Nilsson, R.Gustas, W.Wojtkowski, W.G.Wojtkowski, S.Wrycza, J.Zupancic (Eds.), *Proc. of ISD'2005*, Sweden, Springer-Verlag, 2006, p. 795-806.

Caplinskas, A., A. Lupeikiene, O. Vasilecas (2002). Shared Conceptualisation of Business Systems, Information Systems and Supporting Software. *Databases and Information Systems II*, Kluwer Academic Publishers, 2002, p. 109-120.

Cilia, M. (2002). *An Active Functionality Service for Open Distributed Heterogeneous Environments*. PhD Thesis, Technische Universit¨at Darmstadt genehmigte, 2002. (October, 2005). http://www.dvs1.informatik.tu-darmstadt.de/publications/pdf/active-functionality.

Cilia, M., C. Bornhovd, A. P. Buchmann (2005). Event Handling for the Universal Enterprise (to appear). *Information Technology and Management (ITM)*, Special Issue on Universal Global Integration, 2005. (September, 2005). http://www.dvs1.informatik.tu-darmstadt.de/staff/bornhoevd/ITM-journal.pdf.

Crubézy, M. (2002). The Protégé Axiom Language and Toolset (PAL). 2002. (February, 2005). http://protege.stanford.edu/plugins/paltabs/pal-documentation/index.html.

Humphreys, K., R. Gaizauskas, S. Azzam (1997). Event Coreference for Information Extraction. *The Workshop on Operational Factors in Practical, Robust Anaphora Resolution for Unrestricted Texts*, 35th Annual Meeting of the Association for Computational Linguistics, Madrid, 1997, p. 75–81.

Yu, J., J. Hunter, E. Reiter, S. Sripada (2001). An Approach to Generating Summaries of Time Series Data in the Gas Turbine Domain. In *Proc. of ICII2001* (Beijing), IEEE Press, p. 44-51.

Jakkilinki, R., N. Sharda, M. Georgievski (2005). Developing an Ontology for Teaching Multimedia Design and Planning. (September, 2005). http://sci.vu.edu.au/~nalin/MUDPYOntologyPreprintV2.pdf.

Johnston, S. (2004). Rational UML Profile for business modelling. IBM. 2004. (August, 2005). http://www-128.ibm.com/developerworks/rational/library/5167.html#author1.

Lebedys, E., O. Vasilecas (2004). Analysis of business rules modelling languages. In *Proc. of the IT'2004*, Lithuania, Technologija, 2004, p. 487-494.

Lockhart, J. (2005). The Big Event: Oracle Workflow Business Event System. IMPAC. (February, 2005). http://bcoaug.oaug.org/downloads/Workflow%20BES.ppt.

Mahesh, K. (1996). Ontology Development for Machine Translation: Ideology and Methodology. *Technical Report MCCS 96-292*. Computing Research Laboratory, New Mexico State University. 1996. (September, 2005). http://www.fdi.ucm.es/profesor/vaquero/DOCT/MikroKosmosOnto(MCCS-96-292).pdf.

Michiels, C., M. Snoeck, W.Lemahieu, F. Goethals, G. Dedene (2003). A Layered Architecture Sustaining Model Driven and Event Driven Software Development. LNCS 2890, p. 58-65.

Nüttgens, M., T. Feld, V. Zimmermann (1997). Business Process Modeling with EPC and UML Transformation or Integration? In M.Schader, A. Korthaus (Eds.), *Proc. of the UML'97 Workshop "Unified Modeling Language - Technical Aspects and Applications",* Germany, Physica-Verlag, 1997, p. 250-261.

OMG (2005). EDOC Vision (Extracted from the EDOC specification). Object Management Group (OMG). 2005. (September, 2005). http://www.enterprise-component.com/docs/EDOC%20Vision.pdf.

Siorpaes, K., K. Prantner, D. Bachlechner (2004). Project: e-tourism-v8. Class Event. 2004. (September, 2005). http://e-tourism.deri.at/ont/Event.html.

Valatkaite, I., O. Vasilecas (2004). On Business Rules Approach to the Information Systems Development. In H. Linger at al (Eds.), *Proc. of ISD'2003.* Australia, Kluwer Academic/Plenum Publishers, 2004, p.199-208.

Weigand, H., A. de Moor (2005). Linking event-driven and communication-oriented business modeling. *The Language Action Perspective on Communication Modelling,*Sweden, 2005, p. 107-118.