INFORMATICA, 1999, Vol. 10, No. 2, 189–202

Business Object Modeling Framework for Distributed Enterprise

Lina NEMURAITĖ

Kaunas University of Technology
Studentų 50–108, 3028 Kaunas, Lithuania
e-mail: isd@if.ktu.lt

Received: January 1999

Abstract. In this paper the framework for business object modeling with focus on distributed enterprise is proposed. It is based on Business Object Architecture, UML and Catalysis method. Business Object Architecture is methodology bringing business semantics to component-based development – the next generation of object-oriented methodology. Basic modeling concepts are business objects, business processes and business rules. Process of business process modeling with Business Objects is described and generic modeling patterns are presented. The framework is illustrated via work effort process modeling.

Key words: object-oriented, business object architecture, business process, business rule.

1. Introduction

The purpose of this work is to develop modeling framework based on well-founded methodology for information system development seeking for qualities of integrity, interoperability and evolution. The target audiences are business modelers and information system developers confronting problems with business process and information system reengineering.

Choosing of methodology for information system development today deals with many questions, which were not urgent or had been ascribed to different areas earlier. Choosing methodology you should think about dealing with Business Process and Information System Reengineering, Enterprise Strategic Information Warehouse and Decision Support, Intranet and WWW Application Development, Utilization of Legacy Systems, Workflow Management, and perhaps much more. All these areas of activities must be based on unified information model of underlying business processes. Classical object-oriented methods are not sufficient for building adaptive distributed information systems. Methodology chosen as most promising for this purpose consists of several constituents (Fig. 1). The essence of this methodology lies in Business Object Architecture (BOA) – business and information system design artifacts modeling in the form of the system of interoperating business objects (Patel, 1998; BOA, 1997) and this system representation in Unified Modeling Language (UML) (UML, 1997). Catalysis (D'Souza, 1998)

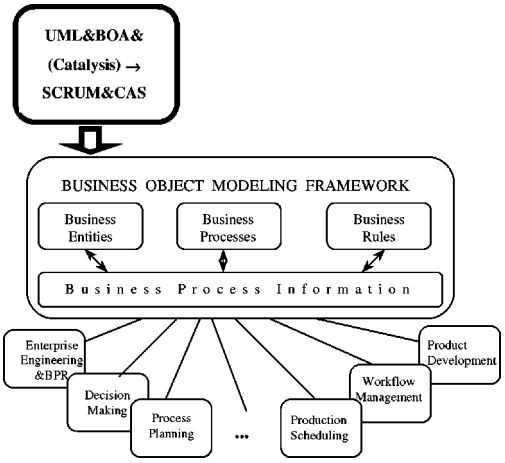

Fig. 1. Business Object Modeling Framework makes a base for shared business process information.

is used as advanced component development method though another methods are available. Rational Unified Process (RUP) represents iterative development life cycle and risk minimization (Kruchten, 1998). SCRUM (Sutherland, 1998) is the method for project management seeking for even more advanced features by employing characteristics of CAS – Complex Adaptive Systems (Patel, 1998) – and effective risk management.

Information System development based on business object architecture is a hard task. This methodology not only brings more semantics to object-oriented business model but is sophisticated as well. It is suited for large companies (development or acquisition of single business object is very expensive). It can be applied for learning and standard model development for different business domains and different business, embodying the best practices and serving as a repository; for individual enterprise business model development; for packaged software, as seldom package distributors may offer configurable business process model serving for explanation of their facilities and adaptation to customer business; etc.

2. Business Object Architecture Explanation

Concept of Business Object Architecture was proposed and elaborated by the workshops of OOPSLA conference and OMG. In business modeling, Business Object describes the real-world business concept being modeled (entity, process, event). In application development, Business Object represents how this business modeling concept is realized in a software design or program code. Interoperable business objects are application components relevant for application developers and users. Many firms of software development are on this way (for example, SAP, Riverton Software, etc.), every of them has its own method for business object modeling. OMG efforts are directed to development of standards giving ability to build and deploy interoperable and reusable software objects

for industry and cross-industry use. Constructs and types that are used to build business object system are described as OMG BOA Metamodel. Functioning of OMG Business Object system is based on CORBA infrastructure tools.

BOA elements are Business Objects and Dependent Types. Business Object Types are Models, Entities and Processes. Dependent Types are, for example, appliances: rules, dependencies, etc. Model is a system of Business Objects, such as Financial, Production Management, etc. Entity represents person, place, thing or concept; Process represents an activity, such as purchasing, approval or production. As business objects encapsulate their internal behavior, integrated behavior of system of business objects is ensured via Event-Rule-Action model. Process Object is similar to any business object with exception that it represents the business activity. Sometimes it is called as reified method. Business process objects may be a part of workflow system. Processes also affect and are affected by rules. Process objects and rules give the possibility for workflow execution: required process objects may be fired by event triggering facility or other object runs. For implementation of business processes, Workflow Management Systems may be used as well.

Business objects may contain methods, rules, states, and Process Objects as ways to describe the actions that it performs. Methods are generally appropriate for the "low-level" implementation, and Process Objects are most-appropriate for domain-level actions (business processes) performed by Business Objects. In modeling business domain, processes are preferable against methods because they have long lifetime and may represent joint actions of multiple users. Also they may have visible states and lifetime, relationships to other objects, invoke and be invoked by methods, etc.

In BOA, the fundamental business object modeling constructs are rules. Using rules one may specify object semantics independent of object implementation. Rules represent things that must be done or validated when any action is taken by or any change is made. Rules are not limited to individual business object, they may specify constraints and actions in context of overall system. They may or may not be implemented as separate components.

There are two extension mechanisms that allow to specify new constructs: business object type parameters and appliances, which allow new constructs to be defined and then "applied" to type definitions.

The particular meaning in business domain modeling stands for the role feature. Business object may have many roles, which may be added or deleted dynamically. People play different roles during their lifetime; in distributed enterprise the same organization may play role of a customer and a supplier, and so on. Object roles allow more stable description of individual business objects and more adaptive object system model: they are able to model real relationships between business objects in multiple applications. The role "inherits" all specifications of its base type to which it is attached. In this view it is similar to concept of subtype but role differs in that it may be changed.

OMG stated requirements concern BOA constructs, their structure, relationships, semantics and specification, but say nothing about process of business object identification and modeling – here any component development method may be used.

3. Business Process Modeling with Business Objects

The process proposed for business object model development is described below. Enterprise information architecture reengineering with business objects requires to construct this architecture from problem domain, which is often represented with convenient data and process models. Here the notion of "construct" was used because it is impossible to simply extract business objects from Entity-Relationship (ER) or classic object-oriented model – you have to make a decision what business object you need. This decision depends on two criteria – low coupling and high cohesion. For entity objects it is done intuitively on expert base. For process objects there are certain guidelines: you have to create process object for every collaboration (Catalysis, 1998). Business object model development consists of several iterative and interoperating phases:

• Object-oriented problem domain model development in UML. In a case of extracting this model from ER model, there are straightforward procedures for converting entities and relationships to relevant object class model (and vice versa). It is recommended to improve input model by using patterns, which reflect features of distributed and adaptive enterprise (e.g. patterns in Figs. 3–5).

• Process model development in a form of UML activity (workflow) diagram with swimlines and object states. This phase is based on event-response analysis. Alternatively, the UML workflow diagram can be obtained from different kinds of workflow diagrams.

• Use-case model development. Use-case model consists of use-case diagrams and use-case specifications. It represents definitions of business processes in terms of goals, responsibilities, preconditions and postconditions. The use-case instances are concrete sequences of events triggering operations or other process instances. These sequences are specified as a set of possible scenarios in use-case specification. Use-case model has to be developed from enterprise workflow model. If you have the workflow model at sufficiently precise level (with conditions and object states), you would lose information if you map workflow model to alone use-case model; you also have to map to interaction model and life cycle model.

• Interaction model development. Interactions in UML are represented by sequence diagrams and collaboration diagrams between objects for every use-case. Use-case diagram does not represent the sequence in which use-cases are used; the user can use the system in several sequences. Some authors (Patel, 1998) propose to create sequence and collaboration diagrams representing interactions among actor instances and use-case instances (also among packages, nodes, subsystems). Sequence and collaboration diagrams between actor instances and use-case instances represent interactions between business processes, they are scenarios between scenarios. These representations are required if you want to understand business processes and to manage workflow. In Catalysis method, the aggregation relationship between use-cases is proposed. The alternative way pursued in current paper is to represent the overall process as high level use-case embracing scenarios between instances of BOA components – business processes. Use-case diagrams can be refined and structured using stereotyped relationships ≪uses≫ and ≪extends≫. If you represent every use-case with a set of collaborations and create a single business process object for every collaboration this activity of your development process is

defined precisely. But you have to decide what your business processes are, and refine them from the top to the lowest level. In this phase Catalysis modeling concepts (type, collaboration and refinement) and principles are useful. In this work all diagrams are represented in UML notation (UML, 1997) and UML extensions proposed in Catalysis method (D'Souza, 1998) had not been used. The reason for this is that OMG has announced also request for proposal on UML extension for business object modeling and work is in progress. Catalysis notation is very rich and precise, but it is universal method directly not intended for business objects.

• Business Object identification and static model development. This phase is done iterating the use-case and interaction model development. Business objects are approximately clear from the real world: order, invoice, agreement are business entity objects, whereas order fulfilling, invoice registration and agreement cancellation are business process objects. But, making static structure diagram you have to relate with order entity additional entities, which are tightly coupled with order and lowly coupled with remaining object types. You should think about interactions in such a way that single business process would be forced to interact with minimal possible number of business objects. Also, you have to refine business objects to required lowest level and combine them into more abstract ones and packages. And you may use patterns, which help you and guide through these iterations.

• Business Object life cycle model development. This kind of model is represented by state and activity diagrams (UML, 1997). For comprehensive business object management recognition and maintenance of all its life cycle phases is necessary. In the most general representation there are nine life cycle phases (GERAM, 1997): identification, conceptualization, requirement engineering, design, implementation, building, operation, change/reengineering and end of life. If we think about business entity "enterprise product", we have to recognize and develop integrated computer supported processes for market analysis (identification, conceptualization, requirement engineering); product development (design); production technology design (implementation) and production system building (building); production planing, running and analysis (operation); product technology analysis and improvement (reengineering); production system destruction or transfer (end of life). Also, these diagrams specify allowable order of events and operations. They may be derived from workflow model and vice versa, state and activity diagrams contribute to variety of business process objects.

• Business object structure and behavior specification. Finally, business object structure and behavior must be specified in the form (Fig. 6), where actions differ from operations in that they may specify methods as soon as business processes, which represent joint actions of objects. This phase iterates with business object identification and interaction model development. You must analyze scenarios of interactions between business objects and test them against scenarios of interactions between use-cases. Perhaps some corrections in use-case model may be done.

The overall process is represented in Fig. 2.

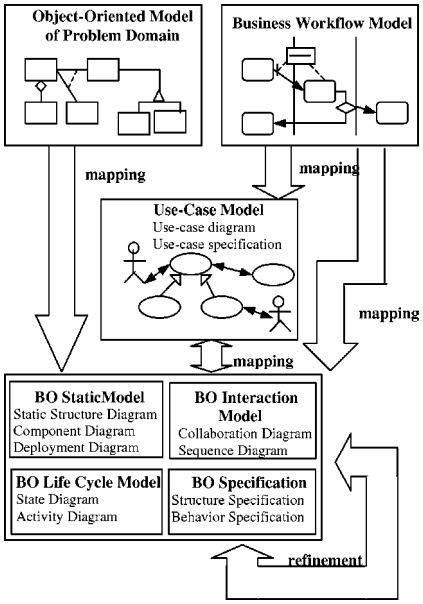

Fig. 2. Business Object model development

4. Framework and Patterns for Business Object Modeling

The framework in this work is meant as a set of classes which are related with structural relationships. It provides a model of interaction among several objects belonging to classes defined by the framework. Framework is a kind of pattern that provides reuse in the large by capturing reusable strategic analysis and design decisions that provide a skeleton on which to build multiple applications. Framework defines a set of patterns, which are the next level in scaling-up abstraction. Using of frameworks and patterns has a sense in all phases of application development – in domain modeling, architectural design, implementation. Reuse of business process models may give the great benefit because it means the reuse of best practice. Of course, in developing applications it is possible every time to begin from scratch and make deep analysis but it takes a long time and it is no guarantee to build successful project. Another way is to use patterns and to win in time and quality.

For the purpose to develop framework for business object modeling a lot of problem domain models common to many enterprises were analyzed – people and organizations, products, ordering, invoicing, work effort, accounting, etc. A good source of these models was ER Data Model Resource Book (Silverston, 1997). During this analysis there were observed some useful patterns concerning problem domain modeling for distributed adaptive enterprise, for example, pattern enabling to represent relationships varying in time (Fig. 3); pattern for object state enabling to trace object states and their completion dates by keeping the required data in data base (Fig. 4); pattern for party role

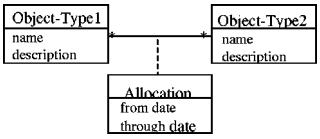

Fig. 3. Time-based relationship for dynamic allocation.

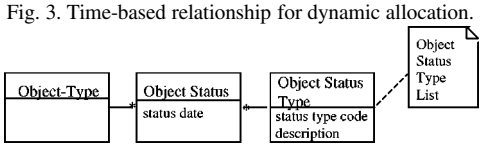

Fig. 4. Object status model.

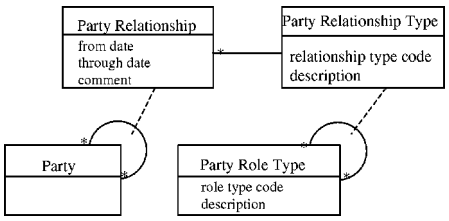

Fig. 5. Party roles are modeled with respect to party relationship.

depending upon relationship between parties enabling flexible representation of internal organizations inside large company (Fig. 5).

Framework proposed for business object and business process modeling is represented in (Figs. 6, 7). First, it is one of possible realizations of OMG Business Object Architecture, and, from another side, it is grounded on analysis of a large set of problem domain models. According to OMG definitions, business process object type has all properties of more general business object type plus additional properties expressed in possibility of being composed of other business processes. The instance of business process constitutes a unit of work which may be executed by single actor in single work place (Fig. 8). This instance may be a step in the thread of activity (workflow under execution). Process instances are created as responses to triggering events; some events are results of instances of business processes. In Fig. 6, it is accepted that goals and rules are modeled as separate objects. Business object may have several representations, for example, business process representation on desktop for user. On the basis of this framework a set of models were developed which may serve as patterns for BOA-based application development in distributed enterprises. In the next section one of such patterns is described.

5. Case Study: Work Effort Modeling

Primary value chain processes of manufacturing enterprises may include (Crove, 1997): process from market knowledge to orders; from customer order to processed order; from concept to successful design/redesign; from unpriced product to final price; from needs

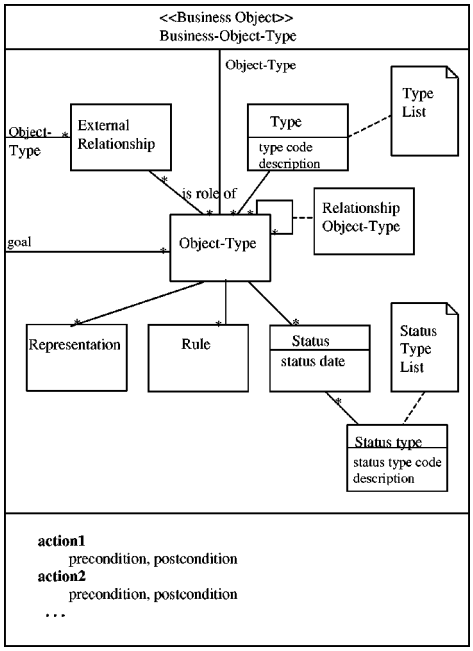

Fig. 6. Business Object Framework.

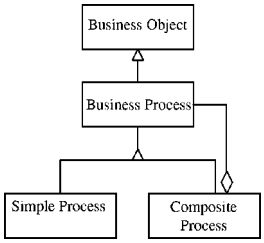

Fig. 7. Business Process Object Framework.

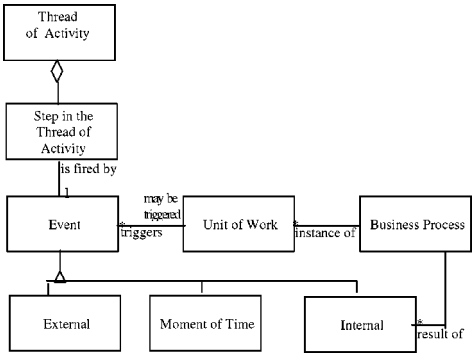

Fig. 8. Process instance (unit of work) as the step of workflow under execution (thread of activity).

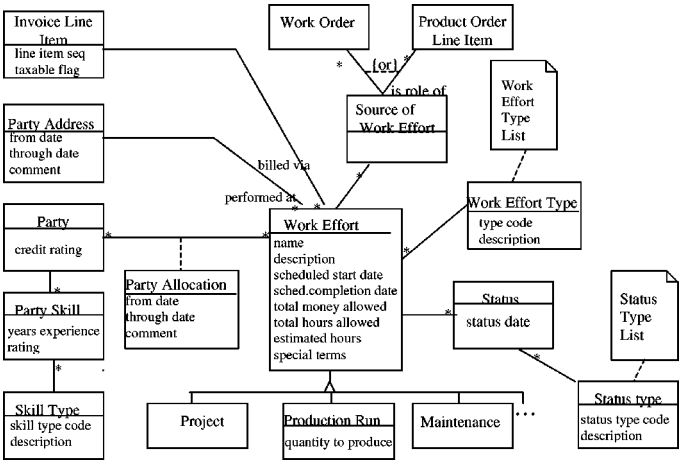

Fig. 9. Object-oriented work effort model.

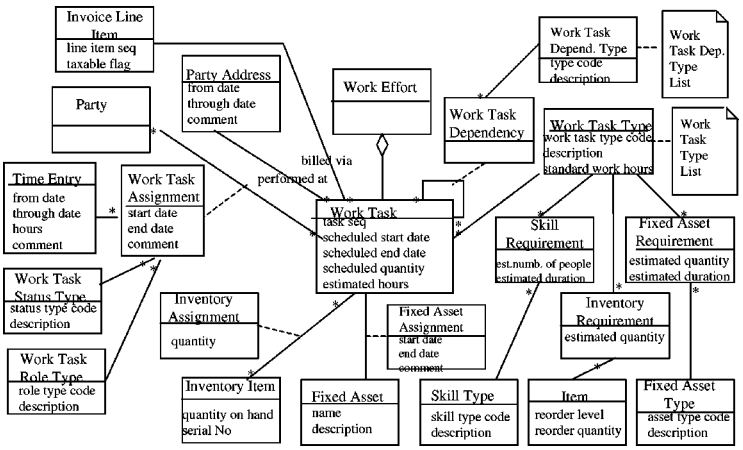

Fig. 10. Object-oriented work task model.

for resources to payment; from raw material to shipped product, from shipped product to payment received and from customer feedback to serviced customer. Processes listed differ in scope and are much more individual for different enterprises than structure of problem domain. However these processes have common parts.

Consider, for example, process “from customer order to processed order”. This process has attracted the greatest attention in information system modeling. Usually there are two types of order model used: “customer order” and “purchasing order”. For distributed enterprise, relevant order object model considers one order object type with two relationships: one relationship to party address from which the order was placed, and second relationship to party address where the order was taken. This is because the same order with respect to party may play role of customer order or purchasing order.

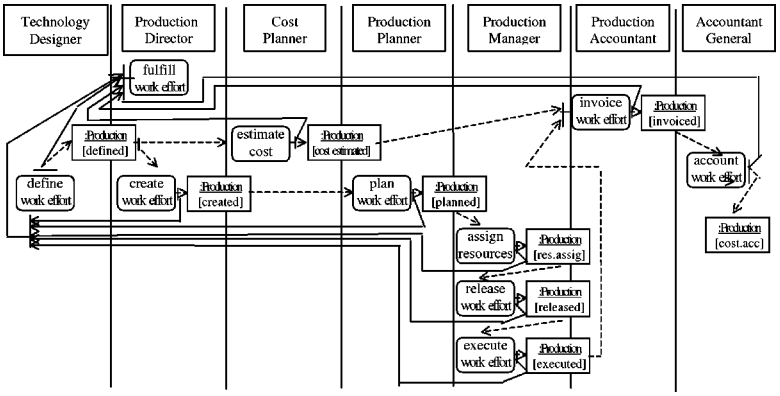

Fig. 11. Production workflow.

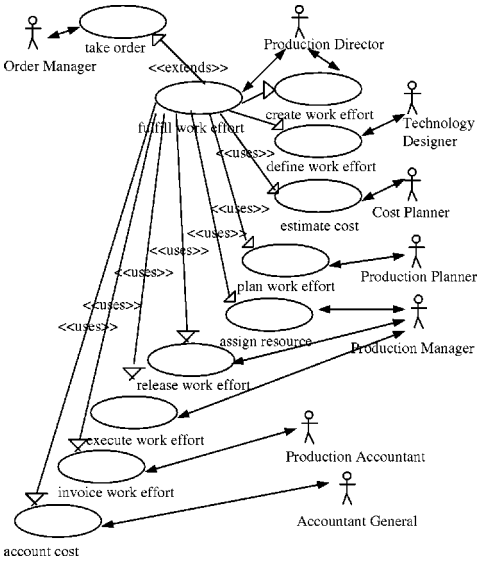

Fig. 12. Work effort use-case model.

Customer order fulfilling may require to delivery product from inventory, to create order for purchasing the product, or to create work effort for manufacturing the product. So in manufacturing enterprise producing products to orders process “from customer order to processed order” is considerably larger in size in comparison with analogous process for commercial enterprise, which is reselling products.

Let we have build object model for work effort (Fig. 9), which consists of work tasks (Fig. 10) and workflow model for the process of production run being a part of process “from customer order to processed order” (Fig. 11). In work effort model, the source of work effort may be work order or customer order product line item, and work effort

subtype list includes “production”, “project”, “maintenance” or other. So this model is more general than required for customer order fulfillment but reflects enterprise needs for the work effort initiation.

Production workflow model mapping to use-case model results in diagram presented in Fig. 12.

Use-case model does not represent the allowable sequence of use-cases executed as workflow steps. So information about processes existing in workflow model may be lost. According to the framework proposed in this paper the overall Business Process “Fulfill work effort” is created, and possible sequences of the rest processes are specified as scenarios of the use-case “fulfill work effort”. Instances of this process are the units of work executable in a single work place of Production Director who initiates and tracks production runs. If such a process does not exist or it is not specified the allowable sequences of use-cases are not defined. The steps of work effort process are the complex processes on their own depending on the properties of products, technology, work practice of enterprise etc. If we consider, for example, process “define work effort”, the execution of this process for concrete production must result in definition of all work task types, work task status types, role types, types of dependencies between work tasks, work task type requirements for skills, fixed assets and inventory items; also the values of a lot of attributes must be estimated, such as “estimated number of people”, “estimated duration”, “estimated quantity”, “standard work hours”, and so on. Sometimes the process “define work effort” may require to fulfill one or more projects for technological process development. Process “estimate cost” may be executed automatically by computer program, but for this a lot of values of attributes must be updated: material purchase costs, work task rates, fixed asset operation costs, and so on. This process must be completed till work effort invoicing. Sequence diagram (part) and Business Process model for work effort fulfillment are presented in Figs. 13, 14, there actions are Business Processes, that must be refined further.

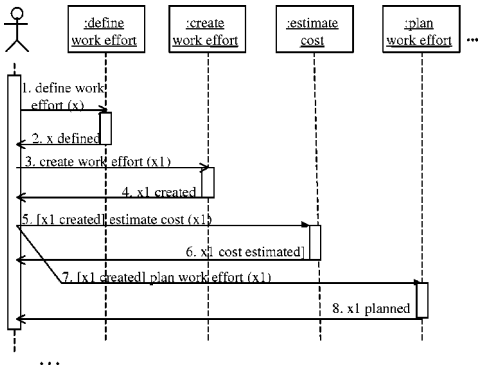

Fig. 13. Sequence diagram for work effort fulfillment (initial part).

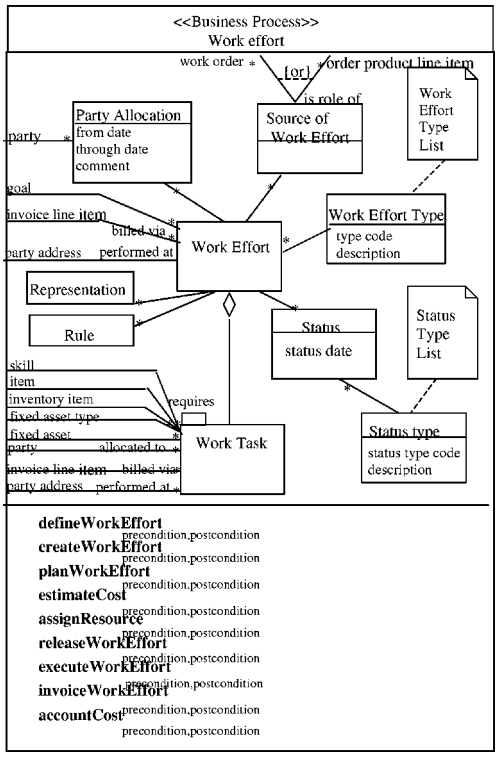

Fig. 14. Work Effort Process Model.

6. Conclusion

Transition to progressive component-based application development methodology must be done using best practice in enterprise problem solution. Framework and patterns, proposed in this paper, combine OMG Business Object Architecture standard with experience in problem domain modeling. The further working with this problem requires elaboration of more comprehensive procedures, including: Business Object discovery; rational employment of business rules; Business Object interaction modeling; precise specification of Business Objects and their interfaces, etc.

References

BOA (1997). *Business Object Architecture (BOA) Proposal.* OMG Document: bom/97–11–09. www.omg.org.

Crowe, T.J., K.Rathi, J.D.Rolfes (1997). *Applying a Taxonomy of Business Processes to Identify Reengineering Opportunities*. University of Missouri, Columbia. www.prosci.com.

D'Souza, D.F., A.C. Wills (1998). *Objects, Components, and Frameworks with UML: The Catalysis Approach*. Addison-Wesley.

GERAM (1997). *An Overview of GERAM*. Report of ICEIMT'97. www.mel.nist.gov/workshop/iceimt97.

Kruchten, P. (1998) *Rational Unified Process: An Introduction*. Addison-Wesley.

Patel, D., J. Sutherland, J. Miller (Eds.) (1998). *Business Object Design and Implementation II. OOPSLA'96, OOPSLA'97 and OOPSLA'98 Workshop Proceedings*. Springer.
UML Notation Guide (1997). Rational Software Corporation. www.rational.com/uml.
Silverston L., W.H. Inmon, K. Graziano (1997). *The Data Model Resource Book*. Wiley Computer Publishing.
Sutherland, J. (1995). *SCRUM Software Development Process*. www.jeffsutherland.org.

L. Nemuraitė is a Doctor of Technical Sciences, a docent of the Information System Department at the Kaunas University of Technology. Research interests include: object-oriented methods for business process and information system modeling and reengineering.

Paskirstytųjų organizacijų veiklos objektų modeliavimo metodas

Lina NEMURAITĖ

Straipsnyje pateiktas paskirstytųjų organizacijų veiklos objektų modeliavimo metodas, kuris remiasi Veiklos Objektų Architektūra bei patirtimi, sukaupta egzistuojančiuose probleminės srities modeliuose. Metodą sudaro organizacijos modelio transformavimo į veiklos objektų modelį procedūra, modeliavimo karkasas ir tipinių modelių rinkinys. Metodas iliustruotas darbų vykdymo proceso modelio sudarymu.