JOURNAL OF FOREST SCIENCE, *51*, 2005 (Special Issue): 13–18

Object-oriented approach in requirement engineering for the analysis of information systems

V. Merunka

Faculty of Management, Czech University of Agriculture in Prague, Prague, Czech Republic

ABSTRACT: One of the major issues of all analysis techniques in information engineering is to capture the intelligible description of processes in the modeled problem. This need is very significant for information systems supporting modern industries and also information systems related to the agriculture and hydrology. In this area, process modeling forms the basis of Business Process Reengineering as a pre-step for subsequent information system analysis, design and implementation. It provides an essential tool to enable software developers, consultants and business users to collaborate to ensure that the necessary understanding of the business context is available to the software developers. In this paper, practically used technique and methodology for process modeling arousing out of software development methodologies will be discussed. The main described method – BORM (Business and Object Relation Modeling), is a result of own research supported by the Know-how Fund of the British Council.

Keywords: object-oriented analysis and design; requirement capture technique; process model; business process re-engineering; information system development; BORM

The attitude of business towards Information Technology (IT) is constantly changing, and increasingly sophisticated. New systems and tools are becoming available. Additionally, there is a constant exchange of ideas between the IT and the business communities, arising out of the development of knowledge-based systems. Today, when modern visual programming tools, combined with the support of rapid web-based application development environments and sophisticated end-user hardware technologies, are available, it would appear that the whole software development process is becoming easier. However, this statement can apply only in those cases where the software complexity of the solution and of the users' requirements is relatively small.

Yet, many systems have a much higher level of complexity, which make development much more difficult (Kotonya, Sommerville 1999). This view is based on our experience with the IT projects we have performed to-date. Rapidly changing regulations, behaviors and the level of the average users' skills in using new communication technologies are creating a situation where IT analysts must expect that all system requirements are not known at the start of the project. The problem is even more complicated because the functions of the built information systems have a great impact on the organizational and management structures and on users' behavior of target area where the system will be implemented.

What are the problems with methodologies?

The major problem here arises in the initial stages of the system development cycle (Cox 1986; Meyer 1988). The initial stage of any methodologies today should be concerned with two tasks. The first is the specification of the requirements for the system. The second is the construction of an initial object model, often called an essential object or conceptual model, built from of a set of domain specific objects known as essential objects. Both these tasks should be carried out with the active participation of the stakeholders, in order to ensure that the correct system is being developed (Davis 1993). Consequently, any

Supported by the Ministry of Education, Youth and Sports of the Czech Republic, Research project No. MSM 6046070904.

tools or diagrams used at these early stages should be meaningful to the stakeholders, many of whom are not 'computer system literate'.

The most common technique for specification of requirements in current object-oriented methodologies is Use Case modeling, and subsequent use of Sequence, Collaboration and State-Chart Diagrams (Fowler, Kendal 1999). This is the foundation of most Object-Oriented development methods. However, this approach is often insufficient by itself to fully support the depths required for initial system specification. Fowler highlights some deficiencies in this approach. Graham says similar ideas (Simone, Graham 1999). There are many views on the effectiveness of Use Cases and related tools as a first stage in System Design. Simone and Graham for example describe a situation where Use Case Modeling obscures the true business logic of a system. Because of standard UML-based tools are too oriented at the world of programming concepts, other methods for business logic and process modeling appeared:

1. The basic grammar of other process modeling tools is based on Petri Nets. The strengths of this approach are that it is both graphical and has strong mathematical basis. A practical implementation of Petri Nets is EPC diagram of Aris methodology, for example (Shriver, Wegner 1987).

2. Another techniques are based on miscellaneous varieties of flowchart diagrams. This approach is the oldest diagramming technique used in computer science. It was primarily user for visualizing the sequences of operations in computer programs. Today, flowcharts are frequently used to model business processes. A practical implementation of flowcharts is workflow diagram used in Proforma Workbench or FirstStep Business CASE Tools. Indisputably, it is also Activity Diagram of UML (Rumbaugh et al. 1999).

3. The third technique used here is the use of state machines. These have the theoretical background, as well as Petri Nets. A practical implementation of state machines is state-chart diagram in UML, for example. Indeed, the sequence diagram of UML has features of state machines as well (Mellor, Shlaer 1993).

The overview of all approaches for modeling business logic and processes described here is presented in following table:

Our approach

The method presented here is Business and Object Relation Modeling (BORM). This method was originally developed to capture knowledge necessary for the development of IT systems, but which has revealed increasing potential for more general knowledge based system development. Work on BORM began in 1993 and was intended to provide seamless support for the building of object-oriented software systems based on pure object-oriented languages, such as Smalltalk, together with pure object databases, such as Gemstone. Today, this method is also recognized for its significant potential to capture knowledge of business processes, business data and business issues. BORM solves problems, when not all system requirements are known at the start of the project and the customer expects that their discovery and refinement will be part of the project. In BORM, it is not difficult to address also the change of these related structures during work on the information systems (Knott et al. 2000).

The BORM approach is based on the fundamental concept of process modeling combined with strong object-oriented approach.

In BORM, any initial analysis diagram supports only problem domain-specific concepts; any software-orientated concepts are left until later in the modeling process. In addition, in the early stages, BORM uses a single diagram that embodies the same information as the numerous diagrams used by other methodologies. This is an attempt to make it easier for the user to form a complete under-

Approach	Theory behind	Advantages	Disadvantages
EPC – Aris	Petri Nets	very popular in Europe, perfectly supported by the Aris CASE Tool, easy and comprehensible method for domain experts	weak relation at subsequent software development techniques, slow analysis, low expressiveness of large models
UML Activity Diagram	flowchart	industry standard, supported by many CASE tools	too software-oriented, difficult to understand by domain experts
UML sequence and state diagram	state machine	industry standard, supported by many CASE tools	too software-oriented, difficult to understand by domain experts
Workflow diagrams	flow chart	easy and comprehensible method for domain experts, perfectly supported by many business CASE Tools worldwide	not very popular in Europe where Aris takes the dominant place, weak relation at subsequent software development techniques

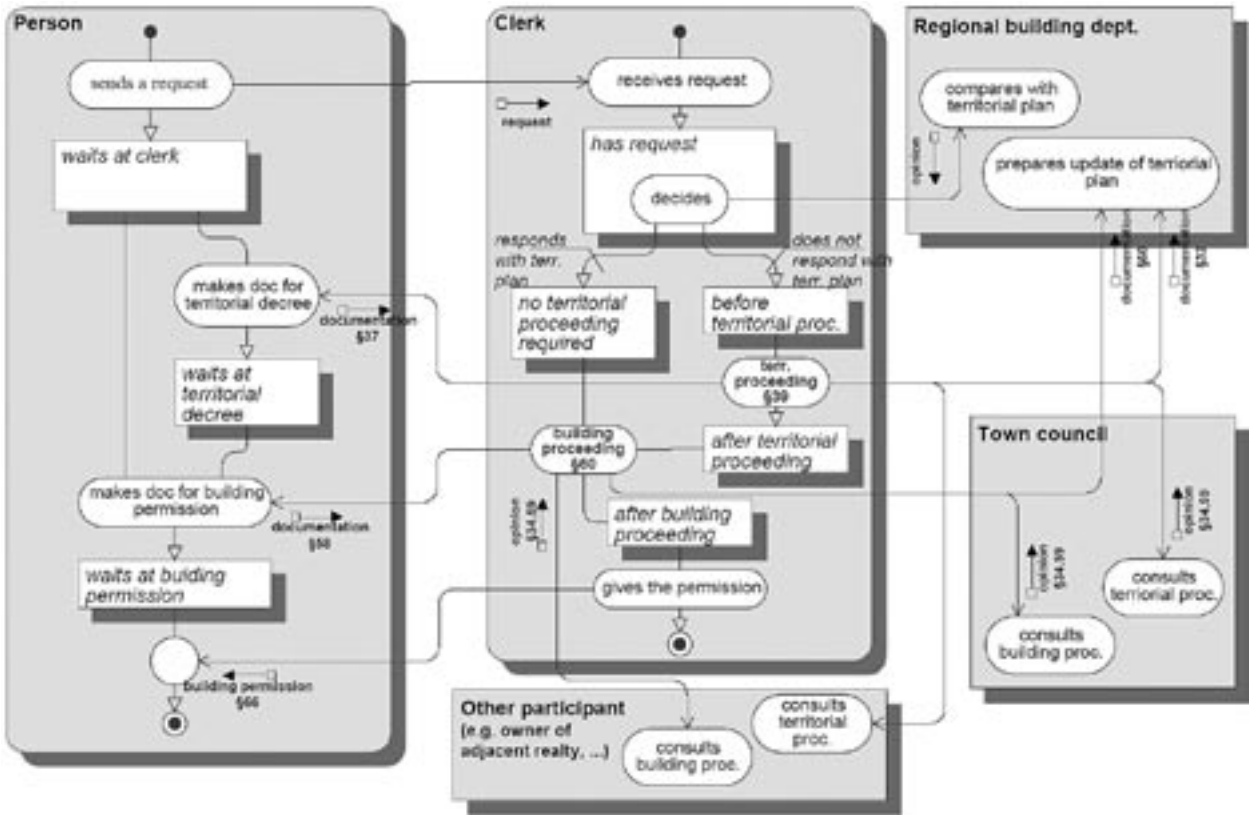

Fig. 1. Business process as a cooperation of participating objects described by state machines

standing of the interaction of the various system components.

BORM concepts and their notation change as the development process proceeds. This is in sharp contrast with UML, which claims to be a universal system in that the same notation is used for analysis, design and documenting the implementation (Derr 1995). Our reasons for changing notation are based on the observation that this universality of the UML's notation hinders the design process. In this we are in broad agreement with the criticism of this aspect of UML expressed by Simone and Graham (1999).

In BORM, every object is viewed as a state machine with states and transitions dependent on the behavior of other objects. Each state is defined by its semantic rule over object data associations and each transition is defined by its behavior, necessary to transform the object from its initial to its terminal state. Consequently, BORM objects have the characteristics of Mealy-type automaton. Business object diagram accents the mutual relationships (communications and associations) of states and transitions of objects in the modeled system (see example in Fig. 1).

BORM follows the process-oriented approach, which has proved to be beneficial in software development. Generally, the process-oriented approach lead to a faster and more comprehensive analysis of the problem being solved.

In our experience, stakeholders from the problem domain are able to understand the BORM approach very quickly – normally a one-hour introduction at the start of analysis is enough. In Deloitte & Touche (Prague office), a business consulting team has worked for the past five years using the BORM system, as well as Aris and other methods. They have found BORM to be on average 3–4 times faster in carrying out the analysis phase, compared to other methods.

The methodology is easily acceptable to domain experts, analysis consultants and developers. BORM is based on a step-by-step transformation of the model. In its each phase, only a limited and consistent subset of modeling concepts is used. This concept progression is depicted in Fig. 2. BORM has been used enthusiastically by Smalltalk and Java programmers and by non-relational object database programmers.

Today, when improved visual programming tools combined with the support of rapid application development environments are available, it would appear that the whole software development process is becoming easier. This statement is true, however, only for those cases where the complexity of the solu-

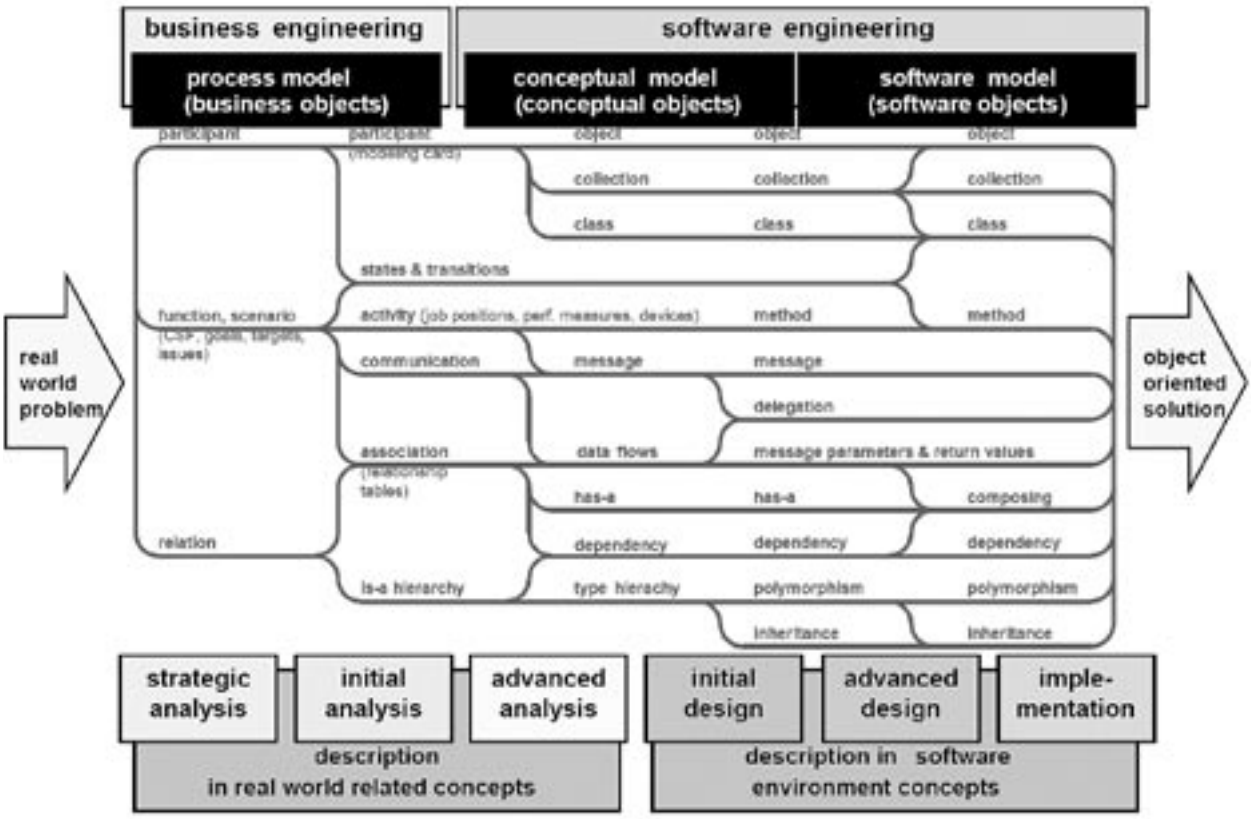

Fig. 2. BORM Modeling Concepts Progression

tion and of users' requirements is relatively simple. Business systems developed for real companies often have a much higher level of complexity which make development much more difficult. Consequently, it is essential (from the software developer's viewpoint) to improve the initial phases of software development.

Until recently, it was correctly assumed that conceptual modeling tools and techniques were used through all stages of project development, from the initial phase to the eventual implementation. However, the position of conceptual modeling is currently being used solely in the implementation phase, as a result of the evolution of software development tools. The analysis is now being performed using newly developed techniques and "business" objects modeling tools.

We believe that Object-oriented programming has changed not only system development but also all of computer science. In software development, any team must be well organized, with clear and common goals. Managers of such projects must be clear about the potential benefits, as well as understand the management of Object-oriented development.

Object-oriented programming can help in the development of a large system by significantly reducing the developing and maintenance time (Taylor 1995). But the adoption of Object-oriented programming requires considerable developments not only in technical knowledge, but also managerial and cultural realignment. Such changes can only be achieved by suitable training, combined with the use of well-designed and easy to use software development tools like those described here.

CONCLUSION

Currently there is not a 'standard solution' to the problem of gathering and representing business knowledge. Our approach, described here, developed out of business experience and enhanced by graphic models with clear connection towards system development seems to be a promising candidate for such a standard. The approach we propose may serve not only as a tool for formal representation of modeled information, but also as we have demonstrated as a useful tool for communicating with developers and experts from the problem domain (managers, employees, etc.). The key advantages of BORM are its graphic models of knowledge representation, which provides easy and effective feedback. There are also clear rules how to progress through the system development process using this knowledge representation.

References

COX B.J., 1986. Object Oriented Programming – An Evolutionary Approach. Boston, Addison-Wesley.

DAVIS A., 1993. Software Requirements – Objects, Functions and States. London, Prentice Hall.

DERR K.W., 1995. Applying OMT – A Practical Guide to Using the Object Modelling Technique, Sigs Books. London, Prentice Hall.

FOWLER M., KENDALL S., 1999. UML Distilled. 2nd Edition. Boston, Addison-Wesley.

KNOTT R. P., MERUNKA V., POLÁK J., 2000. Process Modeling for Object Oriented Analysis using BORM Object Behavioral Analysis. In: Proceedings of Fourth International Conference on Requirements Engineering ICRE 2000, Chicago 2000. IEEE Computer Society Press: 7–16.

KOTONYA G., SOMMER V.I., 1999. Requirements Engineering: Processes and Techniques. New York, J. Wiley and Sons.

MELLOR S., SHLAER S., 1993. Object Lifecycles: Modeling the World in States. Cambridge, MIT Press.

MEYER B., 1988. Object-Oriented Software Construction. London, Prentice Hall.

RUMBAUGH J., JACOBSON I., BOOCH G., 1999. The Unified Modeling Language Reference Manual. Boston, Addison-Wesley.

SHRIVER B., WEGNER P., 1987. Research Directions in OOP. Cambridge, MIT Press.

SIMONE A.J.H., GRAHAM I., 1999. 30 Things that go wrong in Object Modelling with UML 1.3, chapter 17. In: KILOV H., RUMPE B., SIMMONDS I. (eds.), Behavioral Specifications of Businesses and Systems. Amsterodam, Kluwer Academic Publishers: 237–257.

TAYLOR D., 1995. A. Business Engineering with Object Technology. New York, J. Wiley and Sons.

Objektově orientovaný přístup v získávání požadavků při analýze informačních systémů

V. Merunka

Provozně ekonomická fakulta, Česká zemědělská univerzita v Praze, Praha, Česká republika

ABSTRAKT: Jedním z největších problémů technik analýzy informačního inženýrství je najít srozumitelný popis procesů modelovaného problému. Tato potřeba je charakteristická pro informační systémy moderních průmyslových odvětví a také pro zemědělství a hydrologii. V této oblasti procesní modelování představuje základ pro reinženýring podnikových procesů a také je předstupněm následné analýzy, návrhu a implementace informačních systémů. Je to základní nástroj, který umožňuje spolupracovat tvůrcům softwaru zajistit konzultantům a uživatelům nezbytnou míru shody a porozumění nad kontextem problému. V článku jsou diskutovány prakticky používané techniky a metody procesního modelování, které mají původ v technikách softwarového inženýrství. Hlavní popisovaná metoda – BORM (Business and Object Relation Modeling) je výsledkem vlastního výzkumu podporovaného grantovou agenturou Know-how Fund of the British Council.

Klíčová slova: objektově orientovaná analýza a návrh; technika záskávání požadavků; procesní model; reinženýring podnikových procesů; projektování informačních systémů; BORM

Při práci na velkých projektech se analytici informačních systémů setkávají s problémem, kdy funkčnost budovaných rozsáhlých systémů má vliv na vlastní organizační a řídící strukturu podniku nebo organizace, kam se systém zavádí – jsou to například nové či pozměněné pracovní funkce, změna řízení, nová oddělení, nová potřeba legislativní podpory, … . Proto je žádoucí se při práci na informačních systémech zabývat i změnou těchto souvisejících struktur. Těmito problémy se zabývá poměrně nedávno konstituovaný obor aplikované informatiky, který je anglicky označován „requirement engineering". Běžně používané metody tvorby softwaru, ať už jsou či nejsou objektově orientované, se však bohužel touto problematikou příliš nezabývají a spoléhají na to, že procesy systému, jeho požadovaná funkčnost a role jeho uživatelů jsou známy a ověřeny na počátku projektu a že se v průběhu projektu nebudou měnit.

Objektová technologie (OOP) může poměrně jednoduše modelovat jak softwarové systémy, tak i systémy podnikové či organizační. Právě proto, že jedna technologie slouží k modelování obojího, tak lze mo-

delovat podnikový a informační systém ne jako modely dva, ale jako jeden model a změny a vlastnosti procesů přímo promítat do změn a vlastností softwaru a naopak. OOP má všechny předpoklady ke tvorbě takových analýz. Bohužel metodiky a nástroje využívající UML (standard pro objektově orientované projektování) jsou v této oblasti stále na samotném počátku a vesměs předpokládají, že požadavky na informační systém není třeba evaluovat a analyzovat. Uvedené nedostatky se snaží řešit metoda BORM.

Metoda BORM (Business and Object Relation Modeling) je vyvíjena postupně od roku 1993. BORM je možné využít nejen ve tvorbě softwaru, ale i k analýze požadavků na projektovaný systém a na modelování business procesů. BORM lze charakterizovat pomocí následujících tří vlastností:

1. BORM je navržen jako metoda, která pokrývá všechny fáze vývoje softwaru. Velká pozornost je v BORMu věnována úvodním fázím projektu a postupům, jak najít objekty v zadaném problému a zkontrolovat jejich správnost. Techniky z těchto fází BORMu lze používat samostatně pro modelování procesů i takových systémů, které nemají přímý vztah k tvorbě softwaru.

2. BORM pro každou jednotlivou fází životního cyklu využívá v diagramech jen omezenou sadu pojmů. Předpokládá se totiž, že během projektování dochází k postupným přeměnám pojmů na jiné. Nejde jen o postupné zvyšování úrovně detailu ve vytvářeném modelu, ale skutečně o řadu transformací modelu v průběhu životního cyklu.

3. V BORMu je každý pojem reprezentován shodnými symboly bez ohledu na to, jestli se jedná např. o diagramy datové struktury nebo komunikací mezi objekty. BORM používá pro znázorňování konceptuálních a softwarových pojmů většinu symbolů shodně s jazykem UML, ale dovoluje v jednom diagramu znázornit například posílání zpráv mezi metodami různých objektů v různých stavech. Tento přístup dovoluje vyjádřit konzistentním způsobem některé žádoucí detaily softwarové konstrukce, které lze výhodně aplikovat především při návrhu pro čistě objektově orientované programovací jazyky. Tento originální způsob nahrazuje tvorbu více od sebe oddělených třídních, stavových a kolaboračních diagramů a také dovoluje zobrazit větší množství spolu souvisejících informací.

Corresponding author:

Ing. Vojtěch Merunka, Ph.D., Česká zemědělská univerzita v Praze, Provozně ekonomická fakulta,
165 21 Praha 6-Suchdol, Česká republika
tel.: + 420 224 382 272, fax: + 420 224 382 274, e-mail: merunka@pef.czu.cz