Regional management - modeling and simulation approach

Vojtech Merunka[1], Iveta Merunkova[2], Josef Myslin[3]

[1]Department of Software Engineering in Economy Faculty of Nuclear Sciences and Physical Engineering, Czech Technical University Prague, e-mail: merunka@fjfi.cvut.cz
[2]Department of landscape architecture, Faculty of Agrobiology, Food and Natural Resources. Czech University of Life Sciences Prague, e-mail: merunkova@af.czu.cz
[3]Department of Computer science, College of Information Management and Business Administration Prague, e-mail: josef.myslin@vsmie.cz

Abstract. This paper is an attempt to help in the area of regional management. The quality of life of people in cities and villages is dependent of quality of regional management, because the landscape and environment are, today, the main aspects of quality of life. The shortage of knowledge can be problem and the reasons of insufficiently care about landscape. But, especially in the small villages, solution cannot be only in education, but also in making management simpler and in using modern methodologies like modeling and simulation. In this paper we use two approaches known from IT area as methods of regional management and planning.

Keywords: Regional management, modeling, simulation, object normalization, BORM

Introduction

There is fact that landscape is important part of quality of life of people in European Landscape Convention. The progress in techniques of agriculture, forestry, industry and in mining of minerals, but at the first processes in the area of regional and urban planning, transport and infrastructure, and on the general level changes in the world economics has great influence on the landscape changes. Landscape is key element of level of living of people and whole society and its protection is connected with rights and duties. In small villages this fact determines need of knowledge of local representatives and other stakeholders.

But we have problem consist of impossibility for local representatives to be lawyers and experts on the law in this area. The correction should not be in education in law, although this education is not bad, but in making processes simpler and clearer so they will be understandable for people who are not experts or lawyer. This need is all the more urgent, because approximately 80 percent of landscape of the Czech Republic belongs to cadaster of small villages, which don't dispose of great

In: M. Salampasis, A. Matopoulos (eds.): Proceedings of the International Conference on Information and Communication Technologies

for Sustainable Agri-production and Environment (HAICTA 2011), Skiathos, 8-11 September, 2011.

infrastructure and own experts. These villages are depending only upon the people living in this village.

This paper has one goal - present two approaches, know as typical IT approaches, as possibilities how to reach purposes described before.

Our Experience

Our experience in system modeling suggests that classical UML is not suitable for first stages of analysis, where business processes need to be recognized. UML diagrams are too complex for the users from the problems domain community as they often contain too much detail concerning potential software implementations. This means classes, inheritance, public/private methods, attributes, link classes, etc. Almost the same experience we have is documented in Simone and Graham [27]. The UML is suitable in next stages of analysis and design, where we need to show the structure of the system and structure of modeled reality. But we have to understand UML diagrams and models not only in IT way, but we have to find out the business aspects of models. If we try to say this in another way, we have to show our customer what he can know and understand.

We believe that the business community needs a simple yet expressive tool for process modeling; able to play an equivalent role to that played by Entity-Relation Diagrams, Data-Flows Diagrams or Flow-Charts over the past decades. One of the strengths of these diagrams was that they contained only a limited set of concepts (about 5) and were comprehensible by problem domain experts after few minutes of study. Unfortunately UML approach lost this power. In this paper we try to explain how we can use UML (with corrections and with business approach) in better way.

That is why we developed our own BORM process diagram and our own way to start business system analysis. It is a simple methodology going smoothly from business analysis and simulation to subsequent detailed UML software design based on MDA software-oriented concepts necessary for the construction of software-oriented conceptual model. BORM process diagrams are useful in the first stage of development. Then we try to use common approach known as object normalization [33] for next stages.

System Development

Developing systems is a complex activity fraught with many difficulties for software engineers as they endeavor to ensure that the right system is built. A right system being one that meets the user’s needs at a cost they can afford.

On the surface this would appear a straightforward task, first year university students studying system design are often surprised when it is pointed out to them that incorrectly specifying the required system is one of the major causes of software systems failure. Such students, however, have little experience of the complexity of the real world where software developers and experts from the user domain appear to live in different universes, each with their own jargon, which acts as a barrier to true communication.

It is in this context that software developers face the first and perhaps major challenges of software development; to fully understand the user domain and moreover to convey their understanding of that domain to the user.

Adele Goldberg [14] uses the term “concept space” to describe what the user/experts believe, assumes or knows to be the case. The “articulation space” is what the expert/user communicates in response to the analyst’s questions. The analyst then constructs a model to feed back to the user/expert their mental model of the concept space, which they construct out of the information presented in the articulation space. The difference between this analyst’s model and the user space is the concept gap.

To a certain extent, part of this gap is unbridgeable; we cannot easily reduce the gap between concept and articulation space as these exist in the user/expert’s head. It is true, however, that the languages, natural and graphical, used by the analyst in representing this model, are a vital component in the user/expert’s ability to validate this model against the users own concept space.

The problem is to find a common language for the developers to express their understanding of the problem space that is both sufficiently rich for the developers to fully articulate their ideas while also being comprehensible to users from all areas of discourse. Well-formed graphical diagrams should reduce this problem. But the main condition is, that the diagram shows real structure, which is known for our customers.

Use-Case has become a well-accepted part of object-oriented analysis and in many cases has proved a useful mechanism for communication between developers and domain experts. We do not intend to discuss it further here. However, Fowler [12] highlights some deficiencies in the Use-Case approach and also suggests "Activity diagrams can be useful in cases in which workflow processes are an important part of the users’ world."

Same as [7] we think that activities are a key component of business process modeling. Eeeles and Sims [9] define a business process consisting of a number of elements; activities, transitions, states and decisions. They state that the UML activity-diagrams can be a useful modeling tool in capturing business processes as well.

Initial analysis diagram should support only problem domain-specific concepts; any software-orientated concepts can be left until later in the modeling process. This is in sharp contrast with UML, which claims to be a universal system; meaning that the same notation is used for analysis, design and documenting the implementation. Our reasons for this requirement 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 Simons and Graham [27].

It is necessary for the organization modeling and subsequent simulation, that every participating object should be 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. Organizational and business process models must be able to be simulated. Hence it should accent the mutual relationships (communications and associations) of states and transitions of objects in the modeled system.

The BORM Approach

Motivation

Development of the BORM methodology started in 1993. At that time, several "first generation" object or semi-object-oriented analysis methods (OMT, Martin-Odell, Booch, Coad-Yourdon, Jacobson, etc.) existed. These methods were, and still are, very useful for the development of hybrid software systems. For example, an object-oriented client, which collaborates with several relational servers. However the authors felt that these methodologies possessed two fundamental weaknesses, which made them inappropriate for their own development requirements.

Firstly these existing methods did not offer sufficient support for development using a pure object-oriented language like Smalltalk. When developing systems in Smalltalk the authors often used constructs of the language like polymorphism between objects without any inheritance or object dependency, which were not supported and could not be expressed in any of these existing development methodologies. Also in the diagrammatic notations they provided it was impossible to represent most pure object-oriented algorithm. Such algorithms may often be described as mutual asynchronous communications (message passing) between objects, which as the result of receiving messages invoke internal methods with a consequential change in their state.

Secondly, these existing methodologies initially commenced with the construction of a set of classes showing inheritance and aggregation hierarchies. While this is an effective way of expressing the structure required for subsequent coding in an object-oriented language, it is not however effective in illustrating the problem domain. This is because the "object oriented nature" of these diagrams are difficult for domain experts, not educated in computer science concepts, to understand. Consequently such diagrams cannot be used in describing proposed solutions to clients.

BORM Projects

The initial work on BORM was carried out under the support of the Czech Academic Link Program (CZALP) of the British Council, as part of the VAPPIENS3 research project; further development has been carried out with the support of Deloitte Central Europe. (The British Governments CZALP, administered by the British Council funded VAPPIENS. The authors acknowledge the support they received from this source, which enabled them to meet and carry out the initial work, out of which BORM grew.) BORM has been used for a number of large projects including

- the identification of business processes in Prague city hospitals,

- the modeling of properties necessary for the general agricultural commodities wholesale sector in the Central European region,
- as a tool for business process reengineering in the electricity supply industry and
- as a tool for business process reengineering for telecommunication network management in the Central European region.

BORM fundaments

BORM is a unified approach to business and IT system modeling. For more on the BORM method see [18, 20].

BORM is based on the spiral model for the development life cycle as described in [5]. One loop of the object-oriented spiral model contains stages of strategic analysis, initial analysis, advance analysis, initial design, advanced design, implementation and testing.

1. The first three stages are collectively referred to as the expansion stages. Expansion ends with the finalizing of the detailed analysis conceptual model, which fully describes the solution to the problem from requirements point of view.
2. The remaining stages are called as consolidation stages. They are concerned with the process of developing from "expanded ideas" to a working application. During these the conceptual model is step by step, transformed into a software design.

Object-oriented approach.

The object-oriented approach has its origins in the researching of operating systems, graphic user interfaces, and particularly in programming languages, that took place in the 1970s. It differs from other software engineering approaches by incorporating non-traditional ways of thinking into the field of informatics. We look at systems by abstracting the real world in the same way as in ontological, philosophical streams. The basic element is an object that describes data structures and their behavior. In most other modeling approaches, data and behavior are described separately, and, to a certain extent, independently. OOP has been and still is explained in many books, but we think that this one [14] written by OOP pioneers belong to the best.

Automata theory.

In the field of theoretical informatics, the theory of automata is a study of abstract automatons and the problems they can save. An automaton is a mathematical model for a device that reacts to its surroundings, gets input, and provides output. Automatons can be configured in a way that the output from one of them becomes input for another. An automaton's behavior is defined by a combination of its inner structure and its newly - accepted input. The automata theory is a basis for language and translation theory, and for system behavior descriptions. Its usage for modeling and simulation in software engineering activities has been described in [26] and many newer publications. The idea of automata also inspired behavioral aspects of the UML standard [29].

Three areas of BORM modeling in MDA perspective.

MDA (Model-Driven Approach) is a software development methodology. It provides a set of guidelines for the structuring of specifications, which are expressed as step-by-step transformed models. It was created by the Object Management Group (OMG) in 2001 and is the most used software methodology based on the UML (Unified Modeling Language)[29]. BORM can be regarded as a special kind of MDA. In the MDA terminology, we can describe BORM as:

1. The CIM (Computer-Independent Model) modeling, according to the BORM method, is a visualization of the environment in which a project is being executed. It deals primarily with business process models. Its aim is to understand and describe a problem and find a solution. A well-made CIM model enables proper descriptions of settings for information system to be made; a necessary condition for a designed solution. This part of BORM having the special BORM process diagram used for the organizational modeling and simulation is discussed in this paper.
2. PIM (Platform-Independent Model) modeling, according to the BORM method, is a visualization of the required information system in software engineering concepts. The UML (Unified Modeling Language) standard has an important role. There is a set of transforming rules [22] from BORM model to the conceptual UML model [17].
3. The PSM (Platform-Specific) model is a revised form of the PIM model which, unlike PIM, enables specific software implementation, since it includes specific properties of the target environment and reused artifacts of the IT architecture, etc. There is also a set of transforming rules from PIM UML models to the PSM UML models [17].

BORM CIM — organizational modeling

The first part of the method (CIM) covers the organizational modeling. It transforms a project assignment into a model described by miscellaneous hierarchies, process participants, process scenarios, various diagrams and generated reports. The main instrument of verification and validation is the process simulator, which is currently implemented in the Craft.CASE tool [6].

For the following purposes, it is possible to use this part of BORM without any relation to a software engineering phase or organizational structure improvement as is it also presented in the example of this paper. BORM CIM modeling has been used as:

1. Projects documenting processes and organizational structure. These are, for instance, projects whose aim is knowledge management, creating training materials, knowledge visualization, etc.
2. Projects for preparing the groundwork for selection procedures for organizational consultancy, or other consultancy services.

3. Projects for preparing the groundwork for selection procedures for the delivery of information systems, or other software engineering projects.

BORM was initially developed as an object-oriented method for the analysis and design of object-oriented software systems. The process (described by Satzinger [25]) starts from an informal problem specification and provides both methods and techniques, to enable this informal specification to be transformed into an initial set of interacting objects. The tools and techniques developed for requirement analysis and used in the initial phases of BORM, provide an independent method for business process modeling as part of business process reengineering. The authors find that this independent method, referred to as BOBA (BORM Object Behavior Analysis) is frequently used alone.

One advantage of this approach is that it provides a close interactive interchange between the developers and members of the user's organization. As well as identifying initial objects, BOBA elicits from the domain experts, detailed descriptions of their requirements which are fed back to them via easily understood descriptions of the proposed system's behavior using a number of tables and graphs.

The problem specifications from which the process starts are obtained from relevant parties in the problem domain by interviewing. This determines a list of required system functions, which are essentially Use Cases. From this list, a set of system scenarios is formed. BOBA scripts always include at least the four sections shown in Table 1.

Table 1: Scenario structure in BORM.

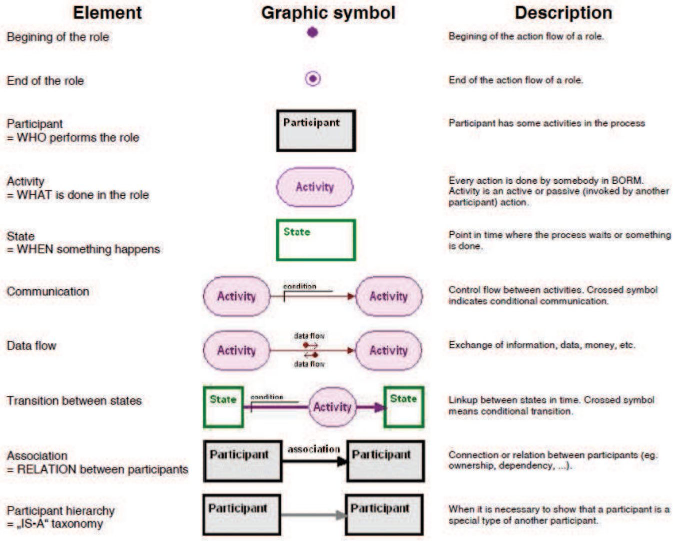
section name	description
initiator	A brief verbal description of the beginning of the scenario including any inputs or entry conditions. It also describes the first event or first activity of some element within the process.
action	A verbal description of the process itself.
participants	The set of those members of the system, which are required for the action. It is often the case that the same participants may be present in several processes of the modeled system.
result	A brief verbal description of the end and outputs of the scenario.

This structure represents the four most important attributes of each scenario. The complete set of scenarios is capable of describing system behaviors, as well as determining the objects that perform these behaviors. In addition to those four attributes each scenario must also refer to the required system function it realizes.

BORM business diagram

BORM uses an original diagram for business process modeling and subsequent simulation (see figure 1). It conveys together information from three separate UML diagrams: state, communication and sequence. The BORM group has found that it is clearly understood by business stakeholders. Main principles of the BORM process diagram are:

1. Each subject participating in a process is displayed in its states and transitions.
2. This diagram expresses all the possible process interactions between process participants. The business process itself consists of a sequence of particular communications and data flows among participating subjects.



Element	Graphic symbol	Description
Begining of the role	●	Begining of the action flow of a role.
End of the role	◉	End of the action flow of a role.
Participant = WHO performs the role	Participant	Participant has some activities in the process
Activity = WHAT is done in the role	Activity	Every action is done by somebody in BORM. Activity is an active or passive (invoked by another participant) action.
State = WHEN something happens	State	Point in time where the process waits or something is done.
Communication	Activity — condition → Activity	Control flow between activities. Crossed symbol indicates conditional communication.
Data flow	Activity — data flow / data flow → Activity	Exchange of information, data, money, etc.
Transition between states	State — condition — Activity → State	Linkup between states in time. Crossed symbol means conditional transition.
Association = RELATION between participants	Participant — association → Participant	Connection or relation between participants (eg. ownership, dependency, ...).
Participant hierarchy = „IS-A" taxonomy	Participant → Participant	When it is necessary to show that a participant is a special type of another participant.

Figure 1. BORM diagram symbols

More formally, BORM process diagrams are graphical representations of interconnected Mealy-type finite state machines of particular subjects. The idea of modelling objects as finite-state machines was firstly discussed in [26]. Visual simulation of a business process is based on market-graph Petri net. This similar approach is described in detail by [3]. Therefore we can show states, transitions and operations for all subjects playing a role in a business process. This is a very powerful, yet simple diagram.

BORM Application Example — Public Regional Management System

One of the recent BORM applications of organizational modeling and simulation was the project of improvement the decision-making on the level of mayors and local administrations. It offers the possibility to model and simulate real life situations in small settlements. The project activities were for modeling; simulation and reengineering processes related to the regional government processes of small towns and villages, and the subsequent development of supporting information systems addressing life situations of local people.

Nowadays we have to solve many problems related to the small settlement development and expansion, landscape care and over-all efforts to improve the quality of life and the level of democracy while preserving the conditions of the sustainable development (addressing living standard, cultural and historic value, agricultural and industrial production, transport infrastructure construction, tourism potential, etc.).

One of the specific problems that our approach can be applied to is the urban sprawl as it is stressed by Frumklin in [13]. The cause of the urban sprawl in the small settlement development is the fact that the elected members of local administrations (e.g. mayors and clerks) are not (and as the logic states they cannot be) fully educated in all the details of law, state and local administration agenda and their effects on living in the settlements. They don't know how to use fully the legislation in favour of the settlements and usually depend on a misleading interpretation provided by their governing bodies and more often by another subjects (usually privately involved in the process in question and thus biased).

Urban sprawl is a phenomenon that emerged in the last decades in the advanced industrial countries (USA, France, Great Britain) and recently also in our country. Inhabitants of affected settlements usually percieve the urban sprawl positively at first, mainly because of the lobbying. It can be described as an uncontrolled expansion of certain kind of urban build-up into the free landscape caused by favourable land prices, demand for cheap but modern estates, etc. Dualny and others write [8] about harmful absorption of original small settlement structures, which causes following negative effects:

1. Pawning of infrastructure development of the original settlement. New inhabitants fulfil themselves and shop only at the place of their work in a metropolis and the settlements are just a kind of sleeping accomodation for them. New inhabitants' lack of interest in contributing to the settlement development leads to misusing of democratic principles of the self administration against the original local inhabitants and inevitably to the rise of social segregation between the original and the new inhabitants.
2. Urban sprawl causes disruption of the cultural and historical value of the settlement, disruption of the ecological stability of the area, deconstruction of the transport infrastructure, loss of touristic attractiveness etc.
3. Loss of the quality agricultural soil.

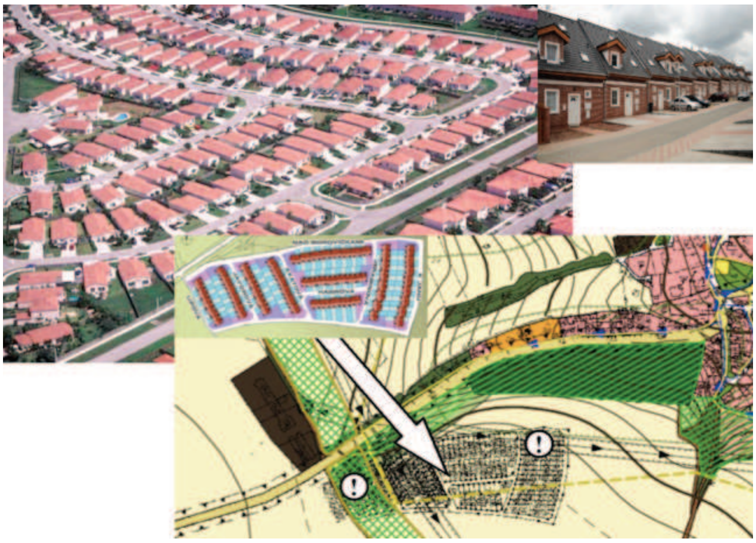


Figure 2. Our project: urban plan having sprawl problem

Modeling and simulation

We analyzed the legislation and local officials' knowledge related to the processes and agendas of the urban planning of the landscape areas and small settlements with regards to the new housing and building law and regional management trends in the European Union. Our approach using process models and their visual simulation helps the officials (especially in the smallest settlements) to clarify the legislation and shows them possible ways of its usage. Our models and their visual simulation show how the BORM can be used to improve the process of decision-making on the level of mayors and local administrations. It offers the possibility to model and simulate real life situations in small settlements. The example at the figure 2 shows the BORM business object diagram of a process of obtaining building permission. The figure 3 shows the concrete simulation step.

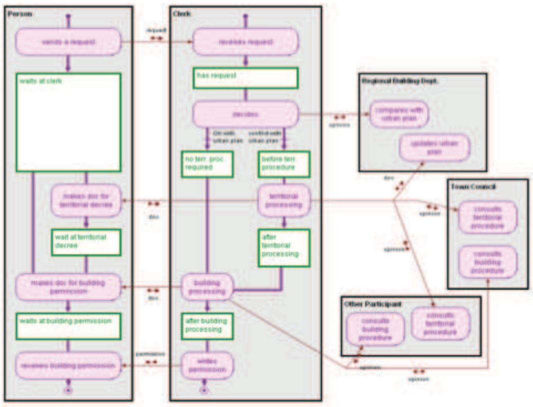


Figure 3. Building permission process

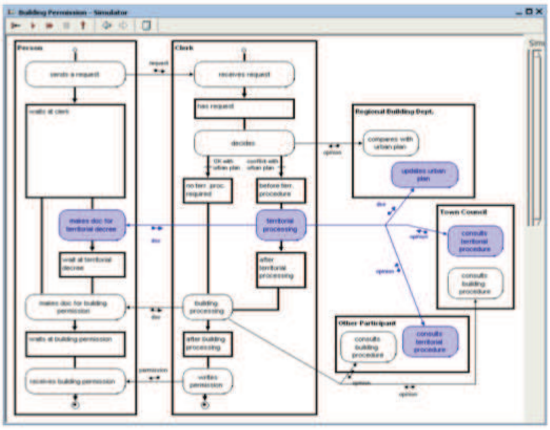


Figure 4. Simulation step example

The object normalization approach

Motivation

Object normalization is a technique that is in recent times already regularly used method for modern information systems´ drafts. Through object normalization we have an opportunity to draft the system in a way that is going to eliminate any redundant information which could case serious problems during system running, such as data inconsistency, during which conceptually same data are appearing in more forms, therefore naturally lowering usability of such system. Moreover, given larger scale it might case that the information system in question is completely incapable of running.

Object normalization has become of the common methods for information systems´ draft. Its importance was so far limited to the IT area itself (as a purely technical method) without investigation of other possibilities of its usage. We personally believe that object normalization, its consequences and, above all, characteristics of respective object normal forms, can be successfully used as tools of bridging known conflicts between IT and business world, i.e. the world of our customers. In this article we will try to show how this method of work could be used and practically applied. We are purposefully going to avoid more than necessary mathematic and other formalization. The reason to this is the fact that formalization and mathematisation are some of the aspects of the dispute between world of technicians and the world of managers we have to communicate with.

What is it about? The basic principle of the method we are about to use must be the fact that after use of this method it is going to be easy for our partners from the business world easier to understand, for example by the way of making resulting model easily understandable and that we are able to defend so far purely technical approach as a method via which we reach even commercially intriguing aim. It has been already suggested that people used from the world of business are used to only minimal amount of abstraction and require the communication with them to be free of abstract terms and approaches as much as possible. Instead of abstraction they ask for actual entities and actual processes. Object normalization can when used appropriately provide a tool for such improvement of communication and understanding. We are going to base my article on the statements that may be found in [33]:

Object model is specialization of much more general conceptual model, that is why object normalization is specialization of conceptual normalization [31]

This definition says that the objective model is specialization of conceptual model, i.e. is based on that model. Conceptual model is result of an ontological on the world around us and it is basically a model of concepts (terms) occurring in the real world. By other words – object model is in figurative sense a model stemming for reality, which can be used e.g. my employment of analogy in teaching OOP. If we accept the thesis that these analogies can contribute to easier understanding of OOP by students (which are by unaware of this paradigm, or they know very little), they we can claim that the same principle can be used when communicating with representatives of the business field, i.e. we have the possibility to use these analogies even here in order to communicate more easily, more effectively, more graphically and more systematically. What is more, not only to communicate, but to understand and consequently on the base of understanding effectively collaborate on the development. Now there is a need to clarify, in what way the object normalization influences an understanding and in what way it can contribute to utilize analogies. If we base the conclusion on the statement (1), then it is immediately clear, that object model is specialized version of conceptual model, therefore object model is only a specialization of model derived from reality, therefore an object model can be understood as a simplified model of reality, which truly corresponds with common understanding of the objective model. In the article we are going to introduce practical example of the objective normalization, which we have taken from (1) and we are going to show on this example than superior object normal forms help not only formal structure from IT professional perspective, but they also form reality in a way that is understood by people from the field of business. We are therefore going to show object normalization is not only a purely technical means, but it can also become an appropriate tool for communication with customers. It is however necessary to realize that object normalization can be just of the tools, not the only, all-covering one.

Object normal forms – simulation of reality in more phases

In order to work with the objective forms and to show in which way they can be useful for communication with the client, it is vital to set an example which we are going to develop further. For example, imagine very small information system which saves information about lots. The most simple way is make list of lots like on the figure 5.

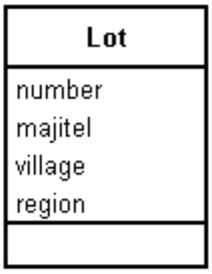


Figure 5. Scheme before object normalization

It can be seen it is very difficult to work with this scheme not only from the IT professional perspective, but this scheme by no means describe any real structure of the common world. For example - imagine that the village will be renamed or two villages will be connected together, all records we have to update. It brings risk because of possibility of forgetting or mistake. For professional from application domain this scheme doesn't represent the real structure, where region is independent object, the village and owner the same. The technical inconsistence is the reason why object normalization exists. But, we can explore that this process can help also in communication between IT professionals and customers.

The object normalization is process consisting of several stages. The result is the third object normal form. Definitions of normal forms are:

The class in the first object normal form (1ONF) when its objects do not contain a group of repeating atributes. These atributes need to be separated to the objects of a new class and the group of repeating atributes must be replaced by one connection to the collection of the new class objects.Scheme is in the 1 ONF when all classes of objects within are in the 1 ONF. [31]

The class in the second object normal form (2 ONF) when all of its objects do not contain any atribute or a group of atributes that would have been shared with some other object. The shared atributes have to be separated to a new class object and they have to be replaced by connection to this new class object in all object they appeared.Scheme is in the 2 ONF when all classes of objects within are in 2 ONF.[31]

The class in the third object normal form, when all of its objects do not contain any atribute or a group of atributes that have the meaning independent from the objects they are contained in. If any such attributes exist, they have to be separated to a new class object and they have to be replaced by connection to this new class *object in all object they appeared.Scheme is in the 3 ONF when all classes of objects within are in 3 ONF. [31]*

The first and second object normal forms are weaker in definition and this forms are steps to final third object normal form. Our scheme converted in the third object normal form is on the figure 6. The different is that all in reality independent objects are independent in scheme. There is not possibility of inconsistence, redundancy and similar problem which occurs in schemes which are not in the third object normal forms.

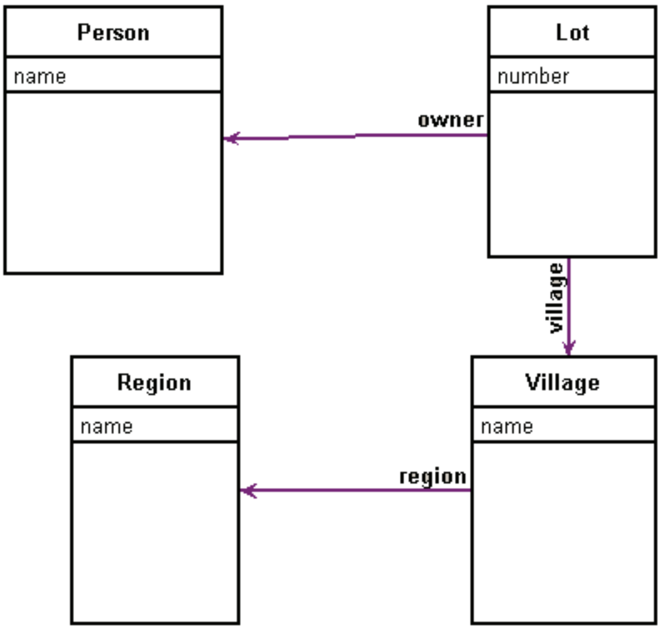
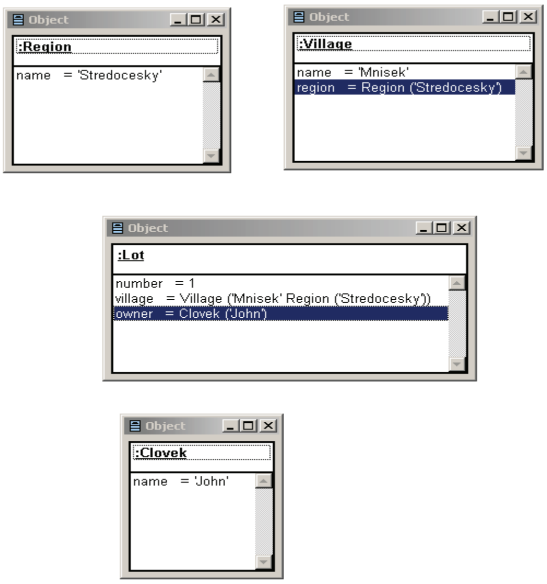


Figure 6. Object scheme after normalization (3rd ONF)

This scheme is, of course, better for technical employees. But it is also better for communication and for people from business. The question is about the reason of this fact. The scheme in third normal form is closer to reality. In our situation – the lot is in some village (village is object independent of lot) and the village is part of some region (and region is independent object). Every lot has owner and owner is again independent. Person can be not only owner of one ore more lots, but it can be, for example, mayor of the village or region. In reality regions, villages, lots and persons are object and the same structure is better for understanding of structure of system by businessmen.

Modeling and simulation

This approach is not so known and so widely used, so we use only theoretical example for demonstration of this approach. The model is the same as we described in paragraph before. For modeling we use software Daskalos [38]. In this software was modeled the scheme you have seen on figure 8. But this is the static model and we want to simulate real structure - so we can instantiate classes in software Daskalos. This instances are shown on figure



:Region

name = 'Stredocesky'

:Village

name = 'Mnisek'
region = Region ('Stredocesky')

:Lot

number = 1
village = Village ('Mnisek' Region ('Stredocesky'))
owner = Clovek ('John')

:Clovek

name = 'John'

Figure 8. Dynamic object model

There is small region with one village and one lot modeled. Village is part of region and lot is part of village. Every object has connection to other objects and all objects create closed structure, very similar to real structure of region. Now you can simulate the real situation. You can, for example, make list of lots depending on some criteria. Daskalos contains programming language based on lambda calculus and Smalltalk, so you can create queries and other structures.

Conclusion

The paper tried to present two approaches, which should be useful in regional management. These approaches are widely used, but in the area of pure computer science or software development. Our efforts are about extends using this methods to the area of regional management and generally, to the area of public service. It the general level - the information and its processing we can found in all areas of peoples activity. And computer information system as computer representation of information processing is something as extension.

Of course, we can find some problems with application of these methods. The problems are in the fact that this methods are not known in the area of public service in the level we need. And the second problem should be that these methods are not constructed for public services and in some situation they are not suitable. These problems have to be solved - theoretically, this means some papers and new methods, as well as practically, it means by education.

References

1. Ambler, S.: Building Object Applications That Work, Your Step—By—Step Handbook for Developing Robust Systems Using Object Technology, Cambridge University Press & SIGS Books, ISBN 0521648262 (1997)
2. Barjis J.: Developing Executable Models of Business Systems, in: Proceedings of the ICEIS - International Conference on Enterprise Information Systems, pp. 5-13. INSTICC Press (2007).
3. Barjis, J., & Reichgelt, H.: A Petri Net Based Methodology for Business Process Modeling and Simulation, in the proceedings of the Fourth International Workshop on Modeling, Simulation, Verification and Validation of Enterprise Information Systems (MSVVEIS), Paphos Cyprus, ISBN 972-8865-49-X (2006)
4. Bellin, D.; Simone, S.S.: The CRC Card Book, Addison-Wesley, ISBN 0-201-89535-8 (1997)
5. Boehm, B. W.: Software Engineering Economics, Prentice-Hall, Englewood Cliffs NJ (1981).
6. Craft.CASE analytical and modeling tool, Craft.CASE Ltd. Torriano Avenue London, http://www.craftcase.com
7. Darnton, G., Darnton, M.: Business Process Analysis, International Thomson Publishing, ISBN 1-861-52039-5 (1997)
8. Dualny A.: Suburban Nation: The Rise of Sprawl and the Decline of the American Dream, North Point Press, ISBN 978-0865476066 (2001)
9. P. Eeles P., Sims O.: Building Business Objects, John Wiley & Sons, Inc., New York (1998).
10. EPC — Event-driven Process Chain, http://en.wikipedia.org/ wiki/ Event-driven_process_chain
11. Eriksson, H. and Penker, M.: Business Modeling with UML, John Wiley and Sons, ISBN 0-471-29551-5 (2000)
12. Fowler M, UML Distilled: Applying the Standard Object Modeling Language, Addison Wesley, Reading Mass (1997)
13. Frumkin H. et al.: Urban Sprawl and Public Health: Designing, Planning, and Building for Healthy Communities, Island Press, ISBN 978-1559633055 (2004)
14. Goldberg, A. and Rubin K.: Succeeding with Objects — Decision Frameworks for Project Management, Addison Wesley, ISBN 0-201-62878-3 (1995)
15. Hall, J. A. et al.: Accounting Information Systems, Cengage Learning EMEA, ISBN 0324560931 (2004)
16. Jacobson, I. et al.: Object—Oriented Software Engineering — A Use Case Driven Approach, Addison Wesley, ISBN 0-201-54435-0 (1991)
17. Knott, R. P., Merunka, V., Polak, J.: Process Modeling for Object Oriented Analysis using BORM Object Behavioral Analysis, in Proceedings of Fourth International Conference on

Requirements Engineering ICRE 2000, Chicago, USA, IEEE Computer Society Press, ISBN 0-7695-0565-1 (2000)

18.Knott, R. P., Merunka, V., Polak, J.: The BORM methodology: a third-generation fully object-oriented methodology, in: Knowledge-Based Systems Elsevier Science International New York, ISSN 0950-7051 (2003)

19.Kotonya, G. and Sommerville, I.: Requirements Engineering: Processes and Techniques, John Wiley and Sons, ISBN 978-0-471-97208-2 (1999)

20.Liping Liu, Borislav Roussev, et al.: Management of the Object-Oriented Development Process - Part 15: BORM Methodology, Idea Publishing, ISBN 1-59140-605-6 (2006)

21.MDA — The Model Driven Architecture, OMG — The Object Management Group, http://www.omg.org.

22.Merunka, V., Brozek, J., Nouza, O.: Automated Model Transformations Using the C.C Language, in Proceedings of the International conference EOMAS 2008, Montpellier, France, Springer LNBIP, ISSN 1865-1348 (2008)

23.MetaCase - Domain-Specific Modeling with MetaEdit+, http://www.metacase.com

24.Partridge C.: Business Objects - Reengineering for Reuse, Butterworth-Heinemann, ISBN 0-7506-2082-X (1996)

25.Satzinger J. W. and Orvik T. U.: The Object-Oriented Approach - Concepts, Modeling and System Development, Boyd&Fraser (1996)

26.Shlaer, S. Mellor, S.: Object Lifecycles: Modeling the World in States, Yourdon Press, ISBN 0136299407 (1992)

27.Simons A. J. H. and Graham, I.: 30 Things that go wrong in Object Modelling with UML 1.3, in: Behavioral Specifications of Businesses and Systems eds. H Kilov, B Rumpe, I Simmonds, Kluwer Academic Publishers, 1999, 237-257 (1999)

28.Taylor, D. A.: Business Engineering with Object Technology, John Wiley ISBN 0-471-04521-7 (1995)

29.The UML standard, OMG — The Object Management Group, http://www.omg.org, ISO/IEC 19501.

30.Yourdon, E.: Mainstream Objects — An Analysis and Design Approach for Business, Prentice Hall, ISBN 0-13-209156-9 (1996)

31.Merunka V., Unal B., Myslin J., Formal techniques of data structure design in modern database systems. IFAFFE'10 Proceedings. Samsun, Turecko 2010, ISBN 978-975-7636-71-7

32.Merunka V., Objektové modelování, Alfa nakladatelství s.r.o, Praha 2008, ISBN 978-80-87197-04-2

33.Molhanec M., Krátká úvaha o normalizaci. Sborník konference Objekty 2009. Hradec Králové 2009. ISBN 978-80-7435-009-2.

34.Myslin J., Využití analogií při výuce OOP, Sborník konference Informatika XXIII/2010, Luhačovice 2010

35.Unal B., Myslin J., Business Process Modeling and Business to IT Transformation, Sborník konference Doktorandské dny, Praha 2010

36.Myslin J., Transformace modelů RUP pro potřeby manažera, Sborník katedry informatiky VŠMIE 2/2009,

37.Wiegers Karl. E., Požadavky na software. Computer Press, Praha 2008, ISBN 978-80-251-1877-1

38.Daskalos software, http://kei.vspj.cz/~richta/courses/obm/daskalos