Agents and Their Behavior's Reconfiguration

Michal Radecký
Ivo Vondrák
Faculty of Electrical Engineering and Computer Science
VSB-TU Ostrava
17.listopadu 15, Ostrava, Czech Republic
e-mail: michal.radecky@vsb.cz, ivo.vondrak@vsb.cz

KEYWORDS
Agent, MAS, Behavior, Process, UML, Activity Diagram

ABSTRACT

The demands for modern, powerful and advanced systems or applications are increased, nowadays. Several current solutions and technologies can be used to comply with these enhanced requirements. The *Multi-Agent System (MAS)* technology is one of these solutions.
A recent point of view on software development process offers many tools and methods for specification, analyze, design, implementation and maintain applications. These facilities can be used for standard applications development, as well as for the systems based on MAS technology. In the case of the multi-agent systems, some of these standard approaches have to be adjusted or extended. This paper describes the ideas and methods to model and develop MAS, based on the internal agent behavior and intelligence of particular agents. This paper is concerned with analyze of the MAS from the agent internal behavior point of view.

Introduction

The MAS technology is quite youthful field of computer science that occurs on the borders where "Software Engineering", "Artificial Intelligence" and other computer, natural and social science areas meet. The MAS technology is based on the concepts of the Complex Systems (e.g. macromolecules, ants colony, economical systems), and on the facilities and capabilities of software information systems (Kubik 2004).

The essential properties of MAS are

- autonomy and intelligence of elements
- communication among elements
- mobility of elements
- decentralization of the control
- adaptability, robustness, etc.

The MAS can be formed as an general information system that is composed from a number of autonomous elements (called *Agents*). In this context, the Multi-Agent System is a framework for agents, their lives, their communication and mobility and it is an environment where the goals of the particular agents should be obtained too. From the perspective of the standard information systems, the *Agent* are components of such systems. These components have a special features like autonomy or intelligence, which is consistent with the essential properties of MAS.

Thus, the *Agents* are the elements of MAS. It is a software entity created in order to meet its design objectives. These objectives are subordinated autonomously with respect to the environment, sensorial perceptions, internal behavior and to the cooperation with other agents.

The typical applications of this multi-agent technology are the large systems with a big volume of decentralization and autonomy of their elements. The examples of such systems could be a transportation and logistical system (vehicles, drivers, dispatcher, etc. are the autonomous and world-wide distributed elements of MAS), system for manufacturing line (robots, transport bands, etc. are the Agents) or some kinds of monitoring system (network elements, computers, etc. are the watched or watch elements). The overall behavior of whole system, that is based on the MAS technology, is formed by internal behaviors of several elements (Agents) and also by the communication between all of these elements.

Classes of the Agents

The essential element of MAS is an Agent. The several types of agents could be found in one Multi-Agent System as well as in the "real world" which is realized by this MAS. These types of agents are able to denote as *Classes of the Agents*, according to the goals, internal architecture and behavior of particular agents. These classes are shown in the figure 1. Each Agent can belong to a given class for its whole life, but the classification of the Agent can be also changed during its life process, as

a reaction to the situations, intelligence, capabilities and behaviors. We speak about "behavior classification" in the case of changing during the agents life (Radecky and Vondrak 2005).

	proactive	reactive
deliberative (route-finding)	**10%** (production agents)	**90%** (classical agent models)
processional (route-discovery)	**90%** (classical agent models)	**10%** (reconfiguration models)

Figure 1: Agents classification

The classes that describe a processional approach in the combination with reactive or proactive perspectives are the main goal of this research. These classes cover the concept of "Intelligent Agents" that is formed on the principles of an internal agent behavior processes. It means that the "classical agent models" and "reconfiguration models" are the subjects of our research.

Modeling of the Agent Behavior

Each Agent is determined by its own objectives and the way to meet these objectives is founded on the internal behavior of a given Agent. Internal behavior of such Agent is specified by the algorithm. The Agent lives, behaves and reacts to stimulus and environment, according to the requirements of the algorithm. Any Agent in the MAS has the main internal life process consequent on the agent's classification. Several templates for this *Primary Process* of agent behavior can be defined for each processional class of the agent ordination. This primary process can be decomposed into a number of sub-processes that refine the internal behavior on. The changing of agent classification during its life is based on sufficient template selection for each process, sub-process as well as primary process. This template selection must take the algorithm structure into account. Nevertheless, the template for primary process is given by default agent classification.

The algorithm structures respective to the particular templates could be follows:

- *Simple proactive agent or behavior* – sequential algorithm structure (just one thread)
- *Simple reactive agent or behavior* – algorithm structure with loop (just one thread processes the incoming requests in the loop)
- *Hybrid reactive/proactive agent or behavior* – parallel algorithm structure (more than one thread, where each thread can be realized as a proactive, reactive or hybrid behavior independently)

It is necessary to take into account the fact that each Agent is an absolutely autonomous element of MAS and thus the internal behavior have to be based only on the processes, activities, knowledge and facilities that belong to a given Agent. Then, the result behavior of whole MAS is formed by communication of separated agents and by the interconnection of several internal agent behaviors. This interaction is realized through the use of message passing adapted to the demands of MAS.

In the context of MAS modeling and agents behavior, the term *Agent* expresses only the "type of agents". The real separate agents are the instances of this type. It is analogous with the terms Class and Object from the object oriented approaches. The real agents (instances of Agent Classes) are not issues of MAS design and modeling phase but they will appear in the implementation phase, simulation and operation of a given MAS.

UML Activity Diagrams Extension

The UML (Unified Modeling Language) is an essential tool for process modeling, both on the business level and analytic level of description (Vondrak 2004; Pilone 2005). It can be applicable for modeling of the internal behavior of agents as well. The UML Activity Diagrams is a standard diagrammatic technique which describes the series of activities, processes and other control elements that together express the algorithm from the dynamical point of view. They are able to capture internal behavior of each Agent, but they are not able to describe the interactions and communication between them. The UML Activity Diagrams are especially suitable for modeling of the agent behavior, though some modifications and extensions are required, in terms of interaction support.

The *Agent Behavior Diagrams* could contain all the elements of the standard UML Activity Diagrams, and moreover some new elements that can be used in the agent modeling process. These new elements are concerned with message passing between agents or with other specific attributes of MAS. In early phases of development, these extensions are supported by the implementation of special "send/receive activities" which include an additional information about message content and message receiver/sender identification, see figure 2. The decision elements from standard activity diagrams are improved too. The modified "decision elements" and their output edges can hold the extra information. This information is usable for message-based determination of following control flow of the behavior led to the agent's objectives.

The simple, clear and formal definition of the internal agent behavior is a precondition for moving to the next phases of the multi-agent software process. Thanks to the new stereotypes established in the *Agent Behavior Diagram* and other additional information, it should be

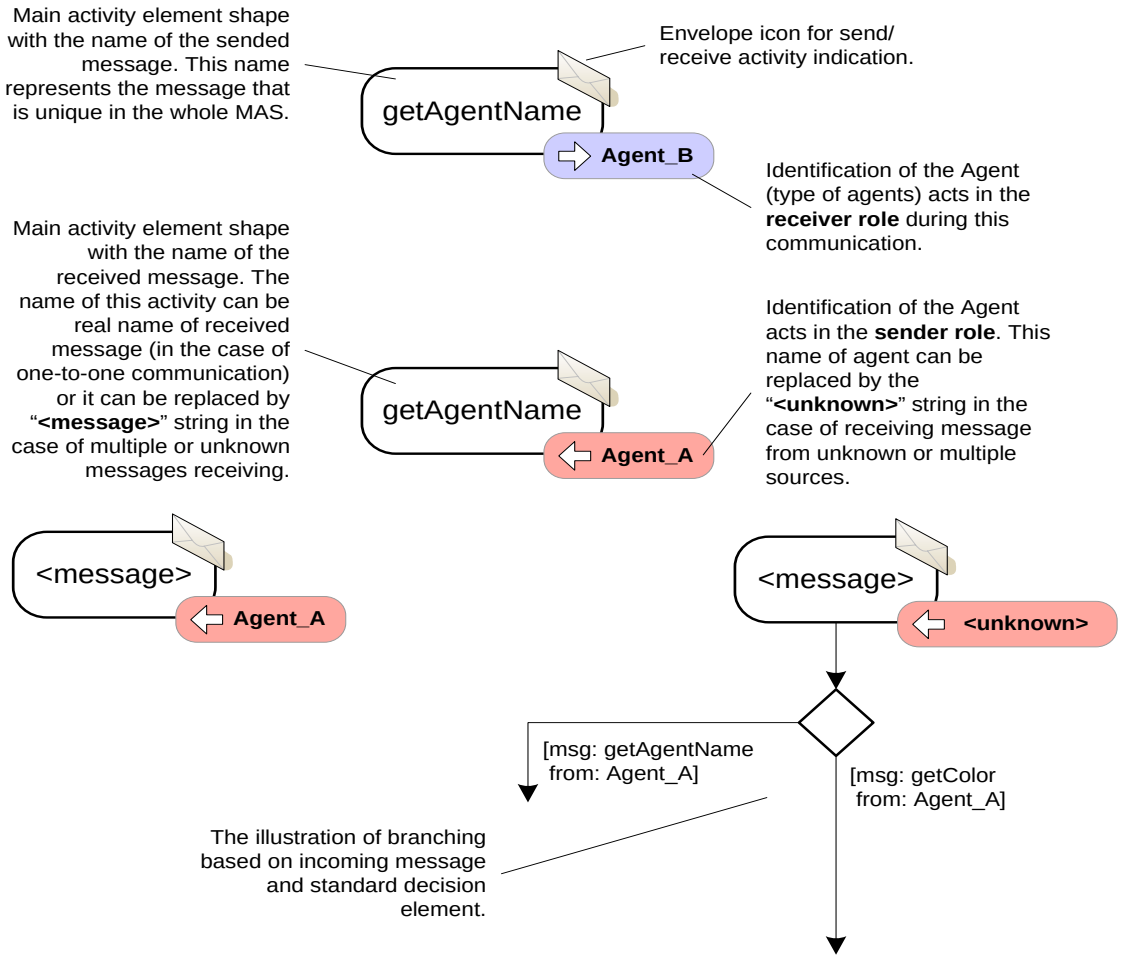

Figure 2: The stereotypes of the new communication activities and illustration of decision point. These send/receive activity nodes are able to use beside the other standard activity and process elements. Internal implementation of these new activities is concerned only with preparation, en/decapsulation and receiving/sending of the message.

able to generate other types of diagrams (e.g. sequential diagrams, maps of the agents communication) as well as the source code templates of the agents and MAS automatically (e.g. interfaces of agents, class and method templates).

As it is mentioned above, the Agent internal behavior is specified by the algorithms expressed in the processes. Each process is modeled as just one *Agent Behavior Diagram* based on UML Activity Diagram. The process can be specified by more than one diagram in the case of multiple realizations belong to a given process. The realizations are the topic of the following text. A couple of rules are bound together with creation process of these diagrams. At first, each process, as well as diagram, have to have just one "initial node" and just one "final node". This prerequisite is necessary for the further connection of the processes together to model overall agent behavior. The demand of the "well-formed" diagrams is also required. There is a set of general structural rules (e.g. the level of split/join nodes preservation, no crossing of the levels of control flow) that should be kept in mind during the drawing of diagrams (Aalst 1997). Thanks to these rules, the well applicable expressions of algorithms are obtained. It is able to verify them, to transform them to other forms or to process them by several formal tools.

Internal Structure of the Agents

The figure 3 illustrates an example of the processional structures of two agents. Each Agent must have one *Primary Process* that covers whole life of a given Agent. This primary process and its realization are unique in the context of a given Agent. Each process, as well as this primary process, can contain a control structures, activities (atomic elements of algorithm) and references to other processes. The usage of "send/receive activities" is the only one way to connect all behaviors of separate agents together. The activities and processes,

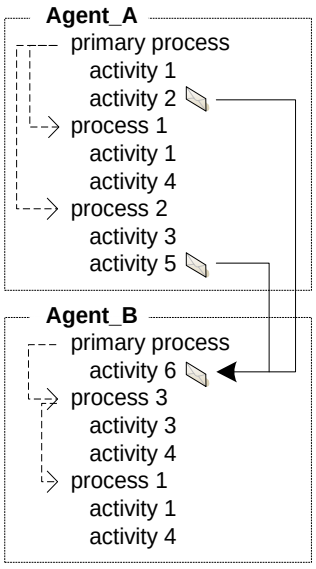

Figure 3: Illustration of the internal agent's structures

except primary processes, can be specified globally in the whole MAS. These globally specified elements are possible to subsume into the behaviors of a set of Agents (types of agents).

Demonstration of afore mentioned situation
For instance, the "moveToPlace" process (`Process 1`) or the "startEngine" activity (`Activity 1`) could be an examples of this approach. In this case, `Agent_A` could be a "Car" and `Agent_B` could be a "Motorbike". So, both vehicles have the same process of moving to a specified place and the same activity of engine starting. These identical elements are subsumed into a number of behaviors; therefore these elements can be specified only once and they can be used multiply. The activities are able to use in a number of processes within the frame of one Agent (`Activity 4` within `Agent_B`) as well as within the whole MAS. The correspondence between activities and processes is based only on definitions, purposes, goals and input/output objects of them but not on the real implementation behind them. This approach is utilizable only for the specification level of MAS modeling and it brings the reusing possibilities of the model elements.

Intelligence within the Agent Behavior

Only the static and structural modeling of agents is mentioned above, though, we want to speak about *Intelligent Agent* too. The *Intelligent Agent* is a standard agent that disposes of certain kind of "brainpower" during its life. These capabilities are hidden inside the agent behavior and they can be found in various points and situations of such behavior. The intelligence is ensured by application of several tools based on logic, artificial intelligence, etc.

The intelligence within the agent behavior can be concerned with three tasks:

- *Intelligence contained in the activities* – this usage is a question of activity implementation phase. The function of the logic is concerned with decision making and derivation within the activity, e.g. weather forecast. This application of intelligence is hidden from the modeling perspective. Only the result of activity execution is relevant for the next parts of process or whole agent behavior.
- *Intelligence of the control flows routing* – this application of several logic or intelligent tools is covered in the decision points. The decision making and derivation are activated whenever the "intelligent decision point" is reached during the process execution. The intelligent routing can be used for all branching that request more complex and knowledge based decision making, e.g. suitable car to a given cargo assignment. The following direction of process control flow depends on the outgoing result of the decision point. The branching is realized by the output edges of decision point and by the edges conditions which represent the possible outputs of the logic tools execution. It is able to substitute this approach with one special activity (with intelligence within it) succeeded by one decision point that uses the results from the previous activity. But the usage of only one "intelligent decision point" could be more clear and straight approach.
- *Intelligent selection of Process Realizations* – The third task of intelligence subsuming into an Agent life is concerned with the real-time running of MAS. The potential of this application is mentioned in the next chapter.

Intelligent Selection of Process Realization

Each Agent must try to realize its tasks and to solve the upcoming situations in order to meet its design objectives during its life. From this point of view, the standard Agent is consisted in definite and constricted description of its behavior already defined during the modeling phase of the Agent. Therefore, there is no way to change the behavior during the agent running. It is able to do this, in the case of *Intelligent Agent*. Such type of agents is based on the main life process model that specifies just the framework of its behavior. The Agent can dynamically change the pieces of its own behavior according to the situations. This principle is denoted as *behavior reconfiguration approach*. The process of reconfiguration is founded on the replacement of a given part of the whole agent's

process by another one that is the most suitable for current situation and conditions. A set of possible applicable behaviors, called realizations (defined by Agent Behavior Diagram), of each "reconfiguration" point (generally the process node of activity diagrams) is defined for this purpose. These definitions of the behavior can be distributed on the whole MAS as *definitions of process realizations*. They can be saved in some global repository or within the particular agents; actually they can be deliberated in the real-time too.

The important and expected situation will appear whenever one process corresponds to the two or more realizations. The logic tools are responsible for solving this situation. But before this state, it is necessary to find the realizations which are applicable to the reconfiguration of a given process. There are two ways to find them – selection based on the process or on the logic grounds.

Pursuant to these ideas, it is able to define such procedure as an algorithm of reconfiguration process.

1. finding a set of suitable realizations for a given process
2. selection of one, the most suitable realization from a defined set
3. realization executing

The first part of this procedure can be solved by two approaches which are mentioned below. A set of process realizations is required for the next step of such algorithm. This second part is concerned with the selection of just one process realization and this selection could be realized by some logic tools. This selection of the most suitable realization is based on its input objects and also on the objects that are occurring within the MAS in the moment of reconfiguration process starting.

Process vs. Logic Approach for Suitable Selection
The implementation of the first algorithm event is a complex problem. It can be implemented by a process or logic approach (fig. 4), where each of them brings different advantages and disadvantages.

The *Process Approach* (figure 4 A) is founded on the idea of assignments of the realizations to the parent process already in the MAS modeling phase. The creation of a needful set is only about surveying of the realizations that are connected to a given process. In this algorithm, it is able to subsume the realizations defined in the local repositories of particular agents as well as in the global repository of whole MAS, it is also possible to apply some restrictions to the scope of validity of separate processes, it means that the process, let us say the realization, can be defined as private, public and so on. This well-founded set of process realizations will be used in the next step of the reconfiguration algorithm – selection of the most suitable realization.

The main advantage of this approach is in the clear selection of realizations, which are usable for the execution of a given process, from which can be chosen just the most suitable one. On the other hand, this approach allows only choosing of the most suitable realization from a set of realizations that is formed in the system modeling phase already. The creation of such set is affected by the skills and knowledge of the model author – the Human Factor acts an important role in this process. This process approach is a less flexible at the expense of almost zero error rate during the creation of a set of realizations, in comparison with a logic approach. In other words, the more effective realization leading to the objectives of a given process which is not connected with such process during the modeling phase may exists. Nevertheless, it is impossible to choose the realization that indeed leads to the process objectives but the execution of this realization is mismatched, unusable or illegal.

In the case of the *Logic Approach* (figure 4 B), the first step of reconfiguration algorithm is changed. The finding of a set of suitable realizations arises from the precondition that no connection between the process and realizations is there and each process and realization has defined its output objects and its objectives in accurate and clear form. The challenge of logic tool is to find (local or global repository can be taken into account with a respect to some restrictions or rules) a set of realizations for the second step of the reconfiguration algorithm. The description, properties, objects and objectives of these realizations have to be in accordance with objectives and objects of a given process. This finding process is automatically executed within the Agent in the moment of necessity during the life of this Agent.

The logic approach when compared with process one is more flexible thanks to the fact that all relations between process and realizations are constructed automatically in the real-time in the moment of request. Unfortunately, there is relatively hi-risk of mismatched, unusable or illegal realizations selection. This problem occurs when the descriptions of processes or realizations are incorrect or incomplete. These descriptions of objects, objectives or properties are made by a human, thus the quality of descriptions and efficiency of selection process are highly affected by the author's skills, knowledge and fantasy. In this case, the Human Factor is much more crucial to success than in the process approach. Generally, thanks to this approach it is able to find some secret and effective realizations leading to

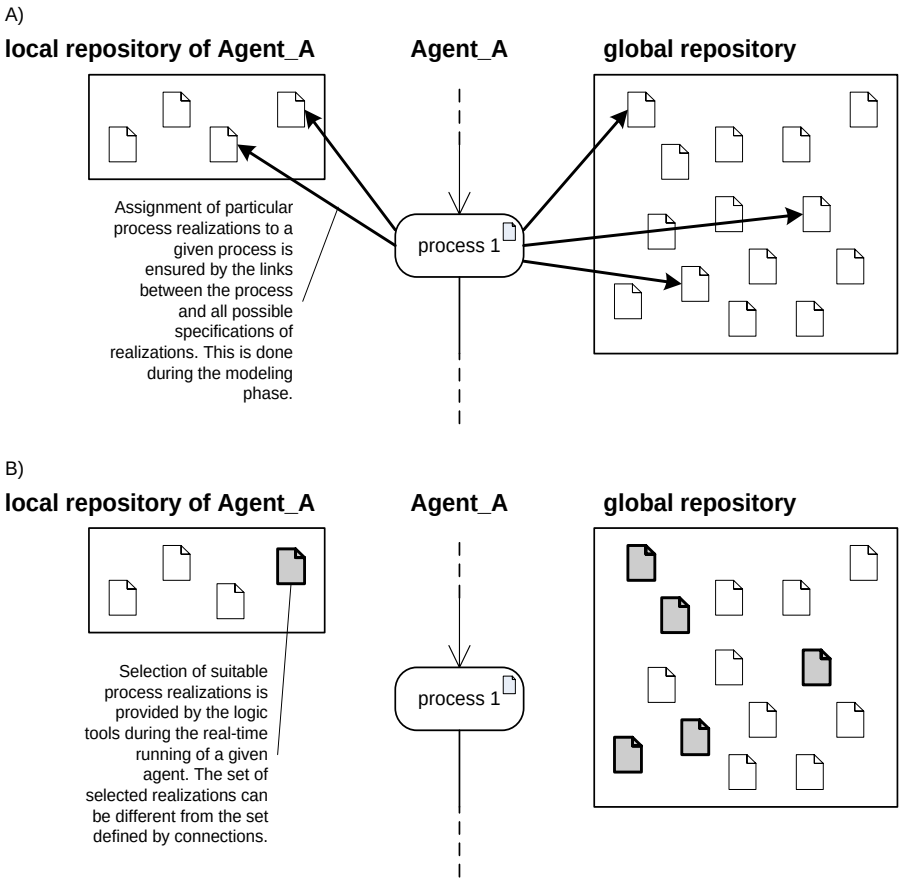

Figure 4: Illustration of two approaches for suitable selection of process realizations

the objectives of a given process. Due to the high demands on the quality and complexity of elements descriptions, the realizations that fit into the descriptions and that will be associated as members of searched set do not need to be usable for the execution of a given process. So, the descriptions of elements in the MAS have to agree with the real world that is expressed by such MAS – and this is a big challenge.

Application Domain

The method that is mentioned in this paper, and that is based on the internal Agent's behaviors, is a headstone for semi-automatic development of Multi-Agent Systems. Thanks to this technique and some developed tool, it will be possible to specify particular components of the system, their properties, skills, features as well as their behaviors. This specification covers the initial phases of software process. The graphical language (extended UML Activity Diagrams) with some formalization and also with some additional textual information (internal activities specifications, objects specifications, etc.) are used for this purpose. This graphical solution could be a good way to move the development process of Multi-Agent System closer to the customers and analysts, it means that the emphasis on the initial phases of software process is there. The next phases of development process should be simplify and automatized, in accordance to the quality, correctness and sophistication of the model that is specified by this technique. Of course, it will be still necessary to implement some parts of agents and system, but the amount and severity of these parts should be minimalized.

The other direction of progress and usage of this technique is focused on the real-time dynamic of Agents' behaviors during the working of a given MAS. Each process has a set of realizations which describe the possible algorithms of the process execution. Because, the number of realizations that belong to a given process can be greater than one, it is necessary to choose just one realization in the time of the process firing – it is a reconfiguration approach mentioned afore. The selection of the applied realization is based on many factors (input and output objects of process and its realizations, costs of the realization execution, etc.) and it is executed just in the moment of necessity during the Agent's life. These Agents, that can change their behaviors in according to the instantaneous situation

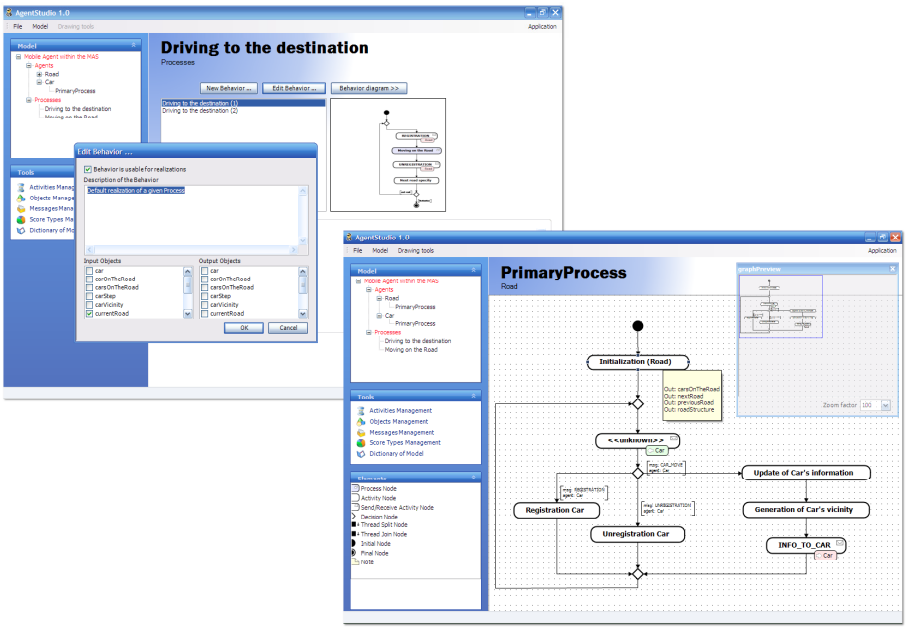

Figure 5: Some screenshots of the AgentStudio software

and conditions, are formed on this reconfiguration idea, they are called "Intelligent Agents" in the sence of this research. It extends the application domain by an ability to build the information system with the Agents which are able to solve the problems separately (based on the available realizations of processes which form the behavior of a given Agent) and in the moment when they occur.

Each software application whose internal structure corresponds to the multi-agent technology is a subject of the application domain of the mentioned approach or research. The application domain is focused on the whole MAS software process and brings the automatization into the design or implementation phases of it. The software applications based on this approach can be also "intelligent", thanks to the usage of Intelligent Agents and the reconfiguration of their internal behaviors. The knowledge redistribution, its sharing and learning of the Agents are possible to realize within these systems too.

Future Work

Although the research of the mentioned technique is in the initial phases, the basic ideas and some concrete conclusions and results concerned with the problematic of Intelligent Agents and the behaviors modeling are done and published at present. The scientific and detailed specifications of all parts of MAS modeling area mentioned in this paper are necessary to do now. It is necessary to define subtle rules, properties and facilities of all elements as well as the relationships between them and whole software process. For instance, many questions concerned with this research are still open (how to specify input and output objects, if the specification of atomic activity should be defined globally or not, which logic tool is suitable for the purposes of this problem domain, which approaches should be use, how to move the model to other phases of software process, etc.) The development of software application that will provide methodology and application framework for the modeling, controlling and operating of MAS system that is based on the mentioned approaches, as well as its components, is an important task. The work on this software application is now in progress. The figure 5 gives a preliminary preview of this "**Agent**Studio" application.

Finally, the goal of this research is to specify complex techniques for modeling of the agents within the MAS based on process modeling. Thanks to this future work, it will be also possible to verify and simulate agents as well as map the descriptions of agents onto the source codes and onto the real implementations of them. The verification, simulation and other models manipulation can be realized by existing algorithms and tools, thanks to the planned support of the process description based on the standard techniques (Petri Nets, BPML, BPEL, XMI, etc.)(Aalst 1997; Matjaz 2006) Above mentioned approaches and techniques also offer the possibilities of distribution of knowledge and learning of the agents.

Conclusion

This paper is engaged with the MAS technology that combines many computer, natural and social science

areas and poses specific claims on the standard software process. The future and power of this technology are, among others, dependent on the facilities and quality of the tools for development process of such systems. These tools must be designed with a respect to the skills of normal users that will be a "modelers" of the MAS and that will determine the objectives, requirements and behavior of whole MAS from the real-world point of view. The fundamental ideas and methods to model and develop MAS, based on the internal agent behavior and intelligence of particular agents are described in this paper. The mentioned problems and solutions are the results of preliminary research which will be elaborated with respect to the future work. The modeling and intelligence approaches, graphical nodes, some other solutions, etc. were discussed along with a preview of future work. The application domain, future possibilities and advantages of MAS development based on these ideas was described too.

This research has been supported by the program "Information Society" of the Czech Academy of Sciences, project No. 1ET101940420 "Logic and Artificial Intelligence for multi-agent systems"

References

Aalst, W.M.P. van der. 1997. "Verification of Workflow Nets". In *Application and Theory of Petri Nets 1997*, P. Azema and G. Balbo (Eds.), Springer-Verlag, Berlin, 407-426

Kubik, A. 2004. *"Intelligent Agents"*, Computer Press, Prague, (publication in czech language).

Radecky, M. and Vondrak, I. 2005. "Formalization of Business Process Modeling". *ISIM 2005*, Hradec nad Moravici.

Radecky, M. and Vondrak, I. 2005. "Modeling of Agents Behavior within MAS". *Information systems in practice 2005*, MARQ, Ostrava, (paper in Czech).

Radecky, M. and Vondrak, I. 2005. "Modeling of Processes". *WOFEX 2005*, Ostrava.

Vondrak, I. 2004 *"Methods of Business Modeling"*, VSB-TUO, Ostrava, (lecture notes in Czech).

Matjaz, B. J. 2006 *"Business Process Execution Language for Web Services BPEL and BPEL4WS 2nd Edition "*, Packt Publishing, Birmingham.

Pilone, D. and Pitman N. 2004. *"UML 2.0 in a Nutshell"*, O'Reilly Media, Sebastopol.