Multi-Agent System and Traffic Simulation

Michal Radecký and Petr Gajdoš
Department of Computer Science
VŠB - Technical University of Ostrava
17. listopadu 15, Ostrava
Czech Republic
E-mail: Michal.Radecky@vsb.cz

Abstract

*MAS development process has similar features to standard Information Systems. However, there are special features that have to be taken into account, in particular: agent based architecture, agents' autonomy and communication, etc. According to mentioned points, MAS development process should be extended or changed. The process development can be handled and documented by the standard UML tool. The output of such development process is MAS with automatic or semi-automatic generated intelligent agents that behave according to defined MAS model. This paper describes a new development and simulation tool called **AgentStudio**. The ideas for intelligent agent generation are applied in the area of traffic simulation.*

1. Introduction

A Multi Agent System (MAS) attracts attention as an approach to complexity systems in recent years. Many MAS frameworks (e.g. JADE, ZEUS, SWARM, etc.) were proposed to help developers to build complex heterogeneous systems. Fundamental element of MAS is represented by agent. Groups of co-operative intelligent agents make MAS system that is flexible and robust at once. MAS is dynamic - its components are not known in advance and can be created or removed from the system. In the case of MAS, we usually speak about distributed systems. It means that agents can exist within different software and hardware platforms and communicate through a communication protocol. These features allow us to use MAS for traffic simulation. This paper will describe new software called ***AgentStudio*** that consists of two main parts. The first one is represented by an ***AgentStudio*** Designer. The second part covers a simulation application designed purely for traffic simulations now. It is called ***AgentStudio*** Simulator. Both applications belong to a wider research (see the note below) dealing with intelligent agents based on Transparent Intensional Logic (TIL).

1.1. Our research background

First, it should be fine to briefly introduce our research domain with emphasis on fundamental request and possibilities. Our research is concerned with Intelligent Agents and their development process. We would like to define methodology to specify, model, implement, simulation and deployment of the agents.

Nowadays, there are many approaches to develop standard information systems from requirement specification to deployment (UML, RUP, etc.). The situation in the area of MAS is not so clear. Of course, it is able to use the standard methodologies for MAS develop support, but it is not ideal way to build complex multi agent system. The specific features of these systems (autonomy of element, communication among elements, brain functions, etc.) require some special tools and approaches. The request specification and modeling are very important phases of system development. The Agent UML and new features of UML 2.1 is able to cover this phases with support of MAS specialty. But, the AgentUML is a quite old approach that is no longer supported. On the other hand, the UML standards are too complicated for normal users and they bring many functions and possibilities which are not necessary for MAS developing.

So, we would like to introduce our research that is focused on the simple, clear and semi-automatic MAS development process. Our approach covers the modeling phase where the results of this phase should be usable for next phases of whole development process, especially development process of Intelligent Agents. This modeling approach extends the standard UML Activity Diagram by elements that cover special features of Agents (as autonomous software components). Based on our model, we are able to build Intelligent Agents with capability of behavior reconfiguration during their lives. Thanks to the formal grounds and intention on agents construction elements (messages, behaviors, etc.) we would like to support the semi-automatic code generation as a part of

agents implementation phase. Our approach also relies on "brain facilities" and complex logical potential.

From implementation point of view, there exist several MAS frameworks at implementation level that make environments for such systems. They are usually based on standards that make it widely used without any platform dependency. FIPA (The Foundation for Intelligent Physical Agents) is an IEEE Computer Society standards organization that promotes agent-based technology and interoperability of its standards with other technologies [2]. For our purpose, JADE (Java Agent Development framework) has been chosen. It is a software framework fully implemented in Java language; however, it has been rewritten into Microsoft .NET J# and C#, respectively. JADE [3,4] is a middle-ware that simplifies the development of applications. Several companies are already using it for very different application sectors including a supply chain management, rescue management, fleet management, auctions, tourism, etc. This type of distributed applications enabled by JADE, in particular, when applied to the mobile environment, ignite a new trend of the software development: the software is equipped with autonomy, intelligence, and capability of collaboration and the quality of the system is given by its capabilities of the devices and by their mutual interaction and collaboration.

2. Intelligent agent and its development

As it was mentioned above, the agents make basic elements of each multi-agent system. The added value of our research also consists in an application of certain intelligence features to the agents. Then we will speak of intelligent agents and intelligent multi-agent systems.

Agents are taken as software components with internal behaviors formed by processes in our approach. Internal processes of agents, their structures, sequences, and applications result from current situation and agents' states. Generally, a simple process agent is regarded as reactive or proactive according to mentioned FIPA standards. Nevertheless, process specification [5] enables agent realization with respect to autonomous determination and behavior changes. It can be said that the internal behaviors of intelligent agents are defined by clearly specified processes. These process specifications and internal behavior result from modeling phase of MAS development.

2.1. AgentStudio Designer

The ***Agent****Studio* Designer has been developed for purposes of process flows specifications, modeling and their transformation. It provides entire processes specification thanks to Agent Behavior Diagrams (ABD). ABD covers standard technique of UML Activity Diagrams extended by new structural elements, executive elements and rules [6]. This tool has several outputs including documentation and graphic representation of internal agents' behaviors (Traffic MAS architecture chapter). An implementation of modeled agents based on the semi-automatic source codes generation will be the prime result of our research because it is not complete yet. These source codes link automatic generated skeletons (classes, methods, objects, architecture, etc.) and particular implementation of atomic activities. Of course, more complex codes have to be completed by programmers and adapted for a given MAS framework (e.g. JADE).

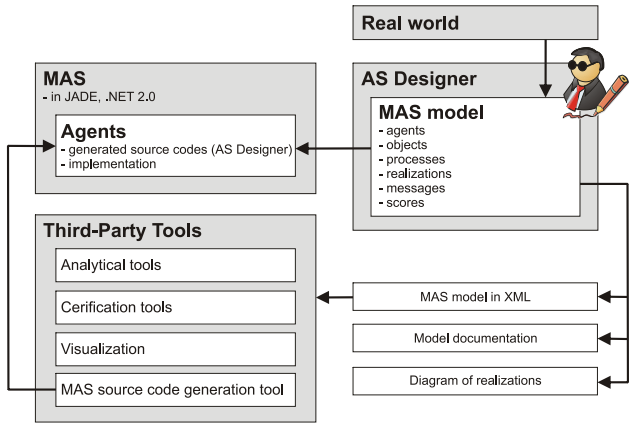

Figure 1: ***Agent****Studio* Designer schema

3. Behavior reconfiguration

The reconfiguration approach represents the way to implement intelligent agents formed on internal processes. [7] The idea of behavior reconfiguration comes from the hypothesis that each process (reconfiguration point), which is fired within agent's life, can be realized by different ways and techniques. These realizations depend upon knowledge, experiences, environment and states of all agents. A reconfiguration algorithm is applied in time of process firing. Each process requires some input objects and can produce output objects. The same holds for process realizations. Each process is described by a set of realizations. Each realization is described by single ABD during the specification phase. ABDs can be stored within agent internal knowledge base or global MAS repository. Moreover, the agents are able to extend their own sets of realizations thanks to communication and cooperation with other agents and/or platform facilities. This feature enables agents to learn from others and share knowledge bases. Process realizations could be deliberated also in real-time without previous ABD specification but this is not an aim of our research.

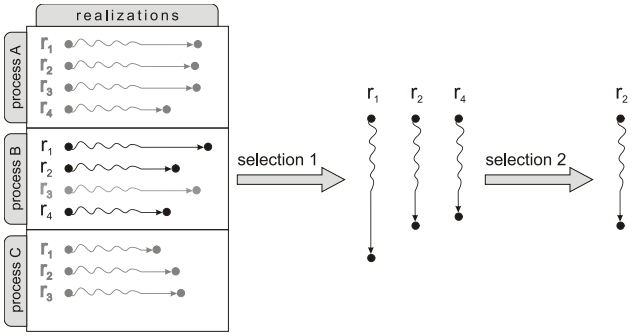

Figure 2: Reconfiguration process

The Figure 2 shows the basic scheme of mentioned reconfiguration method. At the beginning, the set of all processes and their realizations is defined. Next, the selection phase is initiated. Depicted selection consists of two steps. The first one represents a simple selection of applicable realizations, based on input objects occurrences. The second one chooses the most suitable realization according to input objects properties, scores, etc. Methods of multicriterial analysis or logical tools can be used during the selection phase.

4. Traffic simulation

The relationship between our MAS model and subsequent simulation is described in this part. We should point out that this relationship is not fully implemented yet. Currently, there doesn't exist program providing data transformation from ***Agent****Studio* Designer to ***Agent****Studio* Simulator. The work is still in progress.

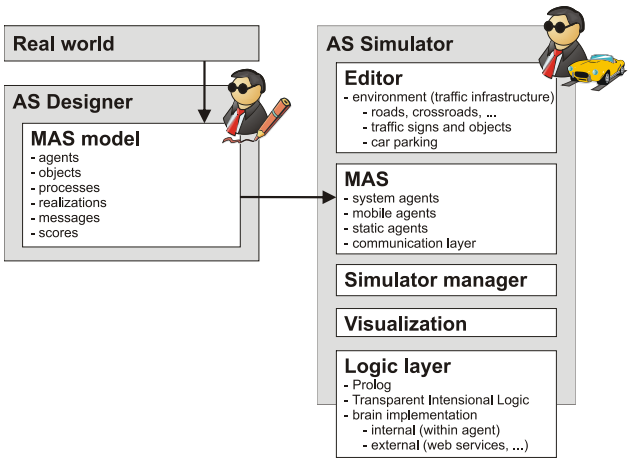

Figure 3: AgentStudio schema

4.1. Reasons for traffic simulations

Proposed approaches and methods have been proven in the area of traffic management. Now, the reasons leading us to MAS application are explained. First, computational simulations are becoming increasingly important because in some cases it is the only way that processes can be studied and interpreted. These simulations may require very large computational power and the calculations have to be distributed on several computers. The MAS technology supports such kind of computation because of its independence from platforms, operation systems, etc. However, we do not want to simulate all possible situations within particular traffic system. We take into account just a set of chosen traffic situations to demonstrate the power of MAS technology, logic and ***Agent****Studio*. Next, several commercial systems pick up actual traffic data, create digital models of traffic infrastructure (roads, crossroads, traffic signs, etc.) and provide such data sets to use them within other projects. Then it is quite easy to use provided data in simplified form for ***Agent****Studio* Simulator and test agents' behaviors on real data. Logic and intelligent decision-making process play an important part in our project. Also this point is inherent with traffic simulations, e.g. if a traffic light is red, a car should stop before a crossroad. In other words, the car has to change its behavior. Most of such rules are well described in Highway Code and can be rewritten into Prolog and/or TIL (Transparent Intensional Logic) formulas. The fourth reason consists in agent behavior description based on UML modeling which is covered by ABD. The last reason is an eye appealing way of presentation of simulation results. Visualization tool makes one part of ***Agent****Studio* Simulator and helps us to see what the agents really do and how they behave depending on simulated environment.

4.2. Target area description

Traffic simulations try to reflect real situations taking place on roads. Nowadays, ***Agent****Studio* Simulator allows us to design and edit simplified infrastructure to test agents' behaviors. In the future, it will also enable to import real GIS data. These are important situations which we focused on:

- Cars overtake each other and they will recognize traffic obstacles.
- They safety pass through crossroads.
- They keep safety distance from other agents (cars).
- They keep basic rules defined in Highway Code.

Previously mentioned situations make basic elements within agent behavior design process. Particular situation is solved during agents' life with respect to its ability to make a decision. Finally, MAS development and simulation can be divided into steps mentioned below.

4.3. Process of modeling and simulating in the AgentStudio

A complete scenario of ***AgentStudio*** utilization can be divided into several parts:

1. Real world requirement analysis → identification of the agents and their goals, objects, processes, etc.
2. Agents' behaviors modeling in ***AgentStudio*** Designer → MAS model based on ABDs.
3. Source code generation and its completion (Complex behaviors have to be modified by programmers.) → source code in specific programming language and selected MAS framework.
4. MAS environment specification → traffic infrastructure description based on retrieved real data.
5. Initial phase of simulation process → starting MAS platforms, encapsulation of traffic information into platform data structures.
6. Simulation, visualization and management.

The described process and its steps can be applied to various implementation areas. Generally, final simulation and implementation respect selected MAS framework and system architecture. Proposed architecture convenient to traffic simulation is described below.

4.4. Traffic MAS architecture

The next figure illustrates our MAS architecture.

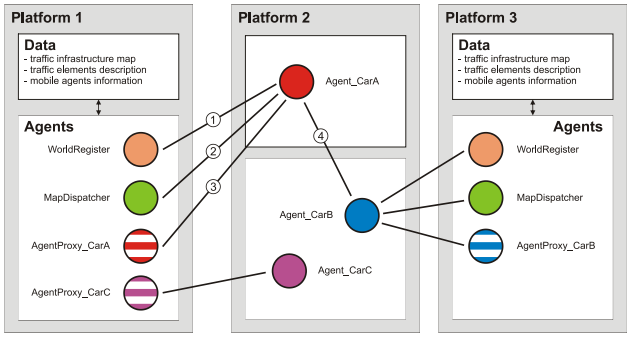

Figure 4: MAS architecture for traffic simulation

Platforms 1 and 3 represent two parts of the real world, e.g. a town district (P1) with a separated parking lot (P3). The data of such platform consists of traffic infrastructure map of a given area. Next, it has a description of traffic elements located on traffic infrastructure. Finally, the data holds mobile agents information obtained from proxy agents. Generally, particular platform data reflects the state of the real world. Second parts of these platforms make environments for system agents which are responsible for the communication with other agents (Proxy agents), map services (MapDispatcher agent) and for agent registration (WorldRegister agent). The platform 2 consists of mobile agents that represent cars moving in the real world. The platform 2 can be distributed on many hardware nodes according to FIPA standard.

The main relationships between platforms/agents are described in the following points (see the Figure 4).

1. A single car (Agent_CarA) registers itself into a given part of the world (P1). It is done through the communication with WorldRegister agent. This agent also creates a proxy agent for Agent_CarA (st. like proxy in Object Oriented Programming).
2. This connection provides an access to map services (road finding, infrastructure description, etc.) ensured by MapDispatcher agent (see the Yellow Pages in [2]).
3. This communication realizes a synchronization of mobile agent data. It runs during whole car agent's life.
4. Mobile agents can communicate between each other to negotiate emergency situations and/or to get some new knowledge, addition information on surrounding world, etc.

5. Car as an intelligent agent

Previously mentioned approach of intelligent behavior is applied within every car agent. Moreover, a Car agent needs certain form of inner structure for intelligent determination. The basic features of an intelligent mobile agent (Car) are perception, decision making and acting.

5.1. Perception

It is a natural feature of each live organism and the same holds for the intelligent agent. In the case of agent, the perceptions are based on technical facilities. Information sources are:

- sensors
- GIS data
- communication with other agents

Implementation and usage of a Car agent in the real operation need the hardware sensors, e.g. digital cameras, ultrasonic detectors, GPS, etc. Nevertheless, this research deals with software simulated perceptions. According to the mentioned architecture, the Proxy agents make such kind of sensors. These agents have a full access to platform information and can simulate the sight and location of substituted agents. Provided data is sent by Proxy agent to its Car agent via ACL message.

Spatial data represents the most important information source for mobile agent. For the simulation, it is better to appear from real data on traffic infrastructure which can be obtained from some Geographical Information Systems (GIS). The mission of GIS is not to provide detailed description of infrastructure. It consists in map

services ensuring road finding, traffic signs positioning, traffic element description, etc.

The last information source appears from agents' communication. This way of information retrieval meets the principles of MAS. Agents can interchange some knowledge (traffic jam, accident location, etc.) or use services (parking payment, call for help, etc.).

5.2. Decision making

A rational agent in a multi-agent world is able to reason about the world (what holds true and what does not), about its own cognitive state, and about that of other agents [8]. A theory formalizing reasoning of autonomous intelligent agents has thus to be able to 'talk about' and quantify over the objects of agents' attitudes, iterate attitudes of distinct agents, express self-referential statements and respect different inferential abilities of agents. Since agents have to communicate, react to particular events in the outer world, learn by experience and be less or more intelligent, a powerful logical tool is of a critical importance.

To this end we make use of Prolog and Transparent Intensional Logic [9, 10]. The logic tool is represented by an agent brain; currently, an external software component used as a remote service. The communication with the brain is based on request/response approach, where the request has to contain relevant information on agent state, intention, it's seeing, nearest surround and other knowledge. The response of this service should determine next step of agent life process with respect to request content. Next, the intelligent decision making is used within reconfiguration process, which was already mentioned.

5.3. Acting

The last one is a natural consequence of two previous features. Acting means the concrete behavior firing which leads to pass agent's objectives. At the end of this, agent's state, as well as the state of whole MAS, has to be updated. Then, the process compound of perception, decision making, and acting is repeated.

6. Illustrative example

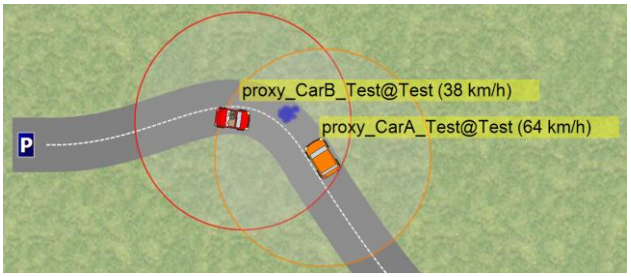

Figure 5: AgentStudio Simulator snapshot – illustrative traffic situation

The Figure 5 illustrates a common traffic situation. There are two cars on a simple road. They are on different lanes, however they have to solve conflict situation because of water puddle on the road. The CarB can continue to drive but the CarA has to change its behavior. The solution of the behavior change is described below.

6.1. Perception of the CarA

The agent sees everything in the circle bounded by the orange color. This perception is realized by environment, particularly by Proxy Agent of a given Car Agent. This is perceptions of CarA agent based on the situation depicted on figure 5:

Infrastructure

- road element composed of two lanes

Objects

- CarB driving on the contra-flow lane in distance 150 m with the speed 38 km/h.
- water puddle placed on the current lane in distance 105 m.

Such information is described by XML in ***Agent****Studio* Simulator and it is used within decision making process.

6.2. Car Agent skills – processes and realizations

Each Car agent has a set of processes which form its internal behaviors. Every process can be implemented by several ways – realizations. The following list illustrates the current state of car agents' internal repositories.

Process: GO -> realizations: 1
Process: SLOW_DOWN -> realizations: 1
Process: SPEED_UP -> realizations: 1
Process: CHANGE_LANE -> realizations: 2 (based on direction of change – left, right)

6.3. Decision making of the CarA

Agent CarA holds some information on its state, e.g. current speed, orientation, intention, physical properties (car weight, maximal speed and acceleration). Such facts together with perceptions (and/or traffic rules) form input data for logic resulting. In this case, the logic consequent of given premises is a decision to SLOW_DOWN because there is no possibility to safety drive around the obstacle. The appropriate process realization (SLOW_DOWN_1) is fired. In the case when there is not car in other lane (CarB), the decision result should be the process CHANGE_LANE. The selection of realization to fire is based on reconfiguration algorithm (CHANGE_LANE_left in this special case).

6.4. Acting of the CarA

The reconfiguration principles have been applied. The CarA changes its behavior according to SLOW_DOWN process specified by ABD. The CarA decreases its speed and tries to solve the situation again with new values and conditions.

7. Conclusion & Future work

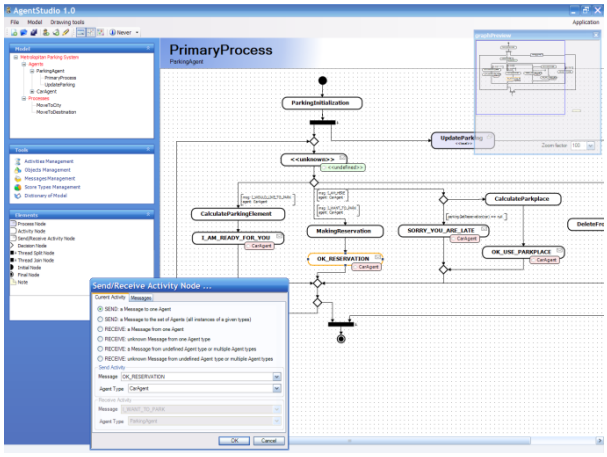

Figure 6: AgentStudio Designer screenshot

In this paper we briefly described our research on Multi-Agent Systems and its applications. The work is still in progress and there is a lot to be done. More attention has to be paid to imperfect and vague information. To this end we use Prolog. The inference machine based on TIL has to be precisely specified and implemented. In the area of process management we are going to develop behavior-model extensions in order to provide message specifications and interfaces, automatically generate new generations of agents that learn by experience, and to distribute behavior schemes and knowledge among the whole system. Next, the source code generator should be improved. Our current application

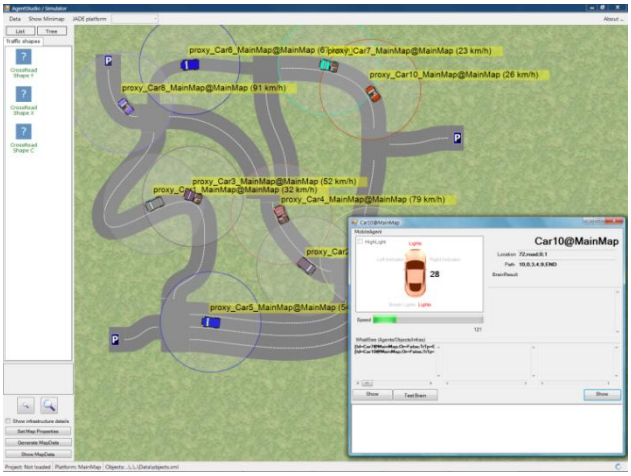

Figure 7: ***Agent****Studio* Simulator screenshot

is a model of a traffic system. The main work will consist in next development and extension of ***Agent****Studio*.

Acknowledgement

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"

8. References

[1] Namee, B. M. *A Proposal for an Agent Architecture for Proactive Persistent Non Player Characters.* 2001.

[2] Odell, J. The Foundation for Intelligent Physical Agents. *http://www.fipa.org.* [Online] 2006.

[3] F. Bellifemine, G. Caire, A. Poggi, and G. Rimassa. JADE - A White Paper. 1999.

[4] Bellifemine, Fabio. Java Agent Development Framework Documentation. *http://jade.tilab.com/.* 2005.

[5] Radecký, Michal and Vondrák, Ivo. Agents and Their Behavior's Reconfiguration. *ECEC 2006.* Athens : EUROSIS, 2006. ISBN 90-77381-24-4.

[6] Radecký, Michal. Intelligent Selection of Realizations. *ECMS 2006.* Bonn : s.n., 2006. ISBN 0-9553018-1-5.

[7] Radecký, Michal and Gajdoš, Petr. Process and Logic Approaches in the Inelligent Agents Behavior.

Information Modelling and Knowledge Bases XVIII. Amsterdam : IOS Press Amsterdam, 2007. ISBN 978-1-58603-710-9.

[8] Schelfthout, K. and Holvoet, T. An Environment for Situated Multi Agent Systems. *International Conference on Autonomous Agents and Multi Agent Systems, AAMAS.* Washington DC : IEEE Computer Society, 2004.

[9] Tichý, Pavel. The Foundations of Frege's Logic. *De Gruyter.* 1988.

[10] Duží, Marie. Concepts, Language and Ontologies (from the logical point of view). *Information Modelling and Knowledge Bases XV.* Amsterdam : IOS Press Amsterdam, 2004.