Proceedings of the 11th International Conference "Reliability and Statistics in Transportation and Communication" (RelStat'11), 19–22 October 2011, Riga, Latvia, p. 374-381. ISBN 978-9984-818-46-7
Transport and Telecommunication Institute, Lomonosova 1, LV-1019, Riga, Latvia

MULTIAGENT APPROACH TO DISCRETE TRANSPORT SYSTEM MODELLING

Katarzyna Michalska, Jacek Mazurkiewicz

Institute of Computer Engineering, Control and Robotics
Wroclaw University of Technology
ul. Janiszewskiego 11/17, 50-372 Wroclaw, Poland
E-mails: Katarzyna.Michalska@pwr.wroc.pl, Jacek.Mazurkiewicz@pwr.wroc.pl

The paper describes a novel approach to monitoring and modelling Discrete Transport System (*DTS*). We propose the formal model of the transport system with the approach to its modelling based on the system behaviour observation. The analysed *DTS* is a simplified case of the Polish Post. Using a multilevel-agent based architecture the realistic data are collected and presented in one common language that is an authors' solution. Described architecture can be finding as an idea of a tool that can visualize and analyse data, with respect to real parameters. No restriction on the system structure and on a kind of distribution describing the system functional and reliability parameters is the main advantage of the approach. The proposed solution seems to be essential for the owner and administrator of the transportation systems.

Keywords: discrete transport system, reliability, agent management system

1. Introduction

The temporary computer engineering still does not define an "agent" term in detailed way, but it is not a real barrier to establish the unified semantic meaning of the word in technical point of view. The agent can play the role of the autonomous entity [4] as a model or software component for example. The agent's behaviour can be noticed as trivial reactions, but is not limited – so we can easily find agents characterised by complex adaptive intelligence. Sometimes is important to point the potential adaptive abilities of the agents [7]. It means the agent can gather the knowledge from the environment around and to tune their behaviour as a reaction for different events.

This way we can say the agents belong to the softcomputing world. The agent's structure is not obligatory plain. We can easily [2] find at least two levels (lower, higher) of the rules created for the agents. This approach allows to tune the level of the sensitivity for the environment and to define the vitality feature of the agent understood as activity or passivity [9, 10].

The presented work uses the agents in task of the transportation system monitoring and modelling, so we propose the following description of the most important agent's features [13]:

- unique identification within the proposed architecture,
- interaction abilities and proper interfaces for communication and different data transfer,
- secure protocols necessary for communication purposes,
- hardware and / or software implementation,
- plug-and-play ability to guarantee promising scalable and flexible structure.

The agent-based approach provides the real great effectiveness comparing with the classical architectures if we think about the data gathering and aggregation from the real sophisticated system characterised by the large network, significant number of nodes and non-trivial addressing aspects. This way it is easy to create the global and detailed enough view for multilevel systems with elements described by various sets of features. We propose to use the agents to create the intelligent hierarchical monitoring architecture (described in Section 3) for the discrete transport system (*DTS*) defined in Section 2.

The section 4 presents an authors' solution of a description language for a proposed model, called *DTSML* (Discrete Transport System Modelling Language). As a format of the proposed language *XML* (Extensible Markup Language) was chosen. Main reason is a simple (easy to learn) and readable structure, that can be easily convert to text or other format.

Moreover, *XML* is supported not only with various tools (providing validation possibilities) but is also supported by many programming languages and framework in case of quicker and more efficient implementation.

2. Transport Model

2.1. Preface

Depending on the level of detail in modelling the granularity of traffic flow, traffic models are broadly divided into two categories: macroscopic and microscopic models. According to Gartner et al. [5], a macroscopic model describes the traffic flow as a fluid process with aggregate variables, such as flow and density. The state of the system is then simulated using analytical relationships between average variables such as traffic density, traffic volume, and average speed.

On the other hand, a microscopic model reproduces interaction of punctual elements (vehicles, road segments, intersections, etc) in the traffic network [1]. Each vehicle in the system is emulated according to its individual characteristics (length, speed, acceleration, etc.) [3]. Traffic is then simulated, using processing logic and models describing vehicle driving behaviour, such as car-following and lane-changing models. Those models reproduce driver-driver and driver-road interactions [8].

Despite its great accuracy level, for many years this highly detailed modelling was considered a computationally intensive approach. Since the last twenty years, with the improvements in processing speed, this microscopic approach becomes more attractive [12].

2.2. System description

The analysed transportation system is a simplified case of the Polish Post. The business service [14], [6], [17] provided the Polish Post is the delivery of mails. The system consists of a set of nodes placed in different geographical locations. We have the headquarter (*HQ*) located in the central part of Poland and two kinds of nodes could be distinguished: central nodes (*CN*) and ordinary nodes (*ON*).

There are bi-directional routes between nodes. Mails are distributed among ordinary nodes by trucks, whereas between central nodes by trucks, railway or by plane. The mail distribution could be understood by tracing the delivery of some mail from point *A* to point *B*.

At first the mail is transported to the nearest ordinary node *A*. Different mails are collected in ordinary nodes, packed in larger units called containers and then transported by trucks scheduled according to management architecture decision to the nearest central node. In central node containers are repacked and delivered to appropriate (according to delivery address of each mail) central node. In the second – closest to the destination place – central node the mail is again repacked and delivered in a container to destination ordinary node.

The headquarter collects all data about the actual situation in whole transportation system and makes the necessary decisions as the reaction for the temporary needs. The headquarter is not in use in transportation action – if we think about the loading, unloading processes, etc. The central nodes aggregate the data from the single region of the country. And finally the ordinary nodes control the local situation to the end-user. The scale of necessary actions depends on the actual needs.

In the Polish Post there are 14 central nodes and more than 300 ordinary nodes. There more than one million mails going through one central node within 24 hours. It gives a very large system to be modelled and simulated.

The process of any system modelling requires defining the level of details. Increasing the system details causes the simulation becoming useless due to the computational complexity and a large number of required parameter values to be given.

2.3. Model

We can model discrete transport system described above as a 4-tuple [15]:

$$DTS = \langle Client, BS, TI, MA \rangle \tag{1}$$

where
Client – client model, *BS* – business service, a finite set of service components, *TI* – technical infrastructure, *MA* – monitoring architecture.
The technical infrastructure of *DTS* could be described by three elements:

$$TI = \langle No, V, MM \rangle, \tag{2}$$

where
No – set of nodes; *V* – set of vehicles; *MM* – maintenance model.

Set of nodes (*No*) consists of single headquarter (*HQ*), central nodes (*CR*), a given number of ordinary nodes (*ON*). The distance between each two nodes is defined by the function:

$$distance : No \times No \to R_+ . \tag{3}$$

Each node has one functional parameter the mean (normal distribution) time of loading a vehicle:

$$loading : No \to R_+ . \tag{4}$$

Moreover, the central node (*CR*) has additional functional parameter: number of service points (in each ordinary node there is only one service point):

$$servicepoints : CN \to N_+ . \tag{5}$$

Each vehicle is described by following functional and reliability parameters [16]:

- mean speed of a journey
  $$meanspeed : V \to R_+ , \tag{6}$$
- capacity – number of containers which can be loaded
  $$capacity : V \to R_+ , \tag{7}$$
- mean time to failure
  $$MTTF : V \to R_+ , \tag{8}$$
  a time when failure occurs is given by exponential distribution with mean equal to a value of *MTTF* function,
- mean repair time
  $$MRT : V \to R_+ . \tag{9}$$

The traffic is modelled by a random value of vehicle speed and therefore the time of vehicle (v) going from one node (n_1) to the other (n_2) is given by a formula:

$$time(v, n_1, n_2) = \frac{distance(n_1, n_2)}{Normal(meanspeed(v), 0.1 \cdot meanspeed(v))} , \tag{10}$$

where *Normal* denotes a random value with the Gaussian distribution [17].

Maintenance model (*MM)* consists of a set of maintenance crews which are identical and unrecognised. The crews are not combined to any node, are not combined to any route, they operate in the whole system and are described only by the number of them.

The time when a vehicle is repaired is equal to the time of waiting for a free maintains crew (if all crews involved into maintenance procedures) and the time of a vehicle repair which is a random value with the Gaussian distribution: *(Normal(MRT(v), 0,1·MRT(v))*.

Business service [17] (*BS*) is a set of services based on business logic, that can be loaded and repeatedly used for concrete business handling process. Business service can be seen as a set of service components and tasks that are used to provide service in accordance with business logic for this process. Therefore, *BS* is modelled a set of business service components (*sc*),):

$$BS = \{sc_1, \ldots, sc_n\}, n = length(BS) > 0 , \tag{11}$$

the function *length*(*X*) denotes the size of any set or any sequence .*X*

The service realised by the clients of the transport system are sending mails from a source node to a destination one [15]. Client model consist of a set of clients (*C*).
Each client is allocated in one of nodes of the transport system:

$$allocation : C \to No . \tag{12}$$

A client allocated in an ordinary node is generating containers (since, we have decided to monitor containers not separate mails during simulation) according to the Poisson process with destination address set to ordinary nodes. In the central node, there is a set of clients, one for each ordinary node. Each client generates containers by a separate Poisson process and is described by intensity of container generation:

$$intensity : C \to R_+ . \tag{13}$$

3. Monitoring Architecture

In case of Monitoring Architecture representation and distributed multilevel agent-based architecture can be constructed. Figure 1 shows the diversification of complexity of a system into layers and their placement in a system.

The lowest components of the structure are *Node Probes* which are the simplest pieces of the architecture representing resident level (i.e. vehicles). These are the simplest and easy to get data that at this level represent small value that is why they are aggregated in upper units are forwarded to appropriate supervising *Node Sensors*.

Next *Node Sensors* collect the data and create an image of the particular area – so they are located in the ordinary nodes (*ON*). Again the information is sent to a higher level – *Local Agent* – combined to the central nodes (*CN*).

$$NS_i = \bigcup_j NP_j ; j \in N \tag{14}$$

This set of information creates a database building representation of local part of a system (subnetwork). It means that the local view of the system and partial administration in the system can be done at this level.

$$LA_i = \bigcup_j NS_j ; j \in N \tag{15}$$

The highest components of this structure are the *Global Agent* – working in the headquarter (*HQ*), that picks and process local information's and view to one central unit.

$$GA = \bigcup_j LA_j ; j \in N \tag{16}$$

This module stores all information from a whole system. It is situated in one point and one dedicated machine (with a strong backup). Assembling all local view at this level we get one homogenous global view. At this level, data-mining techniques can be used since we know the politics of the Transportation Company as much all information's about parcels.

Figure 2 shows the same tree of hierarchy, where we can see that *Node Probes* can be seen as the lowest point in the system – cars, tracks, etc. Next the nearest or small office can be presented as a *Node Sensors*. Each Post – ordinary node (*ON*) area belongs to a bigger Post Office (mainly located in a big city) – called the central node (*CN*) during our analysis.

Afterwards, all information's and some packages (i.e. in case of international packages) are sent to Central Unit – the headquarter (*HQ*) located in one Central point in the Country.

Looking at the Figure 1 and Figure 2, we can see that set of information flow goes to the central unit – *Global Agent*. For this reasons it is the most complex and the simplicity of the data that are needed to describe the system in this point is the highest in hierarchy.

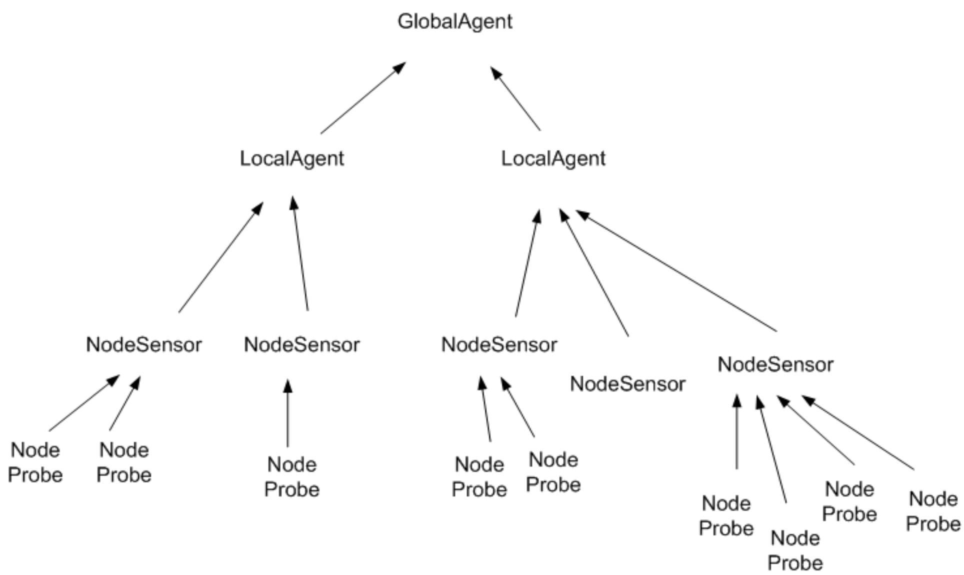

Figure 1. Multilevel architecture schema

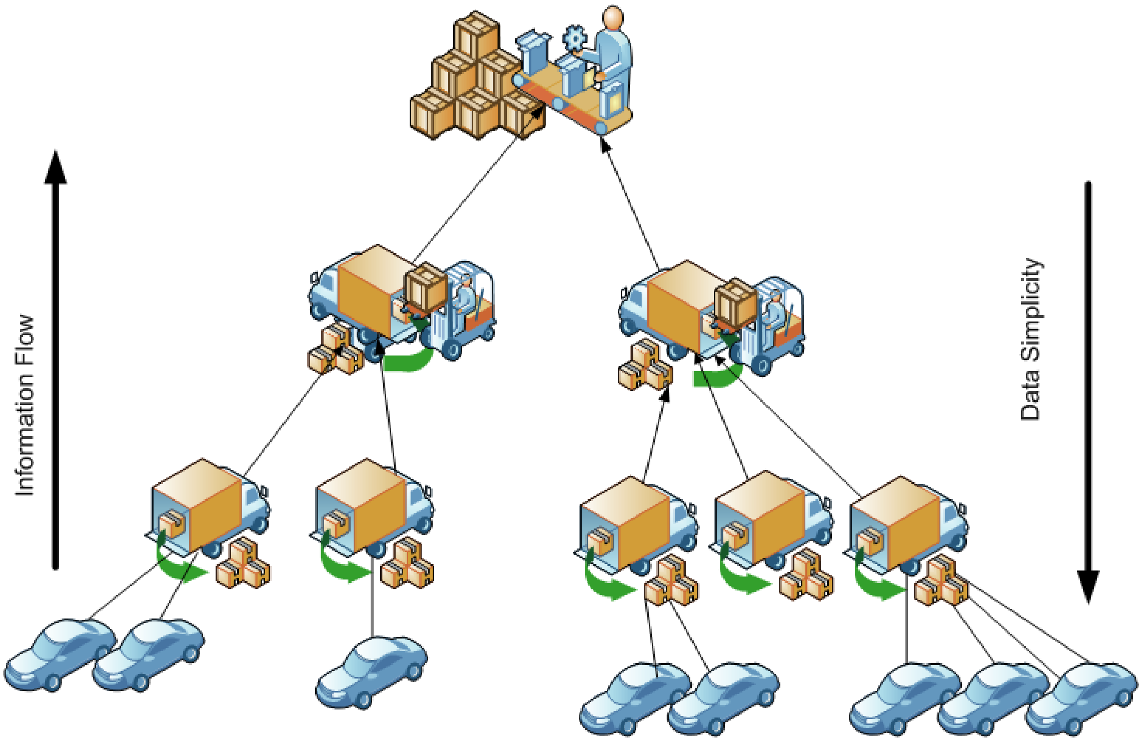

Figure 2. Architecture concept

4. Transport Language

In Section 3 we mentioned that the View of the Transport System can be realised on two levels (local and global). To do that, the tool for visualisation and data processing is needed. Furthermore having this tool we can not only see the topology of the system, but also its elements and parameters.

It gives us an opportunity to see the system more precisely or even make some analysis on a real data that comes from proposed multilevel agent-based architecture.

Still, it requires specify data format that can be shared between tools, but since of the data exchange is done based on *UML* diagrams, there is a need used some other solution that will be more suitable. Since *UML* diagrams are mostly graphical representation of a model, we propose an authors' solution of a *description language* for a proposed (Section 2.3) model, called *DTSML* (Discrete Transport System Modelling Language) [11].

Format of this language is based on *XML* standards, since it is easy to use, and extendable. Moreover the format allows using the language without special tools, since *XML* is supported by many tools. Figure 3 shows a fragment of the language with appropriate elements and attributes related with mathematical model described in Section 2.3.

The proposed language assures aggregation of dependability and functionality aspects of the examined systems. One language describes whole system and provides a universal solution for various techniques of computer analysis as an effective and suitable input for those tools.

Expect easiness of potential softcomputing analysis, promising scalability and portability (between analysis tools) can be named as the main advantage of the language usage.

Proposed language is easy to read and to process using popular and open-source tools; however the metadata used in this format are still a significant problem in case of file processing (large size of the file). Nevertheless since *XML* format is strongly supported by programming languages like: *Java*, *C#*, the usage as much as processing of the file can be done irrespectively from application language.

As previously described (Fig. 1) data send by *Node Probe* are combined in the *Node Sensors*. Each of these entities has assigned to it a supervisor – *Local Agent* that accumulates these files in order to create local view.

This level is more compound and computational complex than previous one considering installed database and some methods that solve additional problems. In this way *XML* files transferred from the simplest level to the next one – creating views on the upper level.

```
<Node>
         <SingleCentralNode to="string1">
          <numberOfPackages>3455</numberOfPackages>
          <numberOfVehicules>8990</numberOfVehicules>
          <ManagementSystem />
          <TechnicalInfastuctureTopology numberofOrdinaryNodes="5819">
                   <timeBetweenSpecificNodes>
                       <linksBetweenNodes from="Wroclaw" to="Opole" />
                        <time>5.7</time>
                    </timeBetweenSpecificNodes>
                <TechnicalInfastuctureTopology>
            </SingleCentralNode>
        </Node>
        <Vehicule>
            <meanspeed>9.7</meanspeed>
            <capacity>678</capacity>
            <MTTF>10.05</MTTF>
            <MRT>50.98</MRT>
        </Vehicle>
```

Figure 3. DTSML – fragment of the language

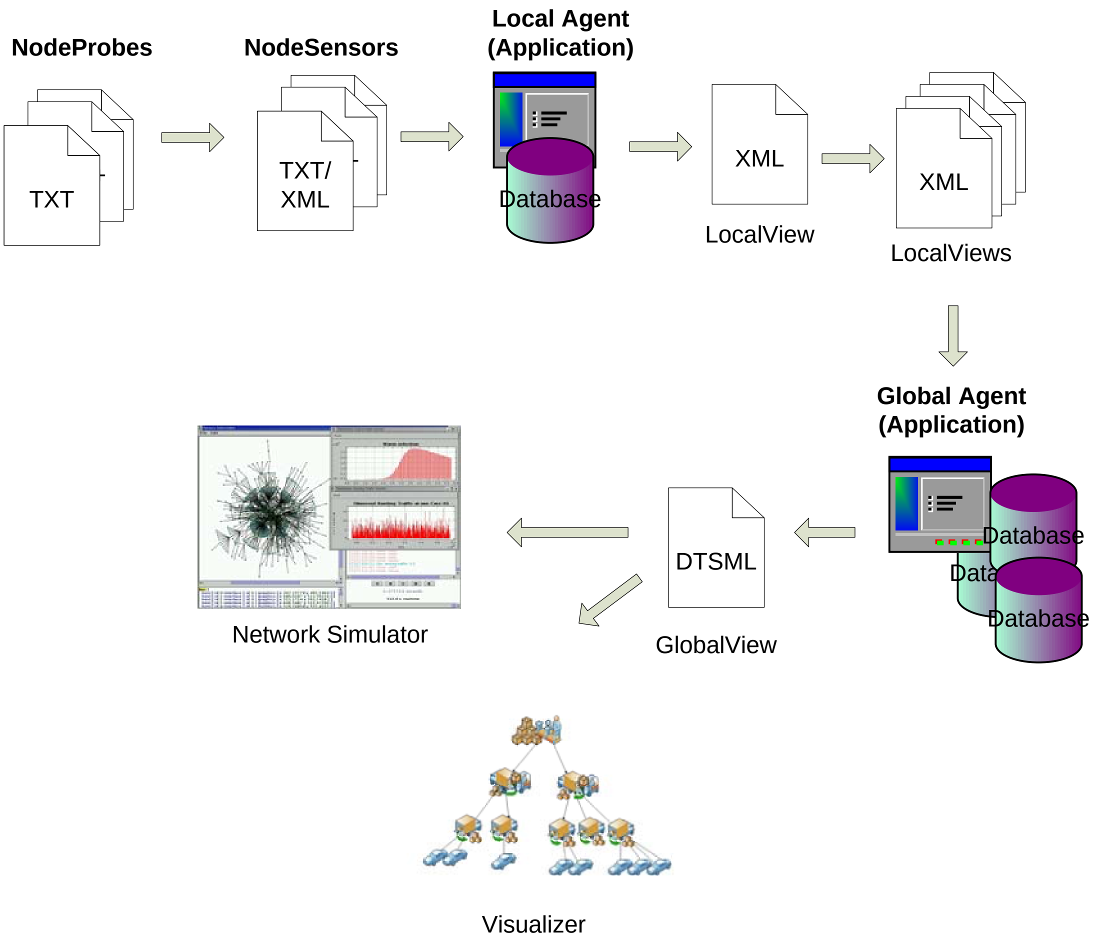

Figure 4. Multi-agent Monitoring Application – architecture concept

Global Agent collects this information and similarly to Local Agent combines all information included in dedicated *DTSML* file. As this Global Agent is the most resourceful entity it may be distributed, so it can contain more than one database. At the end, full description of the system is created, visualised and analysed with respect to dedicated analysis tools.

5. Conclusions

We have presented a formal model of discrete transportation system (*DTS*) including reliability and functional parameters. The *DTS* model is based on Polish Post regional transportation system. Using a multilevel-agent based architecture the realistic data are collected and presented in one common language that is an authors' solution.

Moreover the format of the proposed language is fully expendable since its wide support of an *XML* tools. For this reason it can be seen as a base for further (even more precise) system description in case of both dependability and detailed description of the system.

Described architecture can be finding as an idea of a tool that can visualize and analyse data, with respect to real parameters. No restriction on the system structure and on a kind of distribution describing the system functional and reliability parameters is the main advantage of the approach.

The solution presented can be used as practical tool for defining an organization of vehicle maintenance and transportation system logistics.

Acknowledgment

Work reported in this paper was sponsored by a grant No. N N509 496238, (years: 2010-2013) from the Polish Ministry of Science and Higher Education.

References

1. Barcelo, J., Codina, E., Casas, J., Ferrer, J.L.,Garcia, D. Microscopic Traffic Simulation: a Tool for the Design, Analysis And Evaluation Of Intelligent Transportation Systems, *Journal of Intelligent and Robotic Systems: Theory and Applications*, Vol. 41, 2005, pp. 173-203.
2. Bonabeau, E. *Agent-based modelling: methods and. techniques for simulating human systems*, presented at. Proc Natl Acad Sci, 2002.
3. Duinkerken, M.B., Dekker, R., Kurstjens, S.T.G.L., Ottjes, J.A., Dellaert, N.P. Comparing Transportation Systems for Inter-Terminal Transportation at the Maasvlakte Container Terminals, *OR Spectrum*, Vol. 28, 2006, pp. 469-493.
4. Gao, Y., Freeh, V. W., Madey, G. R. Conceptual Framework for Agent-based Modelling and Simulation. In: *Proceedings of NAACSOS Conference*. Pittsburgh, 2003.
5. Gartner, N., Messer, C.J., Rathi, A.K., Traffic Flow Theory and Characteristics. In: *T.R. Board, ed.* Texas:University of Texas at Austin, 1998. 125 p.
6. Ioannou, P.A. *Intelligent Freight Transportation*. Carolina: Taylor and Francis Group, 2008. 245 p.
7. Jennings, N. R. On Agent-Based Software Engineering, *Artificial Intelligence,* Vol. 117, April 2000, pp. 277-296.
8. Liu, H., Chu, L., Recker, W. Performance Evaluation of ITS Strategies Using Microscopic Simulation. In: *Proceedings of the 7th International IEEE Conference on Intelligent Transportation Systems*, 2004, pp. 255-270.
9. Mascal, C. M., North, M.J. Tutorial on agent-based modelling and simulation. In: *Winter Simulation Conference,* December 2005.
10. Mellouli, S., Moulin, B., Mineau, G. W. *Laying Down the Foundations of an Agent Modelling Methodology for Fault-Tolerant Multi-agent Systems*, ESAW 2003, pp. 275-293.
11. Michalska, K., Mazurkiewicz, J. *Functional and Dependability Approach to Transport Services Using Modelling Language*; Springer LNCS/LNAI series (accepted for publication)
12. Vis, I.F.A. Survey of Research in the Design and Control of Automated Guided Vehicle Systems, *European Journal of Operational Research*, Vol. 170, 2006, pp. 677-709.
13. Parulkar, G., Schmidt, D., Kraemer, E., Turner, J., Kantawala, A. An architecture for monitoring, visualization and control of gigabit networks, *IEEE Network*, Vol. 11 (5) October 1997, pp. 34-43.
14. Walkowiak, T. and J. Mazurkiewicz. Algorithmic Approach to Vehicle Dispatching in Discrete Transportation Systems. In: *Technical approach to dependability / Ed. by Jarosław Sugier, [et al.].* Wroclaw: Oficyna Wydawnicza Politechniki Wroclawskiej, 2010, pp. 173-188.
15. Walkowiak, T. and J. Mazurkiewicz. Availability of Discrete Transportation System Simulated by SSF Tool, In: *Proceedings of International Conference on Dependability of Computer Systems*,

Szklarska Poreba, *Poland, June, 2008*. Los Alamitos: IEEE Computer Society Press, 2008, pp. 430-437.

16. Walkowiak, T., and Mazurkiewicz J. Functional Availability Analysis of Discrete Transport System Simulated by SSF Tool, *International Journal of Critical Computer-Based Systems*. Vol. 1, No 1-3, 2010, pp. 255-266.
17. Walkowiak T. Mazurkiewicz J. Soft Computing Approach to Discrete Transportation System Management. In: *Lecture Notes in Computer Science. Lecture Notes in Artificial Intelligence*. Springer-Verlag, 2010. Vol. 6114, pp. 675-682.
18. Wooldrige M. Agent-based software engineering, *Software Engineering, IEEE Proceedings,* Vol. 144 (1), February 1997, pp. 26-37.