A Multi-agent Scheduler for Rent-a-Car Companies

Slava Andreev[1], George Rzevski[2],
Peter Shviekin[1], Peter Skobelev[1], and Igor Yankov[1]

[1] Magenta Technology 349 N-Sadovaya St. Samara 443125 Russia, +7846 342 51 74
skobelev@magenta-technology.ru
[2] Rzevski Solutions, 3 Ashbourne Close W5 3EF London, UK, +44 20 8998 8538
george@rzevski.net

Abstract. The paper gives overview of a multi-agent real-time scheduler for the European operation of one of the largest rent-a-car company in the world. It describes requirements for scheduling of cars and drivers and outlines main features of the ontology-based, multi-agent approach, including systems architecture and performance measurements. The key design decisions and results of the first stage of the system development are also covered. The system is capable of scheduling complex interdependent operations of a large number of resources and of updating schedules affected by the occurrence of unpredictable events in real time. The multi-agent approach developed for scheduling of car rentals can be applied to a variety of complex real-time scheduling and optimization applications.

Keywords: Dynamic scheduling, real-time planning, multi-agent systems, microeconomics, ongoing optimization, events, criteria of decision-making.

1 Introduction

The problem of resource scheduling in real time is one of the most difficult problems in the modern theory of optimization [1-3]. The problem is particularly complex when it contains a large number of interconnected participants and when changes in the schedule of one participant affect schedules of other participants.

In the present paper we consider scheduling of car deliveries for a rent-a-car company, which falls into this class of problems. The paper shows that the problem can be effectively solved using multi-agent technology. Agents, often with conflicting interests, dynamically create a schedule through a process of negotiations and distributed decision-making. The pilot system has been developed and delivered to the client and is working successfully. The second phase scheduler capable of handling 25 rental stations will be delivered in the near future.

The paper covers specifics of a rent-a-car scheduling problem, outlines multi-agent approach to the solution of this problem, describes architecture of the scheduler and singles out main features of its implementation, including performance measurements. The method developed for scheduling rental cars can be also applied to a wide range of other problems in transportation and production logistics, as well as many other industrial problems.

V. Mařík, T. Strasser, and A. Zoitl (Eds.): HoloMAS 2009, LNAI 5696, pp. 305–314, 2009.

2 The Problem Domain of Rent-a-Car Scheduling

Scheduling problem for a rent-a-car business is deceptively simple: the client specifies the group of the required car, time and place where the car will be picked up and time and place where car will be dropped.

However, the solution to this problem is very complex. The system is required to (a) locate a car belonging to the specified group that can be delivered to the specified location at the specified time, taking into account distances, costs and client's expressed preferences; (b) assign a driver that will wash the car and deliver it to the specified location according to company business processes, taking into account driver's overtime and (c) schedule the transport for the driver to the car location and, after the delivery, to the station where his next job is waiting, or to his home, which may involve another car and another driver. The key difficulty is that the problem is changing while you are attempting to solve it due to the occurrence of unpredictable events: new orders arrive, previous orders are modified or cancelled, cars break down, drivers fall ill or take a day off and delays in delivery occur due to traffic jams.

Usually the territory of a rent-a-car business is divided into a number of small regions each consisting of several rental stations, where a number of skilled managers are engaged in manual scheduling of cars and drivers.

Region-based manual scheduling is however ineffective in respect to the overall optimization of the fleet size, minimization of car delivery distances and achieving acceptable service levels. A more effective scheduling is required for a rent-a-car business to attain sustainable growth and stability.

For developing scheduling solution for rent-a-car business it is necessary to consider the following requirements.

Scheduling of drivers and cars must consider interconnected operations, which are performed at different stages of business processes. A good example is the case when several drivers need to be consolidated in one car to solve delivery and/or collection problems for several orders.

It is important to be able to combine and balance different criteria of decision-making, and to remember that these criteria may change in time. The major criteria include run cost per mile of the car and the driver, the penalty for a delay to rental, the penalty for the late delivery of the returned car to a home station, cost of driver overtimes, and the penalty for not matching the delivered car to the requested car group (upgrade/downgrade).

Managers and drivers in a rent-a-car business are constantly under pressure because of frequent occurrence of unpredictable events and this can negatively affect the quality of schedules of all participants. For example, a car may break down, or the driver could not find keys from the collected car, or could not find the client of the delivered car.

Combined planning and execution is the key feature of a scheduler in the rent-a-car business. The scheduler must respond quickly to a growing gap between the plan and reality, whenever it occurs, because the goal of the system is not only to generate and update schedules but also to monitor and control the execution of planned tasks. The planned tasks are passed into execution by means of mobile devices of drivers who are

obliged to confirm the beginning and the end of each operation. A delay in the confirmation that a task has been executed may cause a chain of re-planning of other tasks.

The scheduling process in practice is a process of discovering and solving conflicts among orders rather than a search for the optimal solution. Thus an ability to reach a reasonable trade-off between different criteria is of great importance. To achieve these trade-offs effectively the system makes use of virtual money.

Rent-a-car problems are characterized by the large solution spaces, which make the use of combinatorial algorithms difficult and makes classical methods of local search ineffective. The method described in this paper rapidly responds to unpredictable events as they occur and, in intervals between events, it attempts to find a near-optimal schedule. Designing a scheduler that can cope with such a variety of operating conditions, handle uncertainty related to the occurrence of events and at the same time continuously produce schedules that maximize the set of specified criteria is a real intellectual challenge. To the best of our knowledge such schedulers have not been described in the literature or implemented in practice.

3 Solution

The solution was developed using the multi-agent approach based on the concept of Demand-Resource Networks (DRN) [4-7]. The scheduling process is executed by dynamic interaction of agents representing demands and resources of a company business network. Basic agents for this type of business include: client agents, order agents, station agents, car agents, driver agents, car delivery agents and car collection agents.

The scheduling process starts with partitioning one big task “to deliver a car” into several smaller sub-tasks. Every sub-task is scheduled independently and initiates a complex mechanism of agent negotiations. For example, the agent of a simple pickup task selects the optimum resource (the most suitable car) and creates a subordinated task, say a car wash task, whose agent searches for the driver able to wash the delivered car in specified time. In a more complex case the number of subtasks can be much greater, and tasks have interdependences, i.e., cause and effect relationships. For instance, during scheduling a car rental, the rental agent searches for a suitable car, the car agent searches for the best driver for delivery to the client, the driver agent searches for another driver (runner) to bring him/her back home after car delivery, etc.

The process of matching a resource (a car) to a demand (a rental order) may create or uncover conflicts that will have to be resolved. For example, if during the matching the rental order has selected the car which is already reserved for another client then the conflict can be resolved by refusing this match or by finding another car for the first client, whatever is the nearest to the optimum. Under certain conditions the new rental order can "move" itself to a point in time which, although less desirable, resolves the conflict. Such compromise is required, for example, if all drivers are already allocated for other rentals, and unless the delivery is brought forward, the order cannot be fulfilled. The degree of dissatisfaction of an agent caused by a compromise can be compensated by virtual bonuses. On other occasions agents may be penalized

for the infringement of restrictions. Values of bonuses/penalties are determined by the client and can be revised at any time.

A matching between a resource and a demand, established during interaction of agents, is not necessary permanent. Agents perpetually search for improvements to the schedule and may reconsider previously made decisions, break old links and establish new ones, thus reaching new deals that satisfy decision-making criteria better. Pro-activity of agents enables the improvement of the schedule when the system is idle.

In comparison with classical approaches to optimization the multi-agent scheduler described in this paper offers important advantages:

- The scheduler can be easily modified and updated as the business environment and consequently business polices and processes evolve. This can be done by the addition of new agents representing interests of new participants, tasks or resources
- The scheduler is suitable for large scale applications where there is a need to handle thousands of interrelated rentals, tasks and resources, each with a variety of features
- The scheduler performs event-driven dynamic scheduling in real time and is capable of uncovering and resolving conflicts in an existing schedule by changing only affected parts of the schedule
- The scheduler considers every rental order and every resource individually with corresponding agents capable of working with a variety of different criteria, preferences and constraints
- The scheduler allows both users and agents to change criteria, preferences and constraints during its operation and also, if necessary, to modify the balance between various criteria and constraints, e.g., between the service level (car class), delivery time or cost, delivery risks, driver discomfort, etc
- The scheduler is capable of explaining to the users how it made decisions and to offer them opportunities to modify created schedules.

The design of the car rental scheduler is based on earlier designs of schedulers for tankers, trucks and taxi [7-12].

4 Ontology of a Rent-a-Car Business

Ontology is a repository of conceptual knowledge that separates the specific knowledge of the rent-a-car domain from the program code. The problem domain knowledge is represented in ontology as a semantic network of domain concepts, defined by their attributes, and relations. Playing a role of the "dictionary" of the system, ontology provides a basis for constructing instantaneous models of the domain, known as Scenes. A scene shows that at the particular point in time the given car is reserved for the given rental and that it is being delivered to the pick-up station by the given driver. Such a model allows agents to track important links between objects, operations and participants.

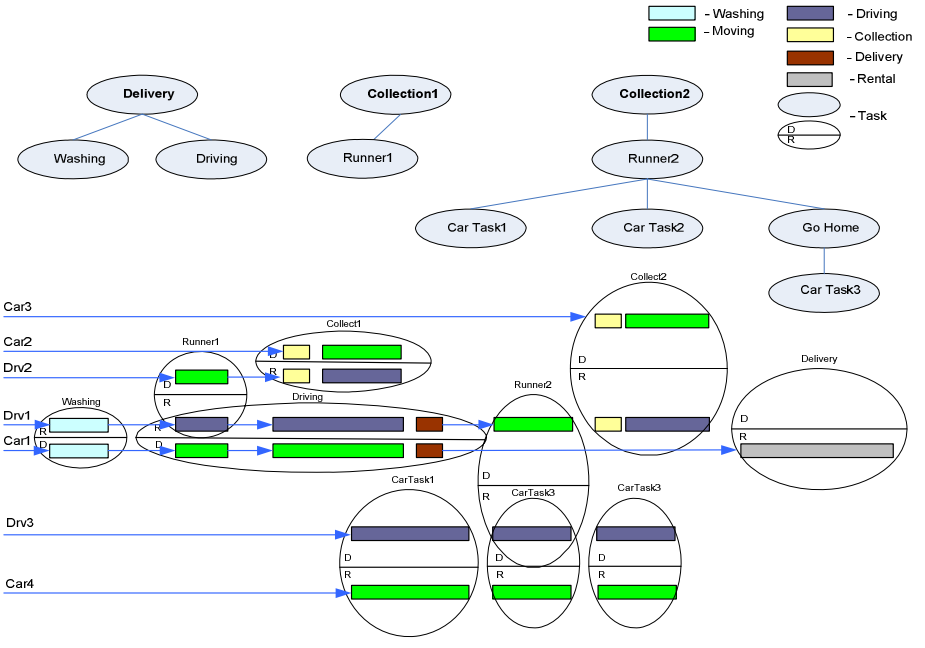

Fig. 1. Example of a Scene

5 Multi-agent World of Rent-a-Car Scheduling

The schedule produced by the multi-agent scheduler is a multi-level network of tasks (operations) linked by relations as shown in figure 1.

Every task and every sub-task have their own agents called Demand Agents. The main goal of a demand agent is to search for the most suitable way of performing the task or sub-task and this includes the interrogation of and negotiation with resources, which are entitled to accept or refuse proposals from demand agents. Agents with different decision-making logic are created for different types of tasks.

A Resource Agent is assigned to every business resource and its first task is to estimate costs of using this particular resource to fulfil a task (rental) proposed by a demand agent. Resource agents calculate preliminary cost (Average Cost) and full cost (Marginal Cost) of using the resource for rental, partition initial tasks into constituent sub-tasks and direct inquiries to corresponding agents. The most important function of a resource agent is to create "promise" (obligation) to carry out a task for a specified sum of money.

The scheduler operates with seven types of tasks, and therefore with seven types of demand agents: Delivery; Washing Task; Driving Task; Collect; Runner Task; Car Task; Go Home Task. It uses two types of resources: Car and Driver.

Agents are driven by Events and majority of events are initiated by demand, for example, scheduling and re-scheduling of a new reservation, change of details of car collection, etc. In some situations resource agents are given power to be proactive and to initiate search for tasks for which the given resource would be the best option. If

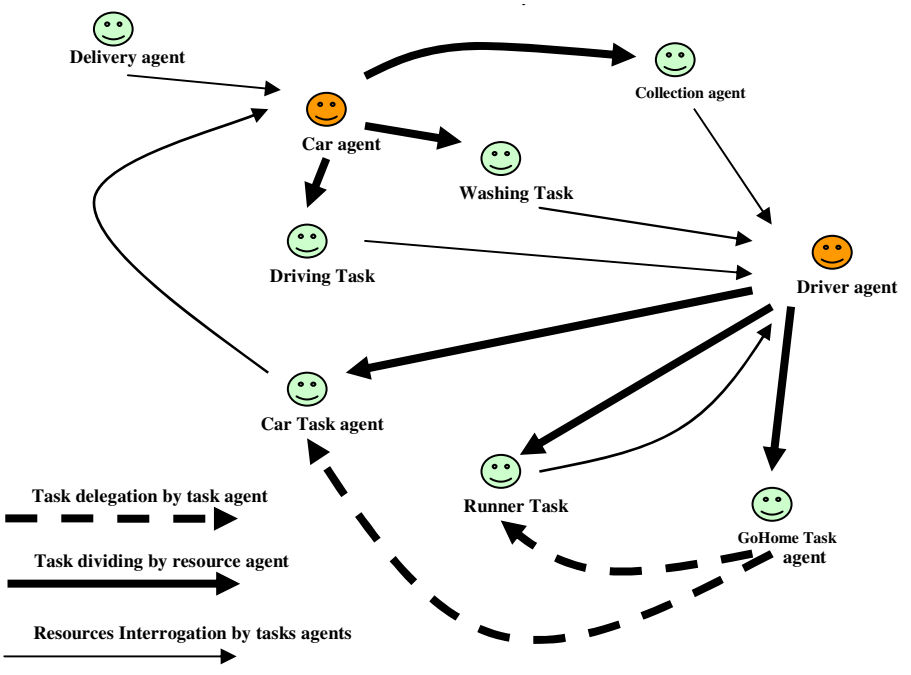

Fig. 2. Multi-Agent World of Scheduling

such a task is found, the resource agent initiates re-scheduling process. Occasionally events are generated by users; these are usually urgent requests for rescheduling due to task failures or changes in schedules of drivers.

The approach to scheduling of car rentals has a number of advantages over other approaches, but it does not guarantee that the created schedule is optimal for all resources for each moment of time, because the scheduler works in real time and the period between the occurrence of events, which cause rescheduling, is too short. The system attempts to create, within the available time interval, a schedule that is as near as practical to the optimum using appropriate heuristics.

To improve the schedule selected agents are temporary made proactive. Perhaps the most important example is the triggering of the Schedule Agent to review the created schedule and report any inefficient use of resources. Using this information, agents identified in the report may attempt to renegotiate their deals. In addition, resource agents from time to time get pro-activity cycles to search for the tasks that may be better suited for them and, if necessary to renegotiate deals with existing tasks. Any renegotiation is permitted only if it improves key indicators for the whole systems. For example, if the driver is waiting in the field, after delivering a car, and is given a new task but his runner is delayed, after a while his agent will start pro-actively searching for a new runner. Thus, in the agent world the pro-activity threads are continuously operating with a view to achieving the optimization of the schedule in a step-wise manner. During re-negotiations the system can temporarily break old links between tasks and resources and thus destroy the integrity of the schedule. As soon as new deals are completed, agents refresh links and recover the schedule, which is now updated.

Ongoing pro-activity is related to KPIs of the schedule. If a current type of pro-activity is not improving KPIs then another type of pro-activity start to push the schedule towards the optimum more aggressively. The approach thus uses "trial and error" method which helps to improve complex schedules and self-regulate internal system activities in a real time.

6 Architecture

The architecture of the multi-regional car rental system is shown in figure 3. The diagram depicts main system components and the 3rd party software.

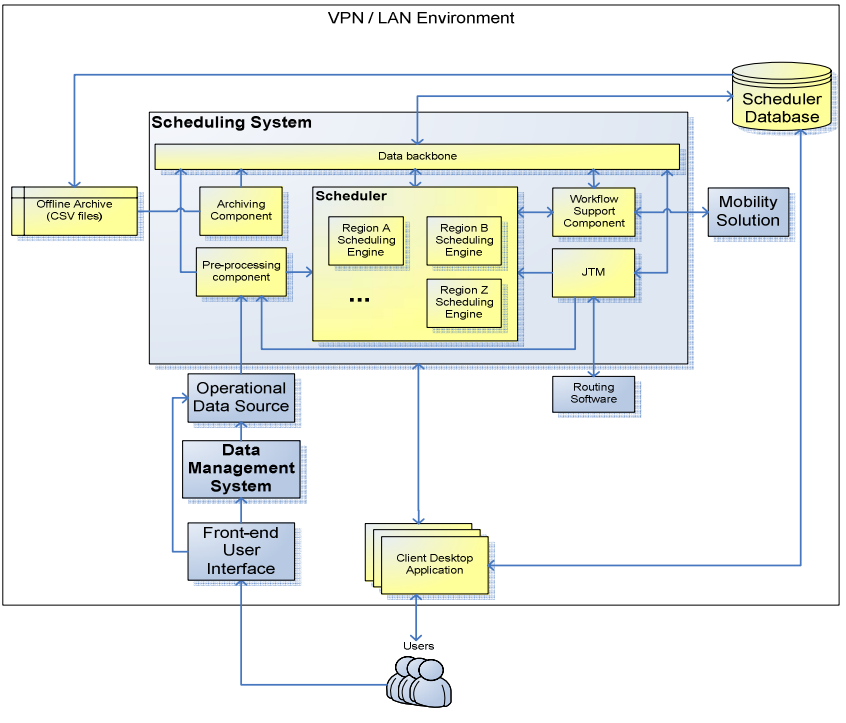

Fig. 3. Architecture

7 User Interfaces

The system provides a comprehensive desktop user interface for day-to-day Station activities (tasks, operational information about cars, drivers and their schedule), Fleet managers (fleet movements functionality) and Administrator (exception management and system maintenance). Client application is automatically deployed and updated on user PCs via Java Web Start service.

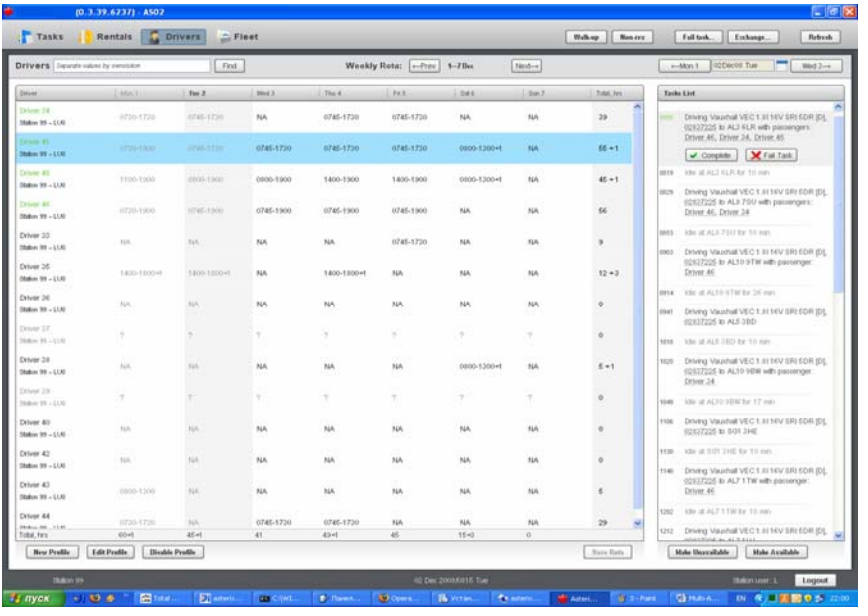

Fig. 4. Task list

Table 1. Performance matrixes

The amount of cars	250
The amount of drivers	30
Average stream of rentals	15-20 reservations per day for one station
Average number of late task	A driver was late once for 5 minutes for one task, on the average. Approx. 60 % of all tasks were late.
Average frequency of events	About 80-120 events per hour
Time for processing of one event	Maximum - 130 sec, average - 7 sec (70 % of all events were processed in the system in less than 3 sec)
Number of the tasks which are scheduled for one event	From 150 till 200, including: Runner Tasks – 30.7 ,Driving Tasks – 7.28, Washing Tasks – 6.64, Delivery – 1.88, Car Task – 137.89, Collect - 3.75
Messaging intensity in the system	Tasks involving a car, establish 200-250 connections. Tasks involving drivers – 30-40.The number of messages per task is at least 1500
Number of simultaneously active agents	Task agents - 2500 Car agents – 250 Driver agents – 30
The greatest depth of a negotiations chain	10 levels (limited manually)
What percent of decisions is made more than an hour before the beginning of schedule execution	Less than 1 %

The Drivers screen (figure 4) displays all drivers and their working hours planned for the current week (Rota): by weekdays and the total number of hours. To see tasks currently planned for a driver the user selects the driver from the list. Tasks are listed in the chronological order.

8 Performance Matrixes

At present the system is tested on a network of stations of a large rent-a-car company. A number of performance matrixes, which represent most important characteristics of the developed scheduler are presented in table 1.

9 Conclusion

The system described in this paper can be considered as a one of the first industrial multi-agent systems for dynamic scheduling.

The system exhibits a considerable intelligence by being able to autonomously react to events in real time, to generate schedules by resolving many conflicting issues, execute the schedule and monitor results. When differences between the schedule and the reality is recognized the systems triggers re-scheduling processes attempting always to match the plan to reality. When there is sufficient time between two events the system autonomously works on improving given KPIs. Pro-activity and self-organisation are the key system features.

A number of R&D issues have been identified, which need further investigations, including measuring quality of produced schedules, effective interaction with users, balancing conflicting performance criteria and freezing last minute changes, etc. For example, frequent human errors, delays of drivers, incorrect addresses of deliveries and collections, and failed driving tasks can lead to long “ripple effects” of changes in schedules, occasionally too near to the execution.

Our first results prove that multi-agent technology forms a solid basis for addressing above issues and developing new types of solutions, which can provide great opportunities for solving very complex scheduling and optimization problems in real time.

References

1. Leung, J.Y.-T. (ed.): Handbook of Scheduling: Algorithms, Models and Performance Analysis. Computer and Information Science Series. Chapman & Hall / CRC, Boca Raton (2004)
2. Voß, S.: Meta-heuristics: The state of the art. In: Nareyek, A. (ed.) ECAI-WS 2000. LNCS, vol. 2148, p. 1. Springer, Heidelberg (2001)
3. Rego, C., Alidaee, B. (eds.): Metaheuristic Optimization Via Memory and Evolution: Tabu Search and Scatter Search, Boston-London. Operational Research & Computer Science Series (2005)
4. Dorer, K., Calisti, M.: An adaptive solution to dynamic transport optimisation. In: Pechoucek, M., Steiner, D., Thompson, S. (eds.) Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multi-Agent Systems: Industry Track, pp. 45–51. ACM Press, New York (2005)

5. Vittikh, V.A., Skobelev, P.O.: The multi-agent models of interaction in demand-resource networks. Automatica and Telemechanica (1), 177–185 (2003)
6. van der Putten, S., Robu, V., La Poutré, H., Jorritsma, A., Gal, M.: Automating supply chain negotiations using autonomous agents: a case study in transportation logistics. In: Proceedings of the fifth international joint conference on Autonomous agents and multi-agent systems, Hakodate, Japan, May 08-12 (2006)
7. Rzevski, G.A., Skobelev, P.O., Andreev, V.V.: MagentaToolkit: A set of multi-agent tools for developing adaptive real-time applications. In: Mařík, V., Vyatkin, V., Colombo, A.W. (eds.) HoloMAS 2007. LNCS, vol. 4659, pp. 303–313. Springer, Heidelberg (2007)
8. Rzevski, G., Himoff, J., Skobelev, P.: Magenta Technology: A Family of Multi-Agent Intelligent Schedulers. In: Proceedings of Workshop on Software Agents in Information Systems and Industrial Applications (SAISIA). - Fraunhofer IITB, Germany (February 2006)
9. Himoff, J., Skobelev, P.O., Wooldridge, M.: Multi-Agent Systems for Ocean Logistics. In: Proceedings of 4-th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2005), Holland (July 2005)
10. Himoff, J., Rzevski, G.A, Skobelev, P.O.: Magenta Technology: Multi-Agent Logistics i-Scheduler for Road Transportation. In: Proceedings of 5-th International Conference on Autonomous Agents and Multi Agent Systems (AAMAS 2006), Japan (May 2006)
11. Glashenko, A., Ivashenko, A., Rzevski, G., Skobelev, P.: Multi-Agent Real Time Scheduling System for Taxi Companies. In: Decker, Sichman, Sierra, Castelfranchi (eds.) Proc. of 8th Int. Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS 2009), Budapest, Hungary, May 10–15 (2009) (in publ.)
12. Rzevski, G.A., Skobelev, P.O.: Emergent Intelligence in Large Scale Multi-Agent Systems. Education and Information Technologies Journal 1(2) (2007)