Bulletin of the *Transilvania* University of Braşov • Vol. 3 (52) - 2010
Series I: Engineering Sciences

PETRI NETS AND AGENTS TO SUPERVISORY CONTROL OF COMPLEX ENVIRONMENT

F. MOLDOVEANU[1] D. FLOROIAN[1] D. PUIU[1]

Abstract: *In semiautonomous mobile sensor networks, since human operators may be involved in the control loop, particular improper actions may cause accidents and result in catastrophes. For such systems, this paper proposes a command filtering framework to accept or reject the human-issued commands so that undesirable executions are never performed. In the present approach, Petri nets are used to model the operated behaviors and to synthesize the command filters for supervision. Also the command filter could be implemented using agent technology by associating a filtering agent for each robot. It is believed that the technique presented in this paper could be further applied to large-scale wireless mobile sensor networks.*

Key words: *sensors network, Petri net, agent, supervisory control.*

1. Introduction

Sensor networks (SNs) have recently received significant attention in the areas of networking, embedded systems, pervasive computing, and multiagent systems due to its wide array of real-world applications. In the last few years, there has been an increasing emphasis on a developing wide-area distributed wireless sensor networks (WSNs) with self-organization capabilities to cope with sensor failures, changing environmental conditions, and different environmental sensing applications [2], [15]. In particular, mobile sensor networks (MSNs) hold out the hope to support self-configuration mechanisms, guaranteeing adaptability, scalability, and optimal performance, since the best network configuration is usually time varying and context dependent [10]. Mobile sensors can physically change the network topology, reacting to the events of the environment or to changes in the mission planning. On the other hand Petriu et al. [12] have studied the networks of autonomous robotic sensor agents for active investigation of complex environments.

In real applications, human operators may use semiautonomous robots, as shown in Figure 1a, to 1) further investigate conditions if several static sensors launch an alert; 2) maintain network coverage for both sensing and communication; 3) charge the static sensors, or 4) repair, replace, or remove the static sensors [9]. For such "human-in-the-loop" systems, human errors have a significant influence on system reliability, at times more than technological failures. Research results indicate that the vast majority of industrial accidents are

[1] Centre "Control Process Systems", *Transilvania* University of Braşov.

attributed to human errors. Lee and Hsu [6] proposes (for the first time using Petri nets (PNs)) a technique to design supervisory agents for preventing abnormal human operations from being carried out. This supervisory approach was also applied to human-computer interactive systems [7]. In [5] Lee and Chung propose a PN-based localization scheme on a discrete event control framework for indoor service robots.

From the high-level point of view, an human-in-the-loop system is inherently a discrete-event system (DES), i.e., a dynamic system with state changes driven by occurrences of individual events. Supervisory control theory provides a suitable framework for analysing DES [13]. Figure 1b adopts the supervisory framework [6], [7], to an MSN system composed of several static sensors and semiautonomous mobile robots regulated by human operators through a wireless network. According to the status feedback from both the sensors and robots, the supervisors provide permitted commands for human operators by disabling the actions which violate specifications. The human operator can then trigger only limited commands based on the observed status. However, the supervision is from an active viewpoint to enable or disable the commands in advance and leads to limited human actions. In addition, the MSN system requires a fast sampling rate with low-latency communication to provide supervisors with an up-to-date status to make the decision. Furthermore, each supervisor and the MSN system is based upon a client/server architecture with centralized communication, which is not an ideal topology for distributed sensor network systems.

In this paper, instead of using a client/ server architecture, distributed peer-to-peer (P2P) communication between mobile robots is applied [9]. Moreover, from a passive point of view, a command filter [7] is proposed to avoid improper control actions from being carried out as the robot receives the human commands.

As shown in Figure 2 [9], the human operator sends command requests to the mobile robot through a wireless network. Inside the robotic computer, the command filter acquires the system status via distributed P2P communication and makes the decision to accept or rejects the commands so as to meet the specifications, e.g., the collision avoidance among robots. The role of a command filter is to interact with the human operator and the mobile robot so that the closed human-in-the-loop system satisfies the requirements and guarantees that undesirable executions never occur.

In such a scenario agents could be used for command filtering. Also with an user interface program, agents could also interact with human operators for set-up the filtering process. There are also special agents for network integration [14].

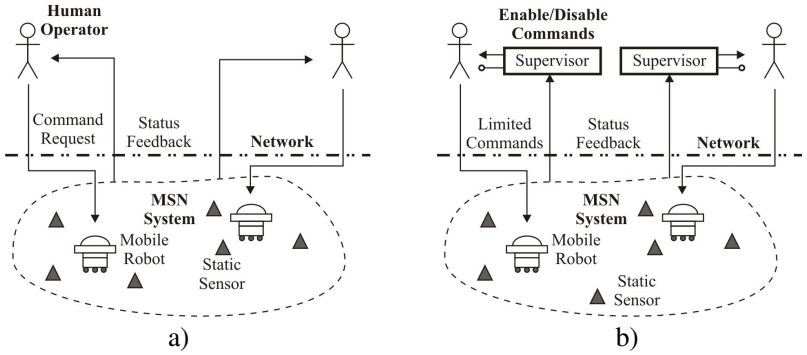

Fig. 1. *a) Human-involved MSNs; b) Applied supervisory framework for such MSN systems*

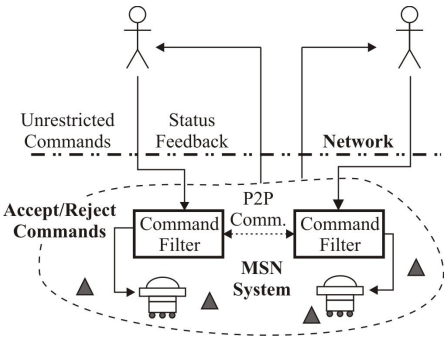

Fig. 2. *Command filtering framework for MSNs*

Supervisory (centralized) - control techniques have been studied to overcome the inherent limitations of decentralized approaches, including lack of the ability to provide fast and globally optimal solutions.

Some significant results in a supervisory control have been obtained using PNs [11], [13]. In this paper, PNs are used in designing the command filters, yielding a compact and graphical model for the MSN. Basically, the PN design of the filters is identical to the design of the supervisors in [6] and [7], except for the implementation framework as shown in Figures 1b and 2.

The organization of the paper is as follows. A systematically design procedure of the command filter synthesis is described in Section 2. Then, in Section 3, an example of a mobile wireless surveillance system is illustrated to show the feasibility. Finally, Section 4 gives the conclusions.

2. Petri Nets - Based Command Filter Design

2.1. Modeling of Semiautonomous MSNs

PNs have been used to model, analyse, and synthesize control laws for DES [11], [13], [17]. Zhou and DiCesare [16], moreover, addressing the shared resource problem recognized that mutual exclusion theory plays a key role in synthesizing a bounded, live, and reversible PN. In this paper, we adopt mutual exclusion theory to build the PN specification model, and then compose it with the plant model to design the supervisor. Moreover, in semiautonomous MSNs, human behavior can be modeled using the command/ response concept. As shown in Figure 3, each human operation is modeled as a task with a start transition, end transition, progressive place and completed place. Transitions drawn with dark symbols are events controllable by the remotely located human through the network. Note that the start transition is a controllable event as "command" input, while the end transition is an uncontrollable event as "response" output. On the other hand, nonhuman actions can be simply modeled as a single event transition.

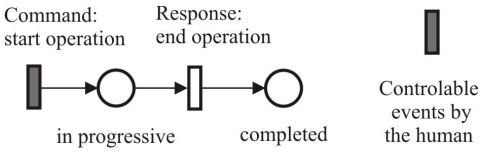

Fig. 3. *Modeling of human behavior using the command/response concept*

2.2. Specification Types

The objective of a command filter is to ensure the reaction of human-issued commands contained within the set of admissible states, called the specification. In this paper, two main types of specifications are considered and described as follows:

1. Collision-avoidance movements: This specification presents the physical constraints of the limited resources, such as the rooms and hallways. Each room limits the number of mobile robots that enter or stay avoids collisions.

2. Deadlock-free operations: This specification ensures that a given command will not lead the system to a deadlock state at which no further action is possible. This specification can be preserved by deadlock avoidance policies.

During the system operation, the proposed command filter enforces these specifications by accepting or rejecting human-issued commands.

2.3. Synthesis of Command Filters

In this paper, an agent that specifies which events are to be accepted or rejected when the system is in a given state is called a *command filter* [8]. The design procedure of PN-based command filters consists of the following steps:

1. Construct the PN model of the human commands and system responses.
2. Model the required specifications.
3. Compose the system and specification models to synthesize the preliminary command filter.
4. Analyze and verify the properties of the composed model.
5. Refine the model to obtain a deadlock-free, bounded, and reversible model according to the defined specifications.

2.4. Implementing Using Agent Technology

Agent technology is a new and important technique in recent novel researches of artificial intelligence [3]. Wooldridge and Jennings [14] depicts an agent as a computer system that is situated in some environment and is capable of autonomous actions in this environment to meet its design objectives. The distributed multiagent coordination system is defined as the agents that share the desired tasks in a cooperative point of view and are autonomously executing at different sites. For command filtering this represents a very important quality that makes agent technology very useful for desired task.

For our purposes, we have adopted the description of an agent as a software program with the capabilities of sensing, computing, and networking associated with the specific function of command filtering for the MSN systems. A filtering agent is implemented to acquire the system status by autonomously sensing and the P2P networking abilities, after which computing is performed to accept or reject the associated commands so that desired specifications are satisfied. This implementation is made in JADE [18] because this development tools is very versatile and could be very well integrated with others development tools (like Protégé-2000 and Java [19]). The general management console for a JADE agent platform (RMA), like in Figure 4, acquires the information about the platform and executes the GUI (Graphic User Interface) commands to modify the status of the platform (creating new agents, shutting down containers etc.) through the AMS (Agent Management System).

To facilitate message reply, which, according to FIPA, must be formed taking into account a set of well-formed rules such as setting the appropriate value for the attributes *in-reply-to*, using the same *conversation-id* etc., the method *createReply*() is defined in the class that defines the ACL (Agent Communication Language) message. Different types of primitives are also included to facilitate the implementation of content languages other than SL, which is

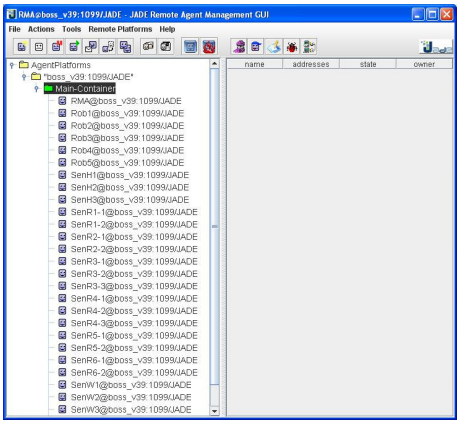

Fig. 4. *Graphic User Interface of JADE development tool, used for manage the agents*

the default content language defined by FIPA for ACL messages. This facility is made with Protégé as depicted in Figure 5.

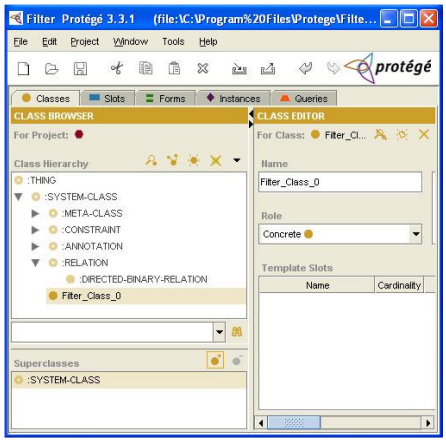

Fig. 5. *Defining ontologies for filtering class, using Protégé-2000*

From JADE point of view, the filtering agent of each robot would run the developed PN model as a state machine and have capabilities with position location (room number in our case).

2.5. Localization Technologies

For a past decade, localization technologies based on WSNs have appeared as an alternative to the traditional approaches. In WSN-based localization, a portable wireless localization tag is attached to a mobile robot, and this tag is localized by distributed sensor networks. The most popular wireless localization tags are ultrawideband (UWB) tag, ZigBee sensor node, radio frequency identification (RFID), and small-size wireless LAN (WLAN), WiFi compliant card.

These existing technologies have their own disadvantages [1]: for example, UWB (standardized in IEEE 802.15.4a) systems require heavy hardware installation and initial network calibration, and it is difficult to locate a mobile object using commercial UWB localization products; in ZigBee (standardized in IEEE 802.15.4) - based localization systems, a signal propagation model should be produced in advance to evaluate the range between reference nodes, and the signal characteristic is environmental and time dependent. The RFID technology is reliable since it uses RFID tags pasted on the floor to indicate the robot position; however, it is disadvantageous to paste many RFID tags on the floor; if tags cannot be attached to the floor, the RFID system requires ID carpet. In WLAN (standardized in IEEE 802.11) - based localization systems, low accuracy and slow response are observed as main technical problems. Moreover, in WLAN systems, a radio map is requested to match the observations to reference signal strengths.

A common and critical problem of the wireless localization networks is that there could be lots of measurement noises in the computed localization information and/or in the measured signal strength. Although some advanced filtering technologies such as Kalman [4] filtering may be utilized to reduce the noise effects in actual applications.

In this paper each room is equipped with a sensing and communication device, such as an RFID reader or a ZigBee module, to provide the vacancy information to the robot which would enter a particular room. On the other hand, as it mentioned before, the robot may also communicate with other robots to obtain their locations.

3. Example: A Mobile Wireless Surveilance System

3.1. System Description

The semiautonomous MSN system in Figure 2 can be applied to a mobile wireless surveillance system, which is composed of many static sensors and several human-controlled mobile robots. In this example, five mobile robots are placed on a floor with eight rooms, as shown in Figure 6.

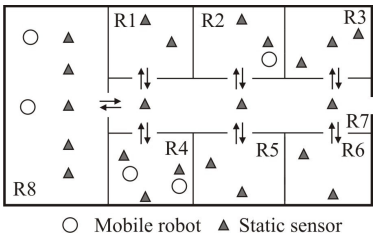

Fig. 6. *Mobile wireless surveillance system with five robots*

Each robot can move to each room according to indicated directions. To avoid possible collisions, the number of robots from each room is limited. So in the rooms R1, R2, R5 and R6 can work only one robot; rooms R3 and R4 admits two robots during the surveillance period, and in the rooms R7 and R8 can stay three respectively five robots. Initially all the robots are in the room R8.

3.2. PN Modelling

By applying the command/response concept and based on the system description, the PN model of the human-controlled mobile robots is constructed as shown in Figure 7. It consists of 22 places and 28 transitions, respectively. Corresponding notation of the PN model is described in Table 1.

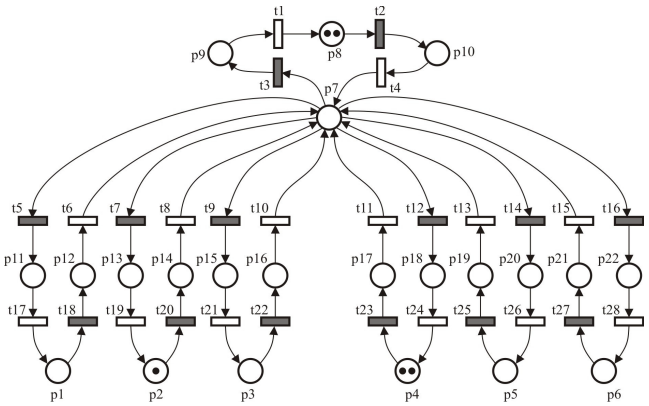

Fig. 7. *PN model of the human-controlled mobile robots*

3.3. Command Filter Design

The eight rooms represent the resources shared by the five mobile robots. Since more than one robot may require access to the same room and each room has a limited number of robots collisions and deadlocks may occur. Hence, the objective is to design a command filter to ensure the whole system against these undesired situations. The required five specifications are formulated as follows:

S-1. Robot moving to Room i is allowed only when Room i is empty, where i = 1, 2, 5, 6. Thus, we have four subspecifications denoted as S 1.1 to S 1.4.

S-2. Robot moving to Room i is allowed only when Room i has no more than one robot, where i = 3, 4. Thus, we have two subspecifications denotes as S 2.1 to S 2.2.

S-3. Robot moving to Room 7 is allowed only when Room 7 has no more than two robots. Thus, we have one subspecification S 3.1.

S-4. Robot moving to Room 8 is allowed only when Room 8 has no more than four robots. Thus, we have one subspecification S 4.1.

S-5. Liveness, i.e., no deadlock states, must be enforced throughout system operation.

Table 1

Notification of the pn places and transitions in Figure 7

Place	Description	Trans.	Description
p1	Robot in R2	t1	Cmd: start moving to R1
p2	Moving to R1	t2	Re: end moving to R1
p3	Moving to R2	t3	Cmd: start moving to R2
p4	Robot in R1	t4	Re: end moving to R2
p5	Moving to R4	t5	Cmd: start moving to R4
p6	Moving to R1	t6	Re: end moving to R4
p7	Robot in R4	t7	Cmd: start moving to R1
p8	Moving to R2	t8	Re: end moving to R1
p9	Moving to R4	t9	Cmd: start moving to R2
p10	Moving to R3	t10	Re: end moving to R2
p11	Moving to R2	t11	Cmd: start moving to R4
p12	Robot in R3	t12	Re: end moving to R4
p13	Moving to R5	t13	Cmd: start moving to R3
p14	Moving to R3	t14	Re: end moving to R3
p15	Robot in R5	t15	Cmd: start moving to R2
p16	Moving to R2	t16	Re: end moving to R2
p17	Moving to R5	t17	Cmd: start moving to R5
p18	Moving to R5	t18	Re: end moving to R5
p19	Moving to R4	t19	Cmd: start moving to R3
		t20	Re: end moving to R3
		t21	Cmd: start moving to R2
		t22	Re: end moving to R2
		t23	Cmd: start moving to R5
		t24	Re: end moving to R5
		t25	Cmd: start moving to R5
		t26	Re: end moving to R5
		t27	Cmd: start moving to R4
		t28	Re: end moving to R4

In the specification model, S-1 to S-4 are enforced by using the mutual exclusion theory with limited tokens. The composed PN model of both the systems are specifications is shown in Figure 8. The filtering places Pf1-Pf8 are drawn thicker and the filtering arcs are shown with dashed lines. A filtering place is modeled as an input place of the transitions that need such a resource, and as an output place of those transitions that release this resource. Take an example of Pf1 that physically means R1 being available. Because only one robot can work in R1 after the firing of the transition t5, it cannot be executed again until t6 is given to signal that R1 is available again. Thus, only one robot is allowed to be in R1 at any time, thereby avoiding any collision. The filtering places Pf1 to Pf8 (for S-1, S-2, S-3 and S-4) are used to prevent undesired human operations that lead to resource conflicts on the part of the system. The corresponding notation for the filtering places is described in Table 2. Note that the Pf1-Pf2-Pf5-Pf6 are one-bounded places, the Pf3-Pf4 are two-bounded, the Pf7 is three-bounded, and Pf8 is five-bounded, indicating the number of admitted robots.

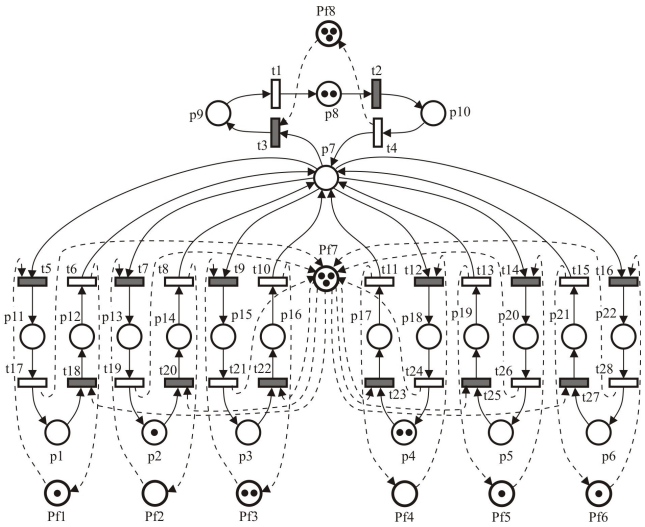

Fig. 8. *PN model with command filtering functions*

Table 2

Notification of the filtering places in Figure 8

Place	**Description**
Pf1	S 1.1: R1 admits one robot
Pf2	S 1.2: R2 admits one robot
Pf5	S 1.3: R5 admits one robot
Pf6	S 1.4: R6 admits one robot
Pf3 (2-bound)	S 2.1: R3 admits two robots
Pf4 (2-bound)	S 2.2: R4 admits two robots
Pf7 (3-bound)	S 3.1: R7 admits three robots
Pf8 (5-bound)	S 4.1: R8 admits five robots

3.4. Analysis and Verification

At this stage, due to its ease of manipulation, support for graphics import, and ability to perform structural and performance analysis, the software package MATLAB PN Toolbox [11] was chosen to verify the behavioral properties of the com-posed PN model using reachability analysis. The validation results reveal that the present PN model is deadlock-free, bounded, and reversible. The deadlock-free property means that the system can be executed properly without deadlocks (S-5), boundedness indicates that the system can be executed with limited resources, and reversibility implies that the initial system configuration is always reachable.

For the human-controlled robot in the proposed command filtering framework, the human commands are accepted or rejected to satisfy the specifications so that collisions are avoided during the surveillance period. As shown in Table 3, without command filtering, the state space is 1540 with undesirable collision states. By using the filtering approach, the state space is reduced to 413. Over 74% of states would be avoided during the surveillance period, i.e., improper actions that violate S-1 to S-5 and lead to these undesired states would be successfully filtered.

Table 3
Comparison of the unfiltered and filetered-frameworks

Petri net models	Places	Trans.	Arcs	State space
Unfiltered syst.	22	28	56	1540
Filtered syst.	30	28	82	413

In this approach, the command filter consists only of places and arcs, and its size is proportional to the number of specifications that must be satisfied. Thus, it is believed that the presented technique could be further applied to large-scale wireless MSNs. However, the specifications designed in this paper lead to limitations for office-like structured environments. New specifications for applications to unstructured environments and large-scale networks should be investigated in the future.

4. Conclusions

In this paper, a framework to develop a command filter for semiautonomous MSNs with the human-in-the-loop has been presented. The command filter is systematically designed and implemented using PN modeling and agent technology. Agents are implemented with JADE and ontologies are defined with Protégé 2000. To demonstrate the practicability of the proposed approach, an application to the mobile wireless surveillance system is illustrated. According to state acquisition via distributed P2P communication, the developed command filter ensures the specifications by accepting or rejecting human-issued commands.

References

1. Ahn, H.S., Ko, K.H.: *Simple Pedestrian Localization Algorithms Based on Distributed Wireless Sensor Networks*. In: IEEE Trans. Ind. Electr. **56** (2009) No. 10, p. 4296-4302.
2. Culler, D., Estrin, D., et al.: *Guest Editors's Introduction: Overview of Sensor Networks*. In: IEEE Computer **37** (2004) No. 8, p. 41-49.
3. Floroian, D.: *Sisteme Multiagent* (*Multiagent Systems*). Cluj-Napoca. Editura Albastră, 2009.
4. Grewal, M.S., Andrews, A.P.: *Kalman Filtering. Theory and Practice.* NJ, USA. Prentice Hall, Inc., Upper Saddle River, 1993.
5. Lee, D., Chung, W.: *Discrete-status-based Localization for Indoor Service Robots*. In: IEEE Transaction on Industrial Electronics **53** (2006) No. 5, p. 1737-1746.
6. Lee, J.S., Hsu, P.L.: *Remote Supervisory Control of the Human-in-the-loop System by Using Petri Nets and Java*. In: IEEE Trans. Ind. Electron. **50** (2003) No. 3, p. 431-439.
7. Lee, J.S., Zhou, M.C., et al.: *An Application of Petri Nets to Supervisory Control for Human-computer Interactive Systems*. In: IEEE Trans. Ind. Electron. **52** (2005) No. 5, p. 1220-1226.
8. Lee, J.S.: *A Command Filtering Framework to Collision Avoidance for Mobile Sensory Robots*. In: Proc. IEEE Int. Symp. Ind. Electron., Vigo, Spain, 4-7 June 2007, pg. 136-141.
9. Lee, J.S.: *A Petri Net Design of Command Filters for Semiautonomous Mobile Sensor Networks*. In: IEEE Trans. Ind. Electron. **55** (2008) No. 4, p. 1835-1841.
10. Low, K.H., Leow, W.K., et al.: *Autonomic Mobile Sensor Network with Self-coordinated Task Allocation*. In: IEEE Trans. Syst., Man, Cybern., C,

Appl. Rev. **36** (2006) No. 3, p. 315-327.
11. Pastravanu, O., Matcovschi, M., et al.: *Applications of Petri Nets in Studying Discrete Event Systems* (*in Romanian*). Iaşi. Editura Gh. Asachi, 2002.
12. Petriu, E., Whalen, T., et al.: *Robotic Sensor Agents: A New Generation of Intelligent Agents for Complex Environment Monitoring*. In: IEEE Instrumentation & Measurement Magazine **7** (2004) No. 3, p. 46-51.
13. Ramadge, P.J., Wonham, W.M.: *The Control of Discrete Event Systems*. In: Proc. IEEE **77** (1989) No. 1, p. 81-98.
14. Wooldridge, M., Jennings, N.R.: *Agent Theories, Architectures, and Languages: A Survey*. In: Proceedings of ECAI'94 - Workshop on Agent Theories Architectures and Languages, Amsterdam, The Netherlands, August 1994, p. 145-160.
15. Zheng, J., Lorenz, P., et al.: *Guest Editorial: Wireless Sensor Networking.* In: IEEE Network **20** (2006) No. 3, p. 4-5.
16. Zhou, M.C., DiCesare, F.: *Parallel and Sequential Mutual Exclusions for Petri Net Modeling for Manufacturing Systems*. In: IEEE Trans. Robot. Automat. **7** (1991), p. 515-527.
17. Zhou, M.C., Jeng, M.D.: *Modeling, Analysis, Simulation, Scheduling and Control of Semiconductor Manufacturing Systems: A Petri Net Approach*. In: IEEE Trans. Semicond. Manuf. **11** (1998) No. 3, p. 333-357.
18. *** Java Agent Development Framework - JADE. Available at: http://jade.tilab.com/. Accessed: 10-10-2009.
19. *** Protégé 2000. Available at: http://protege.stanford.edu/. Accessed: 12-10-2009.