Supervisory Control of Semiautonomous Mobile Sensor Networks: A Petri Net Design Approach

Florin MOLDOVEANU, Dan FLOROIAN, Dan PUIU

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 an filter agent for every robot. An application to a mobile wireless surveillance system is provided to show the feasibility of the developed approach. It is believed that the technique presented in this paper could be further applied to large-scale wireless mobile sensor networks.

Key words: agent, command filters, mobile robots, mobile sensor networks, Petri nets, supervisory control, wireless sensor networks

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 (e.g. disaster relief, environment monitoring) [10]. 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 [1, 7, 16]. 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. Mobile sensors can physically change the network topology, reacting to the events of the environment or to changes in the mission planning. References [3, 10] points out the advantages of sensor mobility and several algorithms for network self-organization after the occurrence of predetermined events are also proposed. Petriu et al. [13] have studied the networks of autonomous robotic sensor agents for active investigation of complex environments. However, most of the MSN literature focuses on a sensory system of fully-autonomous mobile robots without human intervention.

Dr. Florin Moldoveanu, dr. Dan Floroian, dr. Dan Puiu, all; Transilvania University of Brasov, Faculty of Electrical Engineering and Computer Science, Department of Automation

In real applications, human operators may use semiautonomous robots, as shown in *Figure 1 (a)* [9], 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. 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 attributed to human errors. However, the literature classifies human errors and provides few solutions for reducing or eliminating that possibility. Lee and Hsu [5] 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 [6]. PN has been developed into a powerful tool for modeling, analysis, control, optimization, and implementation of various engineering systems [8, 11, 18]. Lee and Chung [4] proposed a

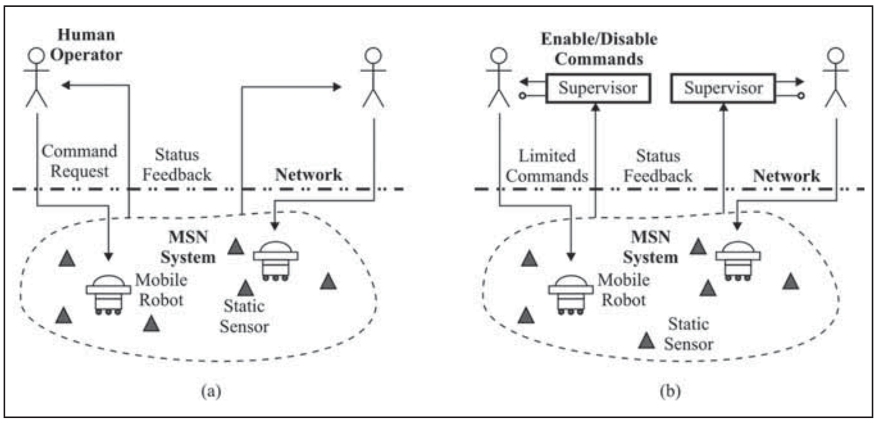


Figure 1. *(a) Human-involved MSNs. (b) Applied supervisory framework for such MSN systems*

PN-based localization scheme on a discrete event control framework for indoor service robots.

In practical applications, some requirements (typically for safety considerations) have to be obeyed for the overall system operations. Therefore, a supervisory framework is needed to facilitate the human control so as to guarantee that undesirable executions never occur.

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 analyzing DES [14]. *Figure 1(b)* adopts the supervisory framework [5, 6], 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]. The advantages of P2P include increased scalability (capacity scales with popularity), robustness (no single point of failure), fault tolerance, resilience to attack, and better support and management in distributed cooperative environments. Moreover, from a passive point of view, a command filter [8] 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 reject 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 modeling both the robots and the sensors and could interact with others to obtain perfect behaviors. Also with an user interface program, agents could also interact with human operators. Human operator could influent the decision process of the agents. There are also special agents for network integration [15].

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. 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 [5] and [6], except for the implementation framework as shown in Figures 1(b) and 2. To demonstrate the feasibility of the proposed filtering framework, an application to a mobile wireless surveillance system is illustrated in this paper. Du-

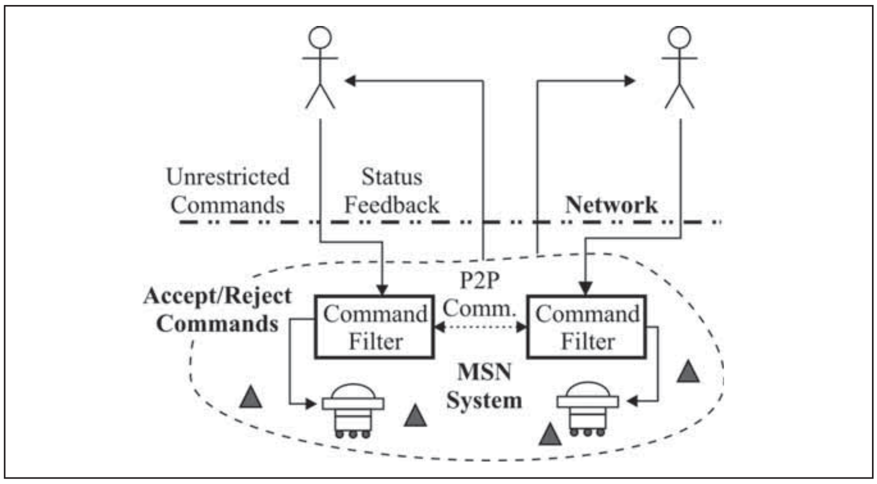


Figure 2. *Command filtering framework for MSNs*

ring system operation, our approach ensures that remote commands from the human operator meet the given collision-avoidance requirements. Note that the research work presented in this paper is conducted in an office-like environment.

The organization of the paper is as follows. Section 2 briefly introduces the PN-based modeling scheme. Next, a systematical design procedure of the command filter synthesis is described in Section 3. Then, in Section 4, an example of a mobile wireless surveillance system is illustrated to show the feasibility. Finally, Section 5 gives the conclusions.

2 Petri Net – Based System Modeling

Most existing methods for supervisory control system design are based on automata models. However, these methods often involve exhaustive searches of overall system behavior and result in state-space explosion problems. One way of dealing with these problems is to model the DES with PNs. PN modeling normally has more compact syntactical representation than the automata approach. Also, from a semnatic point of view, the effect of the state-space explosion problem can be reduced using the structural analysis to investigate the system properties. In addition, PN has an appealing graphical representation with a powerful algebraic formulation and is better suited for modeling systems with parallel and concurrent activities. This section first introduces the basic PN concept, and then shows the modeling of human-involved MSNs.

2.1 Basic Concepts of PN

A PN is identified as a particular kind of bipartite directed graph populated by three types of objects. They are places, transitions, and directed arcs connecting places and transitions. In order to study dynamic behavior of the modeled system, in terms of its states and their changes, each place may potentially hold either none or a positive number of tokens, pictured by small solid dots. The presence or absence of a token in a place can indicate whether a condition associated with this place is true or false, for instance. For a place representing the availability of resources, the number of tokens in this place indicates the number of available resources. At any given time instance, the distribution of tokens on places, called PN marking, defines the current state of the modeled system. Formally, a PN can be defined as [12]:

$$G=(P,T,I,O,M_0), \tag{1}$$

where: $P=\{p_1,p_2,\ldots,p_m\}$ is a finite set of places, where $m>0$; $T=\{t_1,t_2,\ldots,t_n\}$ is a finite set of transitions with $P\bigcup T\neq\varnothing$, and $P\bigcap T=\varnothing$, where $n>0$; $I:(P\times T)\rightarrow N$ is an input function that defines a set of directed arcs from P to T, where $N=\{0,1,2,\ldots\}$; $O:(T\times P)\rightarrow N$ is an output function which defines a set of directed arcs from T to P; $M_0:P\rightarrow N$ is the initial marking.

By changing distribution of tokens on places, which may reflect the occurrence of events or execution of operations, for instance, one can study dynamic behavior of the modeled system. The following rules are used to govern the flow of tokens:

Enabling Rule: A transition *t* is said to be enabled if each input place *p* of *t* contains at least the number of tokens equal to the weight of the directed arc connecting *p* to *t*.

Firing Rule:

a) An enabled *t* transition may or may not fire depending on the additional interpretation, and
b) A firing of an enabled transition *t* removes from each input place *p* the number of tokens equal to the weight of the directed arc connecting *p* to *t*. It also deposits in each output place *p* the number of tokens equal to the weight of the directed arc connecting *t* to *p*.

As a mathematical tool, a PN model can be described by a set of linear algebraic equations, or other mathematical models reflecting the behavior of the system. This opens a possibility for the formal analysis of the model. This allows one to perform a formal check of the properties related to the behavior of the underlying system, e.g., precedence relations amongst events, concurrent operations, appropriate synchronization, freedom from deadlock, repetitive activities, and mutual exclusion of shared resources, to mention some.

Some important PN properties include a boundedness (no capacity overflow), liveness (freedom from deadlock), conservativeness (conservation of nonconsumable resources), and reversibility (cyclic behavior). The concept of liveness is closely related to the complete absence of deadlocks. A PN is said to be live if, no matter what marking has been reached from the initial marking, it is possible to ultimately fire any transition of the net by progressing through further firing sequences. This means that a live PN guarantees deadlock-free operation regardless of the firing sequence. Validation methods of these properties include reachability analysis, invariant analysis, reduction method, siphons/traps-based approach, and simulation [12, 18].

2.2 Modeling of Semiautonomous MSNs

PNs have been used to model, analyze, and synthesize control laws for DES. Zhou and DiCesare [17], moreover, addressing the shared resource problem recognized that mutual exclusion theory plays a key role in synthesizing a bounded, live, and reversible PN. In mutual exclusion theory, parallel mutual exclusion consists of a place marked initially with one token to model a single shared resource, and a set of pairs of transitions. Each pair of transitions models a unique operation which requires the use of the shared resource. In this paper, we adopt mutual exclusion theory to build the PN specification model. 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 pla-

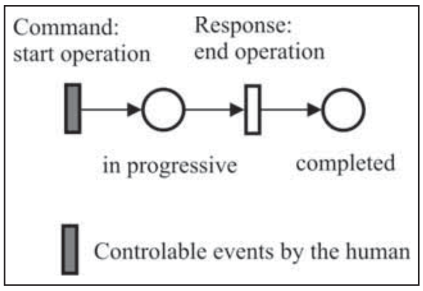


Figure 3. *Modeling of human behavior using the command/response concept*

ce. 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.

3 Petri Net-Based Command Filter Design

3.1 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 avoid 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.

The liveness of a PN is closely related to the complete absence of deadlocks. A PN is said to be live if, no matter what marking has been reached from the initial marking, it is possible to ultimately fire any transition of the net by progressing through further firing sequences. This means that a live PN guarantees deadlock-free operation regardless of the firing sequence.

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

3.2 Synthesis of Command Filters

Definition 3.1: Considering two Petri nets $G_1 = (P_1, T_1, I_1, O_1, M_{01})$ and $G_2 = (P_2, T_2, I_2, O_2, M_{02})$, the synchronous composition of two marked Petri nets G_1 and G_2 is a net $G = (P, T, I, O, M_0)$:

$$G = G_1 \otimes G_2, \tag{2}$$

where: $P = P_1 \cup P_2$; $T = T_1 \cup T_2$; $I(p,t) = I_i(p,t)$ if $(\exists i \in \{1,2\})$, $[p \in P_i \wedge t \in T_i]$, else $I(p,t) = 0$; $O(p,t) = O_i(p,t)$; if $(\exists i \in \{1,2\})$, $[p \in P_i \wedge t \in T_i]$, else $O(p,t) = 0$; $M_0(p) = M_{01}(p)$ if $p \in P_1$, else $M_0(p) = M_{02}(p)$.

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*. For a system with plant model G and specification model H, the filter can be obtained by synchronous composition of the plant and specification models:

$$F = G \otimes H, \tag{3}$$

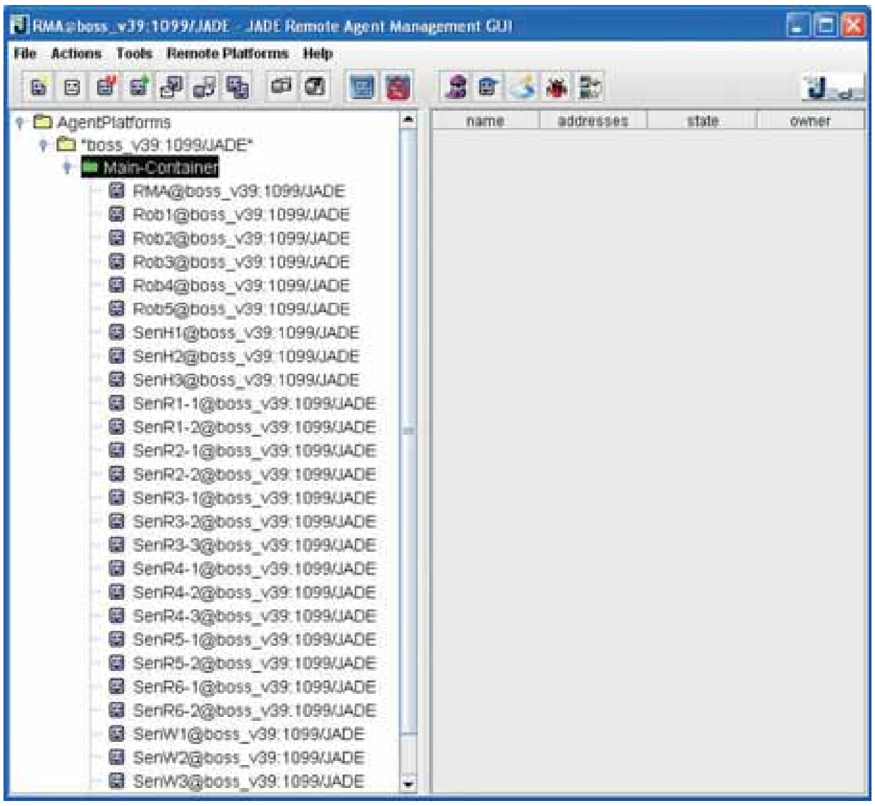
where the transitions of H are a subset of the transitions of G, i.e., $T_H \subset T_G$. Note that F obtained through the above construction, in the general case, does not represent a proper filter, since it may contain deadlock states from which a final state cannot be reached. Thus, the behavior of F should be further refined and restricted by PN analysis. The design procedure of PN-based command filters consists of the following steps:

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

3.3 Implementation Using Agent Technology

Agent technology is a new and important technique in recent novel researches of artificial intelligence. Using agent technology leads to a number of advantages such as scalability, event-driven actions, task-orientation, and adaptivity [2]. The concept of an agent as a computing entity is very dependent on the application domain in which it operates. As a result, there exist many definitions and theories on what actually constitutes an agent and the sufficient and necessary conditions for agency. Wooldridge and Jennings [15] 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. Agents are similar to software objects but they must run continuosly and autonomously. 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. This properties confere inteligence to the agents and togheter they are able to execute tasks that individualy they don't have skills. For command filtering this represent a very important quality that make 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 aquire the system status by autonomously sensing and the P2P (peer to peer) 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 [19] because this development tools is very versatile and could be very well integrated with others development tools (like Protégé-2000 and Java [20]). Also JADE is an open source FIPA (Foundation for Intelligent Physical Agents) compliant Java based software framework for the implementation of multiagent systems. It simplifies the implementation of agent communities by offering runtime and agent programming libraries, as well as tools to manage platform execution and monitoring and debugging activities. These supporting tools are themselves FIPA agents. JADE offers simultaneously middleware for FIPA compliant multi-agent systems, supporting application agents whenever they need to exploit some feature covered by the FIPA standard (message passing, agent life cycle, etc), and a Java framework for developing FIPA compliant agent applications, making FIPA standard assets available to the programmer through Java object-oriented abstractions. 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). The agent platform can be split between several hosts (provided that there is no firewall between them). Agents are implemented as one Java thread and Java events are used for effective and lightweight communication between agents on the same host. Parallel tasks can be still executed by one agent, and JADE schedules these tasks in a more efficient (and even simpler for the skilled programmer) way than the Java Virtual Machine does for threads. Several Java Virtual Machines (VM), called containers in JADE, can coexist in the same agent platform even though they are not running in the same host as the RMA agent. This means that a RMA can be used to manage a set of VMs distributed across various hosts. Each container provides a complete run time environment for agent execution and allows several agents to concurrently execute on the same host. The DF, AMS, and RMA agents coexist under the same container (main-container) togheter with the filtering agents, as it is shown in *Figure 4*.


```
RMA@boss_v39:1099/JADE - JADE Remote Agent Management GUI
File  Actions  Tools  Remote Platforms  Help

AgentPlatforms
  "boss_v39:1099/JADE"
    Main-Container
      RMA@boss_v39:1099/JADE
      Rob1@boss_v39:1099/JADE
      Rob2@boss_v39:1099/JADE
      Rob3@boss_v39:1099/JADE
      Rob4@boss_v39:1099/JADE
      Rob5@boss_v39:1099/JADE
      SenH1@boss_v39:1099/JADE
      SenH2@boss_v39:1099/JADE
      SenH3@boss_v39:1099/JADE
      SenR1-1@boss_v39:1099/JADE
      SenR1-2@boss_v39:1099/JADE
      SenR2-1@boss_v39:1099/JADE
      SenR2-2@boss_v39:1099/JADE
      SenR3-1@boss_v39:1099/JADE
      SenR3-2@boss_v39:1099/JADE
      SenR3-3@boss_v39:1099/JADE
      SenR4-1@boss_v39:1099/JADE
      SenR4-2@boss_v39:1099/JADE
      SenR4-3@boss_v39:1099/JADE
      SenR5-1@boss_v39:1099/JADE
      SenR5-2@boss_v39:1099/JADE
      SenR6-1@boss_v39:1099/JADE
      SenR6-2@boss_v39:1099/JADE
      SenW1@boss_v39:1099/JADE
      SenW2@boss_v39:1099/JADE
      SenW3@boss_v39:1099/JADE

name | addresses | state | owner
```

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

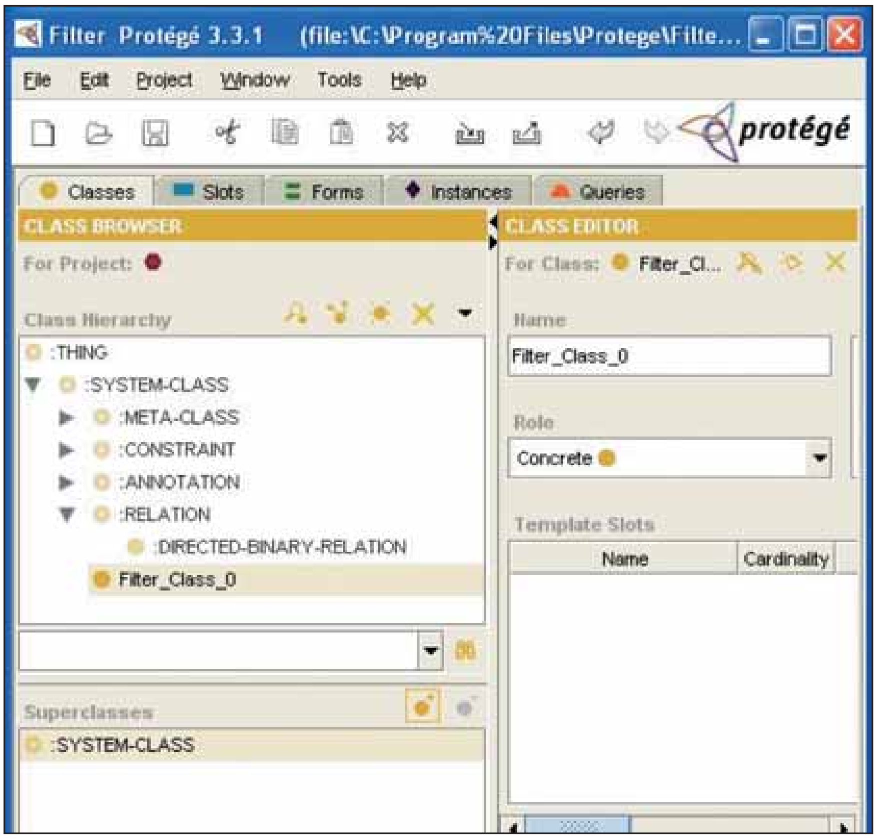


Figure 5. *Defining ontologies for filtering class, using Protégé-2000*

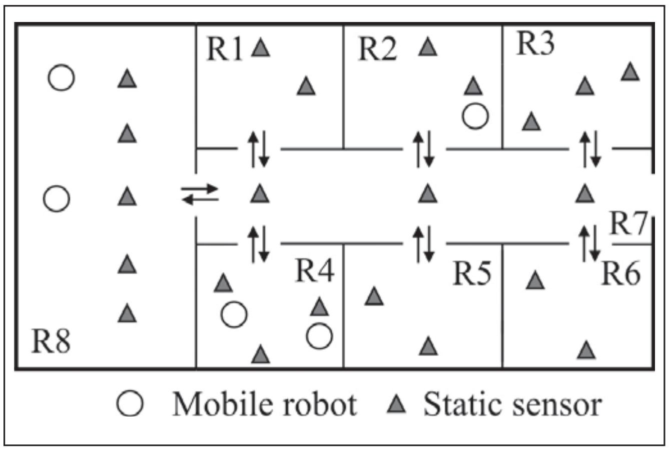


Figure 6. *Mobile wireless surveillance system with five robots*

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

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). Each room is equipped with a sensing and communication device, such as an RFID (Radio-Frequency IDentification) 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.

■ 4 Example: A Mobile Wireless Surveillance System

4.1 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*. 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.

4.2 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*.

4.3 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:

Spec-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 Spec-1.1 to Spec-1.4.

Spec-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 Spec-2.1 to Spec-2.2.

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

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

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

In the specification model, Spec-1 to Spec-4 are enforced by using the mutual exclusion theory with limited tokens. The composed PN model of

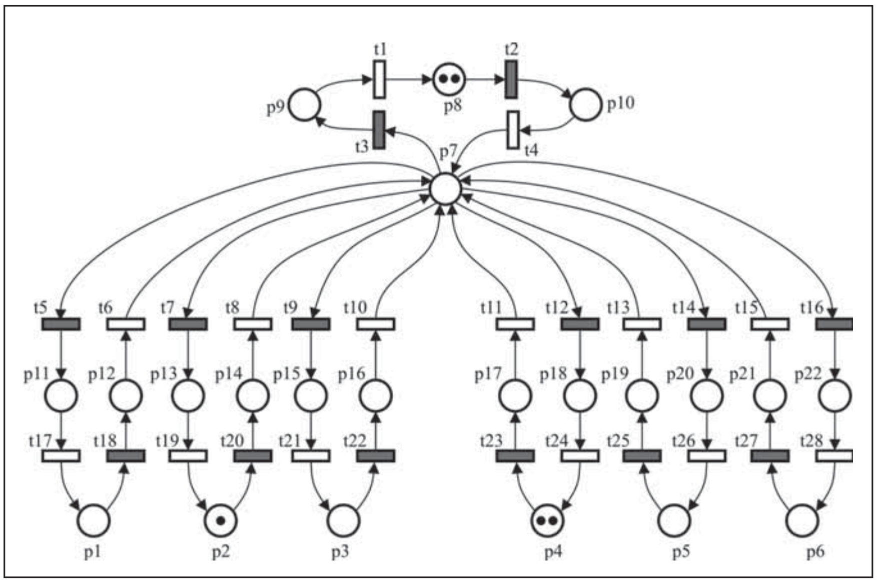


Figure 7. *PN model of the human-controlled mobile robots*

Table 1. *Notation of the PN places and transitions in Figure 7*

Place	Description	Place	Description
p1	Robot in R1	p12	Moving to R7
p2	Robot in R2	p13	Moving to R2
p3	Robot in R3	p14	Moving to R7
p4	Robot in R4	p15	Moving to R3
p5	Robot in R5	p16	Moving to R7
p6	Robot in R6	p17	Moving to R7
p7	Robot in R7	p18	Moving to R4
p8	Robot in R8	p19	Moving to R7
p9	Moving to R8	p20	Moving to R5
p10	Moving to R7	P21	Moving to R7
p11	Moving to R1	p22	Moving to R6
t1	Re: end moving to R8	t15	Re: end moving to R7
t2	Cmd: start moving to R7	t16	Cmd: start moving to R6
t3	Cmd: start moving to R8	t17	Re: end moving to R1
t4	Re: end moving to R7	t18	Cmd: start moving to R7
t5	Cmd: start moving to R1	t19	Re: end moving to R2
t6	Re: end moving to R7	t20	Cmd: start moving to R7
t7	Cmd: start moving to R2	t21	Re: end moving to R3
t8	Re: end moving to R7	t22	Cmd: start moving to R7
t9	Cmd: start moving to R3	t23	Cmd: start moving to R7
t10	Re: end moving to R7	t24	Re: end moving to R4
t11	Re: end moving to R7	t25	Cmd: start moving to R7
t12	Cmd: start moving to R4	t26	Re: end moving to R5
t13	Re: end moving to R7	t27	Cmd: start moving to R7
t14	Cmd: start moving to R5	t28	Re: end moving to R6

Table 2. *Notation of the filtering places in Figure 8*

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

The filtering places Pf1 to Pf8 (for Spec-1, Spec-2, Spec-3 and Spec-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*.

4.4 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 composed 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 (Spec-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

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 modelled 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 Room 1 being available. Because only one robot can work in Room 1 after the firing of the transition t5, it cannot be executed again until t6 is given to signal that Room 1 is available again. Thus, only one robot is allowed to be in Room 1 at any time, thereby avoiding any collision.

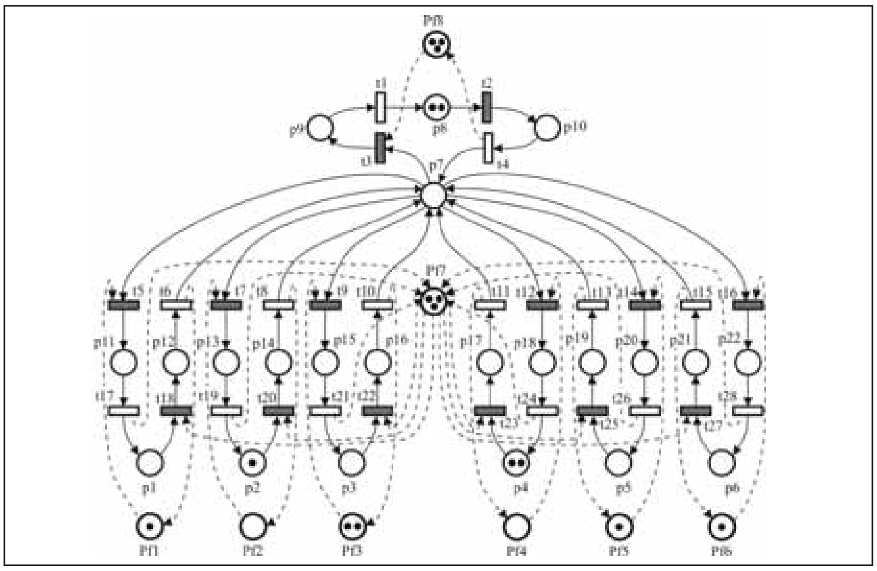


Figure 8. *PN model with command filtering functions*

Table 3. *Comparison of the unfiltered and filtered frameworks*

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

is reduced to 413. Over 74% of states would be avoided during the surveillance period, i.e., improper actions that violate Spec-1 to Spec-5 and lead to these undesired states would be successfully filtered.

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.

5 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.

Compared with previous work [5, 6] which, from an active viewpoint, sought to enable or disable actions leading to limited human commands, this paper has proposed another scheme, from a passive viewpoint, to accept or reject the actions as the robot receives the human commands. Hence, in the proposed filtering framework, human operators could request unrestricted commands, and the command filters would make real-time decisions based on an event-trigger and on-demand P2P networking. Future work will attempt to integrate both the active supervisor and passive filter into a single framework to provide a double protection scheme for semiautonomous MSNs.

Sources

[1] Culler, D., Estrin, D., Srivastava, M.: Guest Editors's Introduction: Overview of Sensor Networks, *IEEE Computer*, Vol. 37, No. 8, Aug. 2004, pg.:41–49 .

[2] Floroian, D.: Multiagent Systems, Cluj-Napoca (Romania), Ed. Albastra (In Romanian), 2009.

[3] Giordano, V., Ballal, P., Lewis, F., Turchiano, B., Zhang, J.B.: Supervisory Control of Mobile Sensor Networks: Math Formulation, Simulation and Implementation, *IEEE Trans. Syst., Man., Cybern. B, Cybern.*, Vol. 36, No.4, Aug. 2006, pg.:806–816.

[4] Lee, D., Chung, W.: Discrete-status-based Localization for Indoor Service Robots, *IEEE Trans. Ind. Electron.* Vol. 53, No. 5, Oct. 2006, pg.:1737–1746.

[5] Lee, J.S., Hsu, P.L.: Remote Supervisory Control of the Human-in-the-loop System by Using Petri Nets and Java, *IEEE Trans. Ind. Electron.* Vol. 50, No. 3, Jun. 2003, pg.:431–439.

[6] Lee, J.S., Zhou, M.C., Hsu, P.L.: An Application of Petri Nets to Supervisory Control for Human-computer Interactive Systems, *IEEE Trans. Ind. Electron.* Vol. 52, No. 5, Oct. 2005, pg.:1220–1226.

[7] Lee, J.S., Huang, Y.C.: ITRI ZB-node: A ZigBee/IEEE 802.15.4 Platform for Wireless Sensor Networks, *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Taipei (Taiwan), Oct. 2006, pg.:1462–1467.

[8] Lee, J.S.: A Command Filtering Framework to Collision Avoidance for Mobile Sensory Robots, *Proc. IEEE Int. Symp. Ind. Electron.*, Vigo (Spain), Jun. 2007, pg.:136–141.

[9] Lee, J.S.: A Petri Net Design of Command Filters for Semiautonomous Mobile Sensor Networks, *IEEE Trans. Ind. Electron.*, Vol. 55, No.4, April 2008, pg.:1835–1841.

[10] Low, K.H., Leow, W.K., Ang, M.H.: Autonomic Mobile Sensor Network with Self-coordinated Task Allocation, *IEEE Trans. Syst., Man, Cybern.*, C, Appl. Rev., Vol. 36, No.3, May 2006, pg.:315–327.

[11] Matlab, 2008. Petri Nets Toolbox.

[12] Pastravanu, Oc., Matcovschi, M., Mahulea, C.: Applications of Petri Nets in Studying Discrete Event Systems, Iasi (Romania), Ed. Gh. Asachi (In Romanian), 2002.

[13] Petriu, E., Whalen, T., Abielmona, R., Stewart, A.: Robotic Sensor Agents: A New Generation of Intelligent Agents for Complex Environment Monitoring, *IEEE Instrum. Meas. Mag.*, Vol. 7, No.3, Sep. 2004, pg.:46–51.

[14] Ramadge, P.J., Wonham, W.M.: The Control of Discrete Event Systems, *Proc. IEEE*, Vol. 77, Jan. 1989, pg.:81–98.

[15] Wooldridge, M., Jennings, N.R.: Agent Theories, Architectures, and Languages: A Survey, *Proc. ECAI-Workshop on Agent Theories, Architectures and Languages*, Amsterdam (The Netherlands), Aug. 1994, pg.:145–160.

[16] Zheng, J., Lorenz, P., Dini, P.: Guest Editorial: Wireless Sensor Networking, *IEEE Netw.*, Vol. 20, No. 3, May/Jun. 2006, pg.:4–5.

[17] Zhou, M.C., DiCesare, F.: Parallel and Sequential Mutual Exclusions for Petri Net Modeling for Manufacturing Systems, *IEEE Trans. Robot. Automat.*, Vol. 7, Aug. 1991, pg.:515–527.

[18] Zhou, M.C., Jeng, M.D.: Modeling, Analysis, Simulation, Scheduling and Control of Semiconductor Manufacturing Systems: A Petri Net Approach, IEEE Trans. Semicond. Manuf., Vol. 11, No. 3, Aug. 1998, pg.:333–357.

[19] ***, Java Agent Development Framework – JADE, http://jade.tilab.com/

[20] ***, Protégé 2000, http://protege.stanford.edu/

Nadzorno vodenje semiavtonomnih mobilnih senzorskih omrežij: pristop z načrtovanjem Petrijevih mrež

Razširjeni povzetek

Pri obravnavanih semiavtomatskih mobilnih senzorskih omrežjih se lahko pri posebno neugodnih akcijah zgodijo nesreče ali celo katastrofe, saj so v regulacijsko zanko lahko vključeni tudi operaterji. V prispevku za takšne sisteme predlagamo ustrezen način filtriranja, ki sprejme ali zavrne človeške ukaze, tako da nikoli ne pride do izvajanja neželenih oz. neprimernih ukazov. V predstavljenem pristopu smo uporabili Petrijeve mreže za modeliranje obnašanja operaterjev in za sintezo ukazov filtriranja za nadzor. Nadzorni filter je mogoče implementirati tudi ob uporabi agentne tehnologije za vsakega od robotov. Opisana je realizacija brezžičnega mobilnega sistema, da bi ilustrirali izvedljivostne možnosti razvitega pristopa. Verjamemo, da je predstavljeno tehniko mogoče uporabiti tudi na velikih brezžičnih mobilnih senzorskih omrežjih.

Ključne besede: agent, ukazni filtri, mobilni roboti, mobilna senzorska omrežja, Petrijeve mreže, nadzorno vodenje, brezžična senzorska omrežja

FLUIDNA TEHNIKA - AVTOMATIZACIJA - INDUSTRIJSKA OPREMA

Hypex

INDUSTRIJSKA PNEVMATIKA

cilindri,enote za vodenje,prijemala,ventili,priprava zraka,fitingi,spojke,cevi in pribor

MERILNA TEHNIKA
IN SENZORIKA

senzorji in merilci sile,temperature,tlaka,magnetnega polja ter indukcijski senzorji

PROCESNA TEHNIKA

krogelni in loputasti ventili,ploščati zasuni,pnevmatski in električni pogoni,varnostni ventili

LINEARNA TEHNIKA

tirna vodila,okrogla vodila,kroglična vretena,blažilci sunkov,regulatorji hitrosti

PROFILNA TEHNIKA
IN STROJEGRADNJA

konstrukcijski alu profili,delovna oprema,ogrodja strojev

STORITVE

konstrukcija in obdelave na klasičnih in CNC strojih

-TRADICIJA
-KVALITETA
-SVETOVANJE
-PARTNERSTVO
-FLEKSIBILNOST
-VELIKE ZALOGE
-POSEBNE IZVEDBE
-KONKURENČNE CENE
-KRATKI DOBAVNI ROKI

Hypex, Lesce, d.o.o.
Alpska 43, 4248 Lesce
Tel.: +386(0)4 53-18-700 Internet: www.hypex.si
Fax.: +386(0)4 53-18-740 E-Mail: info@hypex.si