An Application of Petri Nets to Supervisory Control for Human–Computer Interactive Systems

Jin-Shyan Lee, Meng-Chu Zhou, *Fellow, IEEE*, and Pau-Lo Hsu, *Member, IEEE*

***Abstract*—In a distributed robotic system, both human-controlled (semi-autonomous) and computer-controlled (fully autonomous) robots may simultaneously exist. From the global system's point of view, supervisory control for the interactions between the human and computer are important and necessary. For such human–computer interactive systems, this paper proposes a supervisory framework to guarantee that both human and computer commands meet collision-free and deadlock-free requirements. In the presented approach, Petri nets are applied to construct a system model and synthesize a desired supervisor. An application to a two-robot remote surveillance system is provided to demonstrate the practicability of the developed supervisory control approach. It is believed that the technique developed in this paper is significant in the industrial practice.**

***Index Terms*—Distributed robotic systems, human–computer interactive systems, Petri nets (PNs), remote surveillance systems, supervisory control.**

I. Introduction

RECENTLY, mechatronics has been developed worldwide into a very attractive research area. It combines in a synergistic way the classical engineering disciplines of mechanical and electrical engineering and computer science, leading to new kinds of products. One of the topics in mechatronics is the investigation of the telerobotics because the demand for deploying distributed robotic systems in real-world environments has increased immensely. Telerobotics is the extension of a human's sensing and manipulation capability to a remote location. Application examples are in public safety surveillance, clean-up work in hazardous environments, and rescue work in disaster areas [1]–[3]. It is observed that it is impossible for those robots to carry out all the high-level tasks autonomously. A human operator would be somehow involved in these systems. In general, both autonomous and semi-autonomous robots may simultaneously exist in a distributed robotic system, where the former are fully controlled by associated computer controllers, while the latter are also partly controlled by humans. Thus, a human operator has to interact with the computer controller in such a system, i.e., the so-called human–computer interactive system (HCIS) in this paper. As shown in Fig. 1(a), a human operator issues a command to trigger a human-controlled (semi-autonomous) robot and a computer controller automatically regulates a computer-controlled (autonomous) robot both with the status feedback from the overall controlled system (i.e., both robots) through a network. From the global system's point of view, interactions between human and computer are important and necessary. Also, 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 interactive control between human and computer so as to guarantee that undesirable executions never occur. However, most of the human–robot literature focuses on the interaction between human operators and their controlled robots [4]–[8], and provides few solutions for such an interaction between a human operator and an autonomous computer controller at the same control level.

From the high-level point of view, an HCIS 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 [9]–[11]. For such human–computer interactive systems, this paper proposes a supervisory framework so as to prevent abnormal operations of both human and computer from being carried out, as shown in Fig. 1(b). The supervisory agent acquires the system status and then advises the human operator and computer controller while issuing commands. The agent enables and disables the associated actions so as to meet the requirements. Then, both human operator and computer controller are allowed only to perform the enabled events to control their robots. The role of the supervisory agent is to interact with the human operator, computer controller, and controlled system so that the system meets the required specifications and to guarantee that undesirable executions do not occur. Thus, the human–computer interactive loop is closed in this way.

Most existing methods for supervisory 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 Petri nets (PNs) [12], [13]. PN modeling normally has more compact syntactical representation than the automata approach. Also, from a semantic point of view, the effect of the state-space explosion problem can be reduced using the structural analysis, such as the siphons/traps-based approach [16], to investigate the

Manuscript received March 3, 2004; revised August 13, 2004. Abstract published on the Internet July 15, 2005. This work was supported in part by the National Science Council, R.O.C., under Grant NSC 92-2917-I-009-005 and in part by the MOE Program for Promoting Academic Excellence of Universities under Grant 91-E-FA06-4-4.

J.-S. Lee was with the Department of Electrical and Control Engineering, National Chiao-Tung University, Hsinchu 30010, Taiwan, R.O.C. He is now with the Computer and Communications Research Laboratory, Industrial Technology Research Institute, Hsinchu 31040, Taiwan, R.O.C. (e-mail: jinshyan_lee@itri.org.tw).

M.-C. Zhou is with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07012 USA (e-mail: zhou@njit.edu).

P.-L. Hsu is with the Department of Electrical and Control Engineering, National Chiao-Tung University, Hsinchu 30010, Taiwan, R.O.C. (e-mail: plhsu@cc.nctu.edu.tw).

Digital Object Identifier 10.1109/TIE.2005.855694

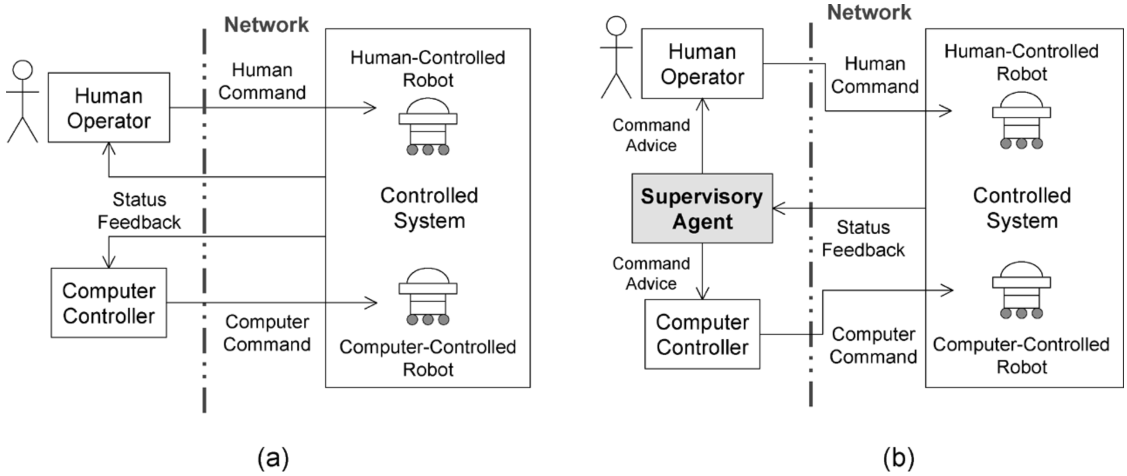

Fig. 1. (a) Basic human–computer interactive system. (b) Proposed supervisory framework for the system.

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. Thus, it has generated intense interest among many researchers [14]–[18]. In our work, PNs are used in designing the supervisory system, yielding a compact and graphical model for the HCIS. To demonstrate the feasibility of the proposed supervisory framework, an application of a remote surveillance system is illustrated in this paper. During the system operation, our approach ensures that remote commands from the human operator and computer controller meet the given collision-free and deadlock-free requirements.

The organization of the paper is as follows. Section II introduces the model construction of human–computer interactive systems by using PNs. Next, a systematical procedure of the PN-based supervisor synthesis is described in Section III. Then, Section IV illustrates our approach through a remote surveillance system. Finally, Section V gives the conclusions.

II. PN-Based System Modeling

This section first introduces the basic PN concept, and then shows the elementary PN models. Finally, the modeling of human–computer interactions is introduced.

A. 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. Formally, a PN can be defined as

$$\mathrm{PN} = (P, T, I, O, M_0)$$

where

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

A transition t is 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. When an enabled transition fires, it removes the tokens from its input places and deposits them on its output places. PN models are suitable to represent the systems that exhibit concurrency, conflict, and synchronization.

Some important PN properties include boundness (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 some further firing sequences. This means that a live PN guarantees deadlock-free operation, no matter what firing sequence is chosen [14]. Validation methods of these properties include reachability analysis, invariant analysis, reduction method, siphons/traps-based approach, and simulation [16].

B. Elementary PN Models

At the modeling stage, one needs to focus on the major operations and their sequential or precedent, concurrent, or conflicting relationships. The basic relations among these processes or operations can be classified as follows.

1) *Sequential*: As shown in Fig. 2(a), if one operation follows the other, then the places and transitions representing them should form a cascade or sequential relation in PN's.
2) *Concurrent*: If two or more operations are initiated by an event, they form a parallel structure starting with a transition, i.e., two or more places are the outputs of a same transition. An example is shown in Fig. 2(b). The pipeline concurrent operations can be represented with a sequentially-connected series of places/transitions in which multiple places can be marked simultaneously or multiple transitions are enabled at certain markings.
3) *Cyclic*: As shown in Fig. 2(c), if a sequence of operations follow one after another and the completion of the last one initiates the first one, then a cyclic structure is formed among these operations.

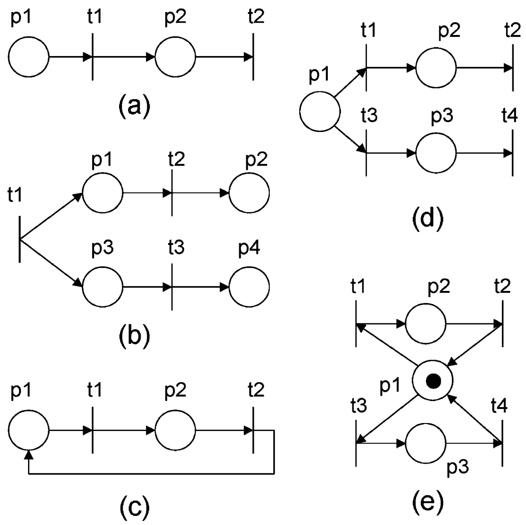

Fig. 2. Basic PN models for (a) sequential, (b) concurrent, (c) cyclic, (d) conflicting, and (e) mutually exclusive relations.

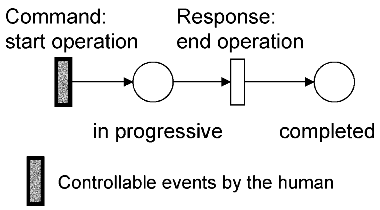

Fig. 3. Modeling of human behavior using the command/response concept.

4) *Conflicting*: As shown in Fig. 2(d), if either of two or more operations can follow an operation, then two or more transitions form the outputs from the same place.
5) *Mutually Exclusive*: As shown in Fig. 2(e), two processes are mutually exclusive if they cannot be performed at the same time due to constraints on the usage of shared resources. A structure to realize this is through a common place marked with one token plus multiple output and input arcs to activate these processes.

In this paper, for the human–computer interactive systems, PN models of the human behavior and computer actions will be constructed based on these elementary models.

C. Modeling of Human–Computer Interactions

On the part of the human-controlled system, the human behavior can be modeled using the command/response concept. As shown in Fig. 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 that are controllable by the remote-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, the computer-controlled action can be simply modeled as a single event transition, since its actions are all uncontrollable from the human-controlled point of view.

III. PN-Based Supervisor Synthesis

This section first shows the required specification types for human–computer interactive systems. Then, a PN-based procedure for supervisor synthesis is introduced.

A. Specification Types

The objective of a supervisor is to restrict the behavior of both human and computer so that it is contained within the set of admissible states, called the specification. In this study, two main types of specifications are considered and described as follows.

1) *Collision-free motions*: This specification presents the physical constraints of the limited resources, such as the rooms and hallways. Two robots are disallowed to enter the same space to guarantee no collision. The shared resources can be adequately expressed in terms of mutual exclusion conditions as mentioned in Fig. 2(e).
2) *Deadlock-free operations*: The deadlock-free specification ensures that a given command will not lead to the system to a deadlock state. At such a state, no further action is possible. This specification can be preserved by deadlock avoidance policies [18].

On the part of the human-controlled system, the proposed supervisor enforces these specifications by restricting the commands available to human operators. On the other hand, the supervisor prohibits undesirable actions of the computer-controlled system so as to meet these specifications.

B. Supervisor Synthesis

PN's have been used to model, analyze, and synthesize control laws for DES. Zhou and DiCesare [19], moreover, addressing the shared resource problem, recognized that mutual exclusion theory plays a key role in synthesizing a live, bounded, 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 task which requires the use of the shared resource. In this paper, we adopt mutual exclusion theory to build the PN models for the collision-free specifications and then compose them with the models of human and computer behaviors to synthesize the supervisor. The supervisor design procedure consists of the following steps.

Step 1) Construct the PN model of the human behaviors and computer actions.
Step 2) Build the PN model of the collision-free specifications using the mutual exclusion concept for shared resources.
Step 3) Compose the behavior and resource models to synthesize the preliminary supervisor model.
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.

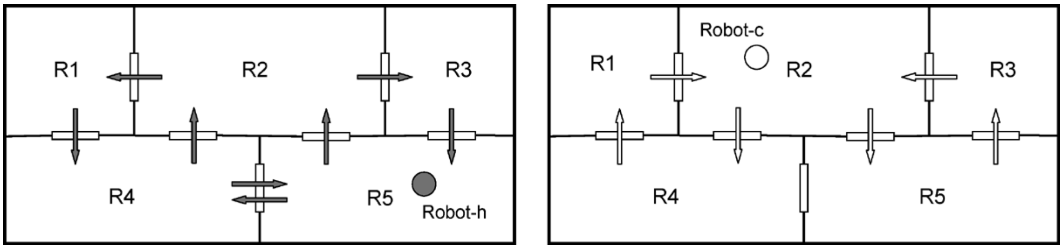

Fig. 4. Schematic diagram of the two-robot remote surveillance system with the moving directions for human-controlled robot (left) and computer-controlled robot (right).

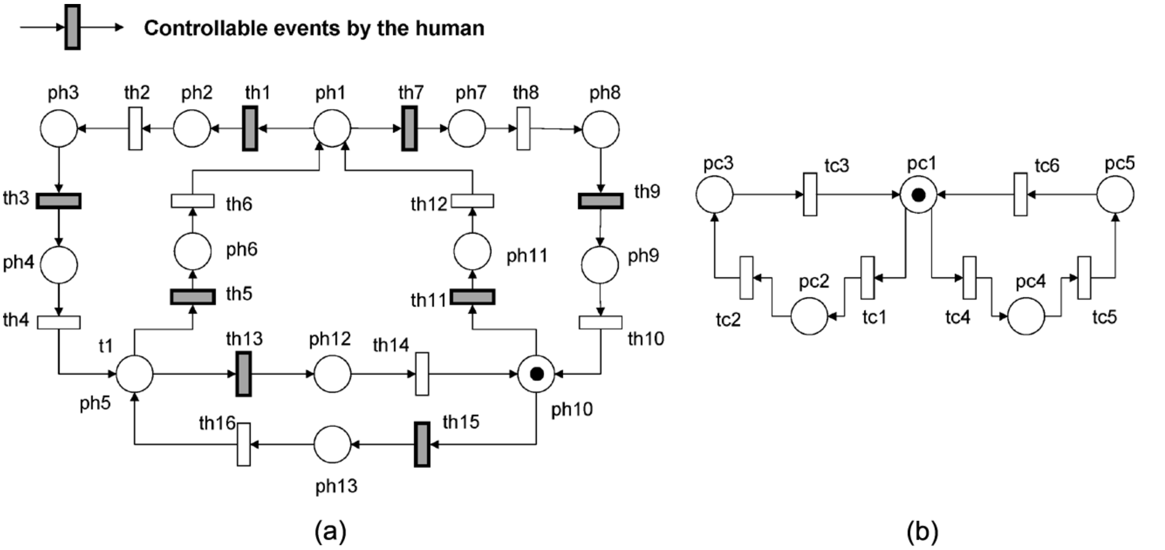

Fig. 5. PN models of (a) human-controlled robot and (b) computer-controlled robot.

IV. Supervisory Control of a Remote Surveillance System

A. System Description

The human–computer interactive system in Fig. 1(b) can be applied as a remote surveillance system, which is composed of one human-controlled robot (simplified as Robot-h) and one computer-controlled robot (simplified as Robot-c). These two robots are placed on a floor with five rooms, and the moving directions for each robot are shown in Fig. 4, respectively. The Robot-h and Robot-c must traverse each doorway in the direction indicated. Moreover, in order to avoid possible collisions, Robot-h and Robot-c are not allowed simultaneously in the same room during the surveillance period. The initial states of the Robot-h and Robot-c are in R5 and R2, respectively.

B. PN-Based System Modeling

By applying the command/response concept and based on the system description, the PN model for the human-controlled robot is constructed as shown in Fig. 5(a). It consists of 13 places and 16 transitions, respectively. On the other hand, for the computer-controlled robot, the PN model is directly built according to its located room, as shown in Fig. 5(b), which, respectively, consists of five places and six transitions. Corresponding notation of both the PN models is described in Table I.

C. PN-Based Supervisor Synthesis

The five rooms represent the resources shared by the two robots. Since more than one robot may require access to the same room, but in order to avoid collisions, each room can only be allowed to have one robot at a time, collisions and deadlocks may thus occur. Hence, the objective is to design a supervisor to insure the whole system against these undesired situations. The required two main specifications are formulated as follows.

TABLE I
Notation for the PNs of the Robots in Fig. 5

Place	Description	Transition	Description
ph1	Robot-h is in R2	th1	Cmd: start moving to R1
ph2	Moving to R1	th2	Re: end moving to R1
ph3	Robot-h is in R1	th3	Cmd: start moving to R4
ph4	Moving to R4	th4	Re: end moving to R4
ph5	Robot-h is in R4	th5	Cmd: start moving to R2
ph6	Moving to R2	th6	Re: end moving to R2
ph7	Moving to R3	th7	Cmd: start moving to R3
ph8	Robot-h is in R3	th8	Re: end moving to R3
ph9	Moving to R5	th9	Cmd: start moving to R5
ph10	Robot-h is in R5	th10	Re: end moving to R5
ph11	Moving to R2	th11	Cmd: start moving to R2
ph12	Moving to R5	th12	Re: end moving to R2
ph13	Moving to R4	th13	Cmd: start moving to R5
		th14	Re: end moving to R5
		th15	Cmd: start moving to R4
		th16	Re: end moving to R4
pc1	Robot-c is in R2	tc1	Move to R4
pc2	Robot-c is in R4	tc2	Move to R1
pc3	Robot-c is in R1	tc3	Move to R2
pc4	Robot-c is in R5	tc4	Move to R5
pc5	Robot-c is in R3	tc5	Move to R3
		tc6	Move to R2

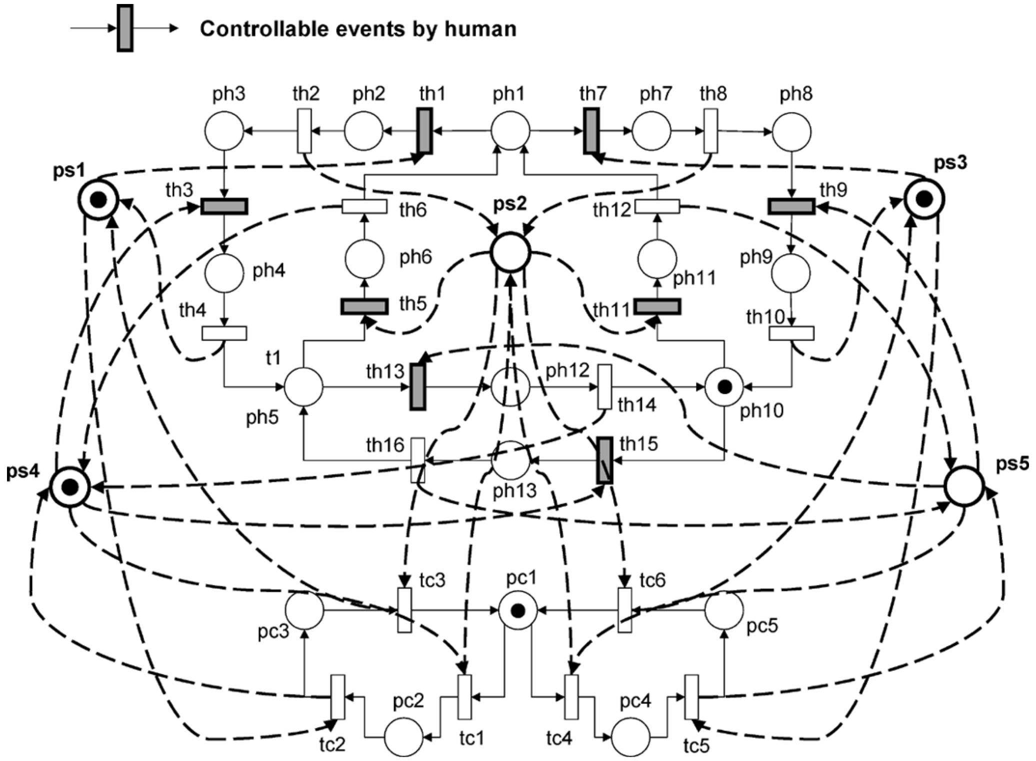

Fig. 6. Preliminary composed PN model of the remote surveillance system.

Spec-1) *Collision-free motions*: Robot-h or Robot-c moving to Room i is allowed only when Room i is available, where $i = 1, 2, \ldots, 5$. Thus, we have five subspecifications denoted as Spec-1.1 to Spec-1.5.

Spec-2) *Deadlock-free operations*: No deadlock states occur throughout system operation.

In the specification model, Spec-1.1 to Spec-1.5 are enforced by using the mutual exclusion concept. The composed PN model of both the systems and specifications is shown in Fig. 6. The supervisory arcs are shown with dashed lines and the places showing the supervisory positions are drawn thicker than those showing the system positions. A supervisory place is modeled as an input place of the transitions that need such a resource, and as an output place of those that release this resource. Take an example of $\mathbf{ps1}$ that physically means Room 1 being available. It makes two transitions $th1$ and $tc2$ mutually exclusive. Intuitively, performance of $th1$ is only allowed if Room 1 is available and $tc2$ has not yet been fired. If $tc2$ has been fired, $th1$ cannot be executed until $tc3$ 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 the collision there.

The supervisory places $\mathbf{ps1}$ to $\mathbf{ps5}$ (for Spec-1.1 to Spec-1.5, respectively) are used to prevent the remote human operator and computer controller from issuing undesired commands leading to resource conflicts on the part of the system. The corresponding notation for the supervisory places ($\mathbf{ps1}$–$\mathbf{ps5}$) is described in Table II.

D. System Verification and Deadlock Resolution

At this stage, due to its ease of manipulation, support for graphics import, and ability to perform structural and performance analyses, the software package ARP [20] is chosen to verify the behavioral properties of the composed PN model using the reachability analysis. The validation result shows that two deadlocks occur with the marked places {ph3, pc2, $\mathbf{ps2}$, $\mathbf{ps3}$, $\mathbf{ps5}$} and {ph8, pc4, $\mathbf{ps1}$, $\mathbf{ps2}$, $\mathbf{ps4}$}, respectively. Fig. 7 shows the real situations of the two deadlock states, of which the physical meaning is that if Room 1 (or Room 3) is occupied with Robot-h and Room 4 (or Room 5) is held by Robot-c, respectively, then no new events can be fired by the human or computer, and the system is deadlocked. Hence, for deadlock-free requirements, Spec-2 has two subspecifications as follows.

Spec-2.1) Robot-h is allowed to enter Room 1 only when Robot-c is not in Room 4, and vice versa.

Spec-2.2) Robot-h is allowed to enter Room 3 only when Robot-c is not in Room 5, and vice versa.

As shown in Fig. 8, $\mathbf{ps6}$ and $\mathbf{ps7}$ are further designed by using the mutual exclusion concept and then combined with the PN model in Fig. 6. Take an example of $\mathbf{ps6}$. It makes transitions $th1$ and $tc1$ mutually exclusive. That means either Robot-h

TABLE II
NOTATION FOR THE SUPERVISORY PLACES OF THE PNs IN FIG. 6

Place	Description
ps1	Spec-1.1: R1 is available.
ps2	Spec-1.2: R2 is available.
ps3	Spec-1.3: R3 is available.
ps4	Spec-1.4: R4 is available.
ps5	Spec-1.5: R5 is available.
ps6	Spec-2.1: Robot-h is admitted into R1. Robot-c is admitted into R4.
ps7	Spec-2.2: Robot-h is admitted into R3. Robot-c is admitted into R5.

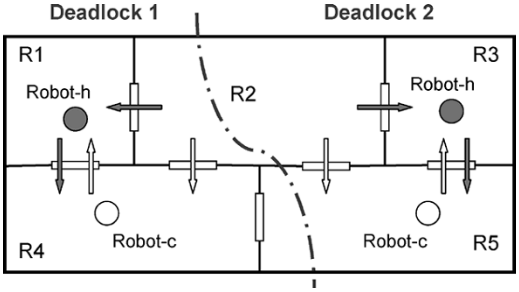

Fig. 7. Two deadlock states of the PN model in Fig. 6.

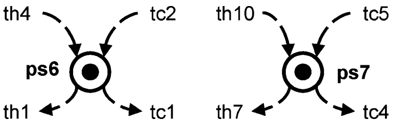

Fig. 8. Supervisory places for the deadlock resolution.

TABLE III
COMPARISON BETWEEN THE UNSUPERVISED AND SUPERVISED FRAMEWORKS

Petri net models	Unsupervised system	Preliminary supervision (with deadlocks)	Complete supervision (deadlock-free)
Places	18	23	25
Transitions	22	22	22
State space	65	44	40

moving to R1 or Robot-c moving to R4 is allowed to perform at a time. If $tc1$ has been fired, $th1$ cannot be executed until $tc2$ is given to signal that Robot-c is not in R4.

Validation results (with $\mathbf{ps6}$ and $\mathbf{ps7}$) 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, while boundedness indicates that the system can be executed with limited resources, and reversibility implies that the initial system configuration is always reachable. The corresponding notation for the supervisory places ($\mathbf{ps6}$ and $\mathbf{ps7}$) is described in Table II.

E. Discussions

On part of the human-controlled robot, in the proposed supervisory framework, the human behavior is advised and restricted to satisfy the specifications so that the collision and deadlock are avoid during the surveillance period. As shown in Table III, without supervisory control, the state space is 65, including the undesired collision and deadlock states. By using our proposed approach, in the preliminary supervision, i.e., only the collision-free specification (Spec-1.1 to Spec-1.5) is enforced, the state space reduces to 44. Finally, with the deadlock resolution, the state space is limited to 40 only. That means the undesired collision and deadlock states will be successfully avoided during the surveillance period. In this approach, the supervisor consists only of places and arcs, and its size is proportional to the number of specifications that must be satisfied.

V. CONCLUSION

This paper has presented a PN-based framework to design a supervisor for human–computer interactive systems. The supervisor is systematically synthesized to enforce the requirements. To demonstrate the practicability of the proposed supervisory approach, an application to a remote surveillance system is provided. According to the feedback status of the remotely located system, the designed supervisory agent guarantees that all requested commands satisfy the collision-free and deadlock-free specifications. On the part of human-controlled systems, the developed supervisor can be implemented as an intelligent agent to advise and guide the human operator in issuing commands by enabling or disabling the associated human-controlled buttons [8]. Hence, for human–computer interactive systems, the proposed approach would be also beneficial to the human-machine interface design.

Future work includes the extension of specifications to timing constraints, the multiple-operator access, and error recovery functions. Moreover, constructive definition of the synthesis algorithm should be investigated. Also, for the scalability of the supervisor synthesis, the hierarchical design [21] can be further applied to more complex and large-scale systems.

REFERENCES

[1] R. Safaric, M. Debevc, R. M. Parkin, and S. Uran, “Telerobotics experiments via Internet,” *IEEE Trans. Ind. Electron.*, vol. 48, no. 2, pp. 424–431, Apr. 2001.

[2] A. Rovetta, R. Sala, X. Wen, and A. Togno, “Remote control in telerobotic surgery,” *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 26, no. 4, pp. 438–444, Jul. 1996.

[3] R. L. Kress, W. R. Hamel, P. Murray, and K. Bills, “Control strategies for teleoperated Internet assembly,” *IEEE/ASME Trans. Mechatronics*, vol. 6, no. 4, pp. 410–416, Dec. 2001.

[4] S. Katsura and K. Ohnishi, “Human cooperative wheelchair for haptic interaction based on dual compliance control,” *IEEE Trans. Ind. Electron.*, vol. 51, no. 1, pp. 221–228, Feb. 2004.

[5] P. Aigner and B. J. McCarragher, “Modeling and constraining human interactions in shared control utilizing a discrete event framework,” *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 30, no. 3, pp. 369–379, May 2000.

[6] Y. Yamada, Y. Hirasawa, S. Huang, Y. Umetani, and K. Suita, “Human–robot contact in the safeguarding space,” *IEEE/ASME Trans. Mechatronics*, vol. 2, no. 4, pp. 230–236, Dec. 1997.

[7] H. J. W. Spoelder, D. M. Germans, L. Renambot, H. E. Bal, P. J. de Waal, and F. C. A. Groen, “A framework for interaction of distributed autonomous systems and human supervisors,” *IEEE Trans. Instrum. Meas.*, vol. 51, no. 4, pp. 798–803, Aug. 2002.

[8] J. S. Lee and P. L. Hsu, “Remote supervisory control of the human-in-the-loop system by using Petri nets and Java,” *IEEE Trans. Ind. Electron.*, vol. 50, no. 3, pp. 431–439, Jun. 2003.

[9] P. J. Ramadge and W. M. Wonham, “Supervisory control of a class of discrete event processes,” *SIAM J. Control Optim.*, vol. 25, no. 1, pp. 206–230, 1987.

[10] ——, “The control of discrete event systems,” *Proc. IEEE*, vol. 77, no. 1, pp. 81–98, Jan. 1989.

[11] S. Balemi, G. J. Hoffmann, P. Gyugyi, H. Wong-Toi, and G. F. Franklin, “Supervisory control of a rapid thermal multiprocessor,” *IEEE Trans. Autom. Control*, vol. 38, no. 7, pp. 1040–1059, Jul. 1993.

[12] J. O. Moody and P. J. Antsaklis, *Supervisory Control of Discrete Event Systems Using Petri Nets*. Boston, MA: Kluwer, 1998.

[13] A. Giua and F. DiCesare, “Supervisory design using Petri nets,” in *Proc. IEEE Int. Conf. Decision and Control*, Brighton, U.K., 1991, pp. 92–97.

[14] R. Zurawski and M. C. Zhou, “Petri nets and industrial applications: a tutorial,” *IEEE Trans. Ind. Electron.*, vol. 41, no. 6, pp. 567–583, Dec. 1994.

[15] M. Uzam and A. H. Jones, “Discrete event control system design using automation Petri nets and their ladder diagram implementation,” *Int. J. Adv. Manuf. Technol.*, vol. 14, no. 10, pp. 716–728, 1998.

[16] M. C. Zhou and M. D. Jeng, “Modeling, analysis, simulation, scheduling, and control of semiconductor manufacturing systems: a Petri net approach,” *IEEE Trans. Semicond. Manuf.*, vol. 11, no. 3, pp. 333–357, Aug. 1998.

[17] J. S. Lee and P. L. Hsu, “Design and implementation of the SNMP agents for remote monitoring and control via UML and Petri nets,” *IEEE Trans. Contr. Syst. Technol.*, vol. 12, no. 2, pp. 293–302, Mar. 2004.

[18] M. P. Fanti, B. Maione, and T. Turchiano, “Comparing diagraph and Petri net approaches to deadlock avoidance in FMS modeling and performance analysis,” *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 30, no. 5, pp. 783–798, Oct. 2000.

[19] M. C. Zhou and F. DiCesare, “Parallel and sequential mutual exclusions for Petri net modeling for manufacturing systems,” *IEEE Trans. Robot. Autom.*, vol. 7, no. 4, pp. 515–527, Aug. 1991.

[20] C. A. Maziero, “ARP: Petri net analyzer,” Control Microinformatic Lab., Federal Univ. Santa Catarina, Florianópolis, Brazil, 1990.

[21] M. Tittus and B. Lennartson, “Hierarchical supervisory control for batch processes,” *IEEE Trans. Contr. Syst. Technol.*, vol. 7, no. 5, pp. 542–554, Sep. 1999.

Jin-Shyan Lee received the B.S. degree in mechanical engineering from National Taiwan University of Science and Technology, Taipei, Taiwan, R.O.C., in 1997, and the M.S. and Ph.D. degrees in electrical and control engineering from National Chiao-Tung University, Hsinchu, Taiwan, R.O.C, in 1999 and 2004, respectively.

Since January 2005, he has been a Researcher in the Computer and Communications Research Laboratory, Industrial Technology Research Institute, Hsinchu, Taiwan, R.O.C. During July 2003–June 2004, he was a Visiting Researcher (supported by the National Science Council of Taiwan, R.O.C.) in the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark. His research work has led to a number of papers published in journals and international conference proceedings. His current research interests include Internet-based monitoring and control, system modeling and simulation, discrete-event systems, supervisory control, hybrid systems, and factory automation.

Dr. Lee was the winner of a SICE International Scholarship, and a finalist in both the Annual International Award and Young Author’s Award at the 2004 SICE Annual Conference, Sapporo, Japan. He was a Co-Organizer and Co-Chair of a special section, “Computer automated multi-paradigm modeling,” at the 2004 IEEE International Conference on Computer-Aided Control System Design, Taipei, Taiwan, R.O.C.

Meng-Chu Zhou (S’88-M’90-SM’93-F’03) received the B.S. degree from Nanjing University of Science and Technology, Nanjing, China, in 1983, the M.S. degree from Beijing Institute of Technology, Beijing, China in 1986, and the Ph.D. degree in computer and systems engineering from Rensselaer Polytechnic Institute, Troy, NY in 1990.

He joined New Jersey Institute of Technology (NJIT), Newark, in 1990, and is currently a Professor of Electrical and Computer Engineering and the Director of the Discrete-Event Systems Laboratory. His research interests are in computer-integrated systems, Petri nets, semiconductor manufacturing, multi-lifecycle engineering, and network security. He has authored over 200 publications , including four books, over 60 journal papers, and 13 book chapters. He co-authored with F. DiCesare *Petri Net Synthesis for Discrete Event Control of Manufacturing Systems* (Boston, MA: Kluwer, 1993), edited *Petri Nets in Flexible and Agile Automation* (Boston, MA: Kluwer, 1995), and co-authored with K. Venkatesh *Modeling, Simulation, and Control of Flexible Manufacturing Systems: A Petri Net Approach* (Singapore: World Scientific, 1998). He was invited to lecture in Australia, Canada, China, France, Germany, Hong Kong, Italy, Japan, Korea, Mexico, Taiwan, R.O.C., and the U.S.

Prof. Zhou served as an Associate Editor of the IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION from 1997 to 2000 and is currently an Associate Editor of the IEEE TRANSACTIONS ON SYSTEMS, MAN AND CYBERNETICS—PART A: SYSTEMS AND HUMANS, IEEE TRANSACTIONS ON SYSTEMS, MAN AND CYBERNETICS—PART B: CYBERNETICS, and IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING. He has organized and chaired over 70 technical sessions and served on program committees for many conferences. He was Program Chair of the 1998 and Co-Chair of the 2001 IEEE International Conference on Systems, Man, and Cybernetics (SMC) and the 1997 IEEE International Conference on Emerging Technologies and Factory Automation. He has been a Guest Editors for the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, IEEE TRANSACTIONS ON SEMICONDUCTOR MANUFACTURING, and IEEE/ASME TRANSACTIONS ON MECHATRONICS. He has been an Editor of the *International Journal of Intelligent Control and Systems* since 1996. He was General Co-Chair of the 2003 IEEE International Conference on System, Man and Cybernetics, Washington DC, and the 2003 and 2004 IEEE International Conferences on Networking, Sensing and Control, Taipei, Taiwan, R.O.C. He has led or participated in 26 research and education projects with total budget over $10 000 000, funded by National Science Foundation, Department of Defense, Engineering Foundation, New Jersey Science and Technology Commission, and industry. He was the recipient of the national Science Foundation Research Initiation Award, CIM University-LEAD Award from the Society of Manufacturing Engineers, Perlis Research Award from NJIT, Humboldt Research Award for U.S. Senior Scientists, Leadership Award and Academic Achievement Award by the Chinese Association for Science and Technology-USA, Asian American Achievement Award from the Asian American Heritage Council of New Jersey, and the Outstanding Contribution Award from the IEEE Systems, Man, and Cybernetics (SMC) Society. He was the founding Chair of the Discrete Event Systems Technical Committee of IEEE SMC Society, and Co-Chair (founding) of the Semiconductor Factory Automation Technical Committee of the IEEE Robotics and Automation Society. He was elected an AdCom member of the IEEE SMC Society twice. He is a Life Member of the Chinese Association for Science and Technology-USA and served as its President in 1999.

Pau-Lo Hsu (M’92) received the B.S. degree from National Cheng Kung University, Tainan, Taiwan, R.O.C., in 1978, the M.S. degree from the University of Delaware, Newark, in 1984, and the Ph.D. degree from the University of Wisconsin, Madison, in 1987, all in mechanical engineering.

Following two years of military service in King-Men, he was with San-Yang (Honda) Industry during 1980–1981 and Sandvik (Taiwan) during 1981–1982. In 1988, he joined the Department of Electrical and Control Engineering, National Chiao-Tung University, Hsinchu, Taiwan, R.O.C., and became a Professor in 1995. During 1998–2000, he served as the Chairman of the Department of Electrical and Control Engineering. His research interests include mechatronics, CNC motion control, servo systems, network-based control systems, and diagnostic systems.

Prof. Hsu served as President of the Chinese Automatic Control Society during 2002–2003.