Coordination Control of Discrete Event Systems using Petri Nets

J. E. Pellicer, R. Carelli

Instituto de Automática - Facultad de Ingeniería.

Universidad Nacional de San Juan (UNSJ).

Av. San Martín 1109 (oeste). - C. P. 5400. San Juan. Argentina.

Tel: +54-64-213303. Fax: +54-64-213672

E-mail: pellicer@inaut.edu.ar

Abstract

This work presents a framework for the coordination control of Discrete Event Systems (DES) based on Petri Nets (PN). The concept of enabling arcs is used in the transitions for signal reception by the controller. The structural properties of the PN controller are studied by analyzing it as an interpreted PN. PN models of both the controller and the system to be controlled are presented, the latter performed through a Temporized Petri Net (TPN). A software module for DES coordination control was developed, based on the above cited structure and programmed in C-language which interacts with the DES local controllers. This program is run in an IBM-compatible 486 PC.

Keywords

Discrete event systems, coordination control, Petri nets.

1 INTRODUCTION

Petri Nets (PNs) have been used mainly for modelling, simulating and validating Discrete Event Systems (DES) (Silva, 1985), (Murata, 1989). As regards of specification and validation of DES controllers, some works have been developed mainly oriented to the application of Programmable Logic Controllers (PLCs) (Furukawa, 1990), (Silva, 1982), (Silva, 1985), (Gomes, 1993). Few references are available on implementation of DES coordination control using PNs. (Brand, 1988) presents a PN representation for the communication of a process control task with the process itself. The PN model does not use enabling arcs and is not validated as a framework for coordination control. (Cofer, 1992)

does not present a framework for DES coordination control. (Feldmann, 1995) presents an approach for real-time control of FMS using PN. In (Miyagi, 1993) the Mark Flow Graph/Production Flow Scheme (MFG/PFS) for DES control is used. The MFG, which is derived from PNs, focuses on modelling the DES local controller.

The implementation of DES *coordination control* necessitates a PN structure capable of sending and receiving signals to and from the external world, namely, the local controller, and of solving the conflicts arising in the PN which models the coordinator. The implementation also needs that the control be continuously monitored and that its safe operation be warranted. The main contribution of the present work is to propose an adequate PN framework for DES coordination control and to discuss the aspects related to its validation and implementation. As a previous step towards implementation, a PN model is studied which combines an ordinary PN for modelling the controller and a TPN for simulating the process to be controlled. Then, the link of the controller to a simple real system is also considered, to show the practical feasibility of our proposal.

The paper is organized as follows: Section 2 presents a discussion on the ways for applying PNs to the control of DES by using the concept of enabling arcs in the transitions. This section also presents a modification to the traditional equation for the evolution rule of PN markings with internal enabling arcs. Section 3 gives the characteristics of the PN framework used as coordinator controller and describes a software module based on this framework, to which a "watchdog" was added. Section 4 presents a simulation example linking the PN coordination controller to a Temporized Petri Net (TPN) representing the system operation. Section 5 shows a simple experimental example. Finally, Section 6 presents the conclusions and future works.

2. INTERPRETED PETRI NETS

Generalized Petri Nets (GPN) have been considered as a mathematical structure having a dynamical property (marking). For allowing a GPN to represent a system, it needs to be given an *interpretation*. To interpret a GPN means to set an agreement to define the following aspects (Silva, 1985):

- A physical significance to the necessary conditions for firing a transition. The evolution rules are slightly modified by the interpretation, which also turns to be an evolution function of the process to be controlled.
- The actions resulting from the marking evolution.

The internal state of the system is always represented by the PN marking. As regards of a PN interpretation, it is worth mentioning that there is no single possible interpretation for describing an application class. Therefore, for describing logical systems there are two large groups, according to whether the generated actions are associated to the places or to the transitions (Silva, 1985).

The autonomous PN (which lacks interpretation) cannot be used for describing the operation of system controllers. These nets lack the necessary information for taking into account the present state of the plant, and for determining the necessary operations for modifying the state of the system. For specifying the control function on a DES by means of a PN, it is necessary to introduce three new functions. One of these is in charge of reading the present state of the plant; another function modifies such plant state. The third function

is assigned to solving conflicts (Treviño, 1990). Consequently, the characteristics for such a controller model should regard the following aspects (Miyagi, 1993):

- To prevent the occurrence of conflicts.
- To warrant the system will not get blocked. There always has to be an activated event about to occur.
- To be capable of sending the model state signals to the external devices.
- To be capable of receiving the signals generated from the external devices and, based on these signals, to control the occurrence of model events.

2.1 Petri Nets with Enabling Arcs

The concept of enabling arcs is used in special classes of PNs such as Mark Flow Graphs (MFG) where the arcs are called gates (Miyagi, 1989), (Miyagi, 1993). The enabling arcs are generally associated to external events or conditions, and it is also possible to use them as internal elements of a PN. An enabling arc is used when a transition is enabled by the marking as well as by a precondition which should remain after the transition firing. It differs from a common arc in that the transition firing does not cause to delete the marking from the place linked to the transition through the enabling arc. An enabling arc, is represented by means of a black circle at the end connected to a transition. Figure 1 presents enabling and firing situations of a PN having enabling arcs.

Figure 1: PN with enabling arc.

2.2 Marking evolution rule in a PN with internal enabling arcs.

The evolution of a PN is stated in the following equation:

$$M_f = M_i + A.d \quad (1)$$

where: M_f : final marking; M_i : initial marking; A : incidence matrix; d : firing vector

(Silva, 1995) shows the marking evolution rule of a PFS/MFG model with enabling and disabling arcs. In a similar way, in order to obtain an equation valid for a PN with enabling arcs, we resort to a G matrix which expresses by 1's the places and transitions linked by an enabling arc,

$$M_f = M_i + A.d + G.d \quad (2)$$

where: Mf , Mi and d have the same meaning as in (1). A : incidence matrix; it includes enabling arcs as common input arcs, that is to say: $a(i,j) = -1$ if p_i is precondition for t_j , or if there exists an enabling arc between p_i and t_j ; $a(i,j) = 1$ if p_i is post-condition for t_j ; $a(i,j) = 0$ for any other case. G : compensation matrix for enabling arcs, where: $g(i,j) = 1$ if there exists an enabling arc between place p_i and transition t_j ; $g(i,j) = 0$ for any other case.

Given a PN, matrix A represents the enabling arcs as common arcs and through matrix G the effect of enabling arcs is obtained; that is, every time a transition with enabling arc is fired, the marking of the place to which it is linked, is not lost.

Note that, if in (2) it is practiced right-hand common factor, the vector of the equation becomes:

$$Mf = Mi + (A + G).d = Mi + A'.d \quad (3)$$

where: $A' = A + G$. Matrix A' is the incidence matrix of the PN, equivalent to a non-pure PN, where the transitions having enabling arcs are linked through self-loops to the respective places.

3. COORDINATION CONTROL FRAMEWORK USING PETRI NETS

Bearing in mind the convention which states that PN places are associated to conditions or states of the system to be controlled and that transitions are associated to system events or state changes, therefore the beginning of an activity carried out by the system will be caused by the firing of a transition of the controller's PN. This firing clears the marking of the place which models the present state (precondition for the transition). Likewise, when the desired state is reached, that is, when the activity being modelled for this transition is finished, it should proceed to mark the place corresponding to the condition following to such transition. Then, the autonomous PN becomes an interpreted PN, which can be in a simulation condition context, equivalent to a TPN with a period elapsing between the clearing of the marking of the pre-condition place and the marking of the post-condition place corresponding to the time associated to the simulated activity. Figure 2 illustrates the above stated concepts.

Figure 2: Relationship between autonomous PN, Interpreted PN, and TempORIZED PN.

3.1 Basic Coordination Control Scheme

Following the concepts stated in the previous section, a PN based coordination control scheme is now proposed. Figure 3 shows the basic scheme for controlling a task, in which an additional place is created in order to keep the mark while the task is being executed. This task may be an activity of an actual physical process, such as the motion of a robotic manipulator, the displacement of an AGV, etc; or, it may also be the simulation of the activity carried out through a simulator of a continuous variable system (Carelli, 1995). The activity to be controlled is referred as a transition from state E1 to state E2 ($E1 \rightarrow E2$).

Figure 3: Coordination control framework by means of PN.

The description of PN elements are: p1: State E1 (precondition of activity $E1 \rightarrow E2$), p2: Intermediate state: $E1 \rightarrow E2$, p3: Capacity monitor of p2, p4: State E2 (post-condition of activity $E1 \rightarrow E2$), t1: Action starts (signal is sent), t2: Confirms the completion of action (if a signal is received).

When place p1 is marked (state E1), t1 can be fired, because it is enabled by this marking and by that of p3. When t1 is fired, it sends a signal to the process and p2 becomes marked, thus indicating that the evolution of state $E1 \rightarrow E2$ is now in progress. If any other marking arrives to p1, t1 is unable to fire due to the capacity monitor of p3 which is not marked. This condition causes that no other signal could be sent to the process until the activity reaches completion (firing of t2). Upon completion of this activity, the process reaches state E2 and sends a signal to t2, thus enabling t2 by means of the enabling arc and the marking of p2. Then, t2 is immediately fired, making p3 be marked again, clearing the marking of p2 and marking p4 as well, thus indicating in the coordination controller that the system has been led to state E2.

3.2 Structural Analysis

In the sub-PN of figure 3, it can be observed that the linking to the process does not cause any mark to be cleared from nor entered to the PN. Since it is assumed that a marking invariant is a constant relationship of markings between the PN places (Silva, 1985), (Pellicer, 1993), it can be concluded that, with this coordination control framework, the PN

modelling the controller can be structurally analyzed, regardless of the process. This is a sufficient condition, that is, if the autonomous PN is structurally bounded and conservative, so will be the PN for the control. The condition is not necessary, because it is possible for an interpreted PN to be structurally bounded and conservative, whereas its corresponding autonomous PN may not be so. A comprehensive example is given in (Silva, 1985).

In a like manner, bearing in mind that a firing invariant is any relationship satisfied by all firing sequences applicable from the initial marking, it can be clearly seen that in the proposed framework, the linking to the external world does not alter the T-invariants. This is true while it is guaranteed that there are not anomalies in the system. For example if there exist an anomaly in the change of states $E1 \rightarrow E2$, and it never appears a true sign for the enabling arc in $t2$, this transition could not discharge, making that the PN used for the control be not repetitive, in spite of the fact that the corresponding autonomous PN is so. Therefore, as a conclusion for the analysis of repetitiveness, it could be stated that: If an autonomous PN is repetitive, all interpreted PN built from the previous one is also repetitive, provided all the events associated with the transitions are able to occur.

3.3 Software Module

For implementing a DES coordination controller through PN, a special software module with enabling arcs, for receiving external signals, was developed. This module is also capable of sending signals to the exterior by transition firings. These features are attributes pertaining to the PN transitions in such a module. The software module is programmed in C Language and operates according to a treatment cycle presented in (Villarroel, 1992), taking into account that the codes associated to transition firings are signals sent by the PC's parallel gate. This module is currently being included in the PASCELL software package (Colombo, 1994).

This software module must include detection of process anomalies that interrupt the evolution of the PN controller. A simple method for detecting the interruption of the PN evolution (deadlock) based on a simple watch dog is proposed. The watch-dog (Velilla, 1984) is a hardware device, assigned to be alert on the time elapsed for going from state $E1$ to $E2$ not to surpass a preset maximum time. The device works as follows (see Figure 3): When $t1$ fires for commanding the activity $E1 \rightarrow E2$ to begin, place $p2$ becomes marked, therefore indicating that a change of states is in progress. Should a process anomaly occur, such that no signal ever comes through the enabling arc to $t2$, then a watch-dog is needed, after the preset time has elapsed, for clearing the markings of $p2$, making $p1$ and $p3$ be marked and deleting the prompt for state change $E1 \rightarrow E2$, originated upon firing $t1$. In conclusion, the watch-dog is a simple alarm programmed into the software module which actuates after a preset time has elapsed, in order to reset the system.

4. SIMULATION EXAMPLE

In this example, a physical system is in any of two states $E1$ or $E2$. For instance, an AGV may be at any of two physical places. The system has to go from state $E1$ to state $E2$ and viceversa. Figure 4 shows its coordination control scheme using PNs.

Figure 4: Coordination control framework of a two states system by means of PN.

Figure 5: PN model for the controller and process.

PN elements are described as follows: p1: State E1, p2: Intermediate state: $E1 \rightarrow E2$, p3: Capacity monitor for p2, p4: State E2, p5: Intermediate state: $E2 \rightarrow E1$, p6: Capacity monitor for p5, t1: Initiates action for going from E1 to E2, t2: Confirms action executed ($E1 \rightarrow E2$), t3: Initiates action for going from E2 to E1, t4: Confirms action executed ($E2 \rightarrow E1$).

In Figure 4 it is clearly seen that whenever place p1 is marked, and due to p3 is marked as well, t1 is fired. That is, the coordination control directs the process (local controller) to initiate the actions for carrying out the desired state change. Place p2 is marked indicating that the process state is changing. Once E2 is reached, transition t2 is enabled; t2 is fired, therefore marking p3 again, as well as marking p4 and clearing the mark of p2. The marking of p4 indicates, in the coordination controller, that state E2 has been reached. Now the signal for state change $E2 \rightarrow E1$ is sent, by means of firing t3. Places p5 and p6 are assigned the same duties as for p2 and p3, respectively, in the previous state change. It can be noted that, in this particular case, both p3 and p6 can be dispensed of. The reason is that, once t1 is fired, it is impossible to have p1 marked again until activities ($E1 \rightarrow E2$ and $E2 \rightarrow E1$) are completed.

In order to analyze the operation of the framework proposed for the coordination controller using PN, a model is suggested embodying both the coordination controller and the physical process. The PN which simulates the physical process is a Temporized Petri Net (TPN). Figure 5 shows the corresponding PN model. Note the controller PN transitions are represented by lines, because they correspond to instantaneous transitions, and those corresponding to the simulated process are denoted by rectangles, because they are part of a TPN. These last transitions, in turn, have as an attribute the time representing the mean duration of the activity they are modelling. For the PN evolution, the simulator uses eq. (2) because the PN has internal enabling arcs.

In Figure 5, the PN is described as follows: p1: State E1 in the coordination controller, p2: Intermediate state $E1 \rightarrow E2$, p3: State E2 in the coordination controller, p4: Intermediate state $E2 \rightarrow E1$, p5: Pre-condition for state change in the process ($E1 \rightarrow E2$), p6: State E1 in the process, p7: State E2 in the process, p8: Pre-condition for state change in the process ($E2 \rightarrow E1$), t1: Initiates action for going from E1 to E2, t2: Confirms action executed ($E1 \rightarrow E2$),

t3: Initiates action for going from E2 to E1, t4: Confirms action executed ($E2 \rightarrow E1$), t5: Simulates the duration of the process $E1 \rightarrow E2$, t6: Simulates the duration of the process $E2 \rightarrow E1$. The operation is easy to interpret taking into account the description of figure 4 and the temporized transition t5 and t6 in the process PN model.

5. EXPERIMENTAL DEVELOPMENT

The proposed coordination control framework was used for coordinating an experimental setup of an overhead travelling crane. A simplified scheme is shown in Figure 6, where the possible positions of the crane, as controlled by the coordination controller, can be seen. From all possible crane positions, only four are considered here, namely: E1: crane down - left side, E2: crane up - left side, E3: crane up - right side y E4: crane down - right side.

5.1 Controller model

The controller's PN model is shown in Figure 7, where a block represents the actual plant with its local controller.

Figure 6: Scheme of the process to control.

Figure 7: PN model of the crane coordination control.

The description of the PN is as follows: p1: Crane in position E1, p2: Intermediate state $E1 \rightarrow E2$, p3: Intermediate state $E2 \rightarrow E1$, p4: Crane in position E2, p5: Intermediate state $E2 \rightarrow E3$, p6: Intermediate state $E3 \rightarrow E2$, p7: Crane in position E3, p8: Intermediate state $E4 \rightarrow E3$, p9: Intermediate state $E3 \rightarrow E4$, p10: Crane in position E4, p11: Intermediate state $E4 \rightarrow E1$, p12: Intermediate state $E1 \rightarrow E4$, t1: Command $E1 \rightarrow E2$ (signal s1), t2: Confirmation $E2 \rightarrow E1$, t3: Confirmation $E1 \rightarrow E2$, t4: Command $E2 \rightarrow E1$ (signal s2), t5: Command $E2 \rightarrow E3$ (signal s3), t6: Confirmation $E3 \rightarrow E2$, t7: Confirmation $E2 \rightarrow E3$, t8: Command $E3 \rightarrow E2$ (signal s4), t9: Confirmation $E4 \rightarrow E3$, t10: Command $E3 \rightarrow E4$ (signal s5), t11: Command $E4 \rightarrow E3$ (signal s6), t12: Confirmation $E3 \rightarrow E4$, t13: Command $E4 \rightarrow E1$ (signal s7), t14: Confirmation $E1 \rightarrow E4$, t15: Confirmation $E4 \rightarrow E1$, t16: Command $E1 \rightarrow E4$ (signal s8).

Figure 7 shows the crane's initial state, position E1, denoted with place p1 marked. The possible transitions to be fired are t1, which commands the crane to move up, and t16 which

moves the crane to the right. If the operator decides to move the crane upwards, t_1 is fired, sending the signal s_1 to the local controller and marking the place p_2 , which in turn indicates that the crane is changing state from E1 to E2. During this state change there is no transition enabled in the PN of the coordination controller. Upon reaching state E2, the confirmation signal is sent from the plant to the coordinator (signal C2), enabling t_3 through the enabling arc. The remaining state changes of the crane are carried out in a similar way.

Note that there exist the following conflicts: between t_1 and t_{16} for place p_1 ; between t_4 and t_5 for place p_4 , between t_8 and t_{10} for place p_7 , and between t_{11} and t_{13} for place p_{10} .

In other words, for every case where an action has to be performed for making the system change its present state, there is a chance for choosing one out of two possibilities. In this case the conflict is solved by the operator by firing some of the enabled transitions. (Pellicer, 1995) shows a possibility for solving conflicts using predicates associated to the transitions.

5.2 Validation of the PN model

The results obtained in the simulation and validation are the following ones:

P-Support: $sp_1: \{p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9, p_{10}, p_{11}, p_{12}\}$; T-Supports: $st_1: \{t_1, t_2, t_3, t_4\}$; $st_2: \{t_5, t_6, t_7, t_8\}$; $st_3: \{t_9, t_{10}, t_{11}, t_{12}\}$; $st_4: \{t_{13}, t_{14}, t_{15}, t_{16}\}$; $st_5: \{t_1, t_3, t_5, t_7, t_{10}, t_{12}, t_{13}, t_{15}\}$; $st_6: \{t_2, t_4, t_6, t_8, t_9, t_{11}, t_{14}, t_{16}\}$;

Interpretation of supports

The support sp_1 establishes that the token cannot be at more than one place at the same time. This ensures that the crane will not be at two or more positions at the same time. In terms of PN properties, it is said that all PN places are in mutual exclusion.

The T-supports establish the possible operating cycles of the system. For instance, st_1 sets that there is a cycle firing t_1 , t_2 , t_3 and t_4 . That is, the crane at E1 moves to E2 and, then, it can return to E1 and viceversa. The supports st_2 , st_3 and st_4 are likewise analyzed. The remaining two supports, st_5 and st_6 , give the firing sequences necessary for making the crane move through all possible positions.

st_5 : Sets the transition sequence for the crane trajectory: $E1 \rightarrow E2 \rightarrow E3 \rightarrow E4 \rightarrow E1$.

st_6 : Sets the transition sequence for the crane trajectory: $E1 \rightarrow E4 \rightarrow E3 \rightarrow E2 \rightarrow E1$.

From these P-supports and T-supports, the structural properties of the PN model are obtained. These properties are:

- *Repetitive net*: The system can reach a repeated state through a sequence in which all the transitions are fired. This is an example of deadlock-free system.
- *Conservative and bounded net*: No place can have an infinite number of tokens. The initial marking of the system is kept.

6. CONCLUSIONS

In the present work, the development of a PN-based coordination control for DES was proposed. PN's have been proven as an appropriate means for modelling, simulation and validation of DES, and applicable for implementing DES coordination control as well.

A framework for DES coordination control has been presented, based on PNs with enabling arcs in their transitions. An analysis was performed on the structural properties of the proposed model studied as and interpreted PN (IPN), concluding that such an IPN is both bounded and conservative whenever the autonomous PN is bounded too.

It has been presented a PN model for the controller and the system to be controlled; the latter was modelled through a Temporized Petri Net (TPN). This net is useful for simulating the entire control system.

Future work will include the addition of the coordinator's software module to the PASCELL software package developed at the Institute of Automatics (INAUT). It is also proposed to research and develop techniques for detecting and debugging errors.

REFERENCES

- Brand, K. and Kpoansky, J. (1988). Principles and engineering of Process Control with Petri Nets. IEEE. Transactions on Automatic Control. V. 33. Nº2. pp. 138 - 149.
- Carelli, R., Colombo, W., Bernhardt, R., and Schreck, G. (1995). Discrete Event and Motion - Oriented Simulation for FMS. BASYS '95. Balanced Automation Systems. Ed. Chapman and Hall. pp. 363-372. Brasil.
- Cofer, D. and Garg, V. (1992). A Timed Model for the Control of Discrete Event Systems involving Decisions in the Max/Plus Algebra. Proc. of the 31st Conf. on Decision and Control. pp. 3363-3368. Arizona, USA.
- Colombo, A., Pellicer, J., Martin, M. and Kuchen, B. (1994). Simulation system of Flexible Manufacturing System by using Temporized Petri Nets. Proceedings of X Brazilian Congress on Automatica - VI Latinoamerican Congress on Automatic Control, Vol. 1, pp. 100-102. Rio de Janeiro, Brasil. (Spanish).
- Feldmann, K., Colombo, W. and Schnur, C. (1995). An approach for modeling, analysis and real-time control of FMS using petri nets. Proceedings of European Simulation Simposium, pp. 661-665.
- Furukawa, C. M., M. Marchese and P. Miyagi. (1990). Methods and Techniques for PLCs efficient programming in complex applications. Proceedings of IV Latinoamerican Congress of Automatic Control, pp. 250-255. Puebla, México. (in Portuguese).
- Gomes, L., A. Steiger-Garçao, L. Gama and Correia, N. (1993). Controllers Programming by means of Petri nets. Robótica Inteligente. Faculdade de Ciências e Tecnologia. Universidade Nova de Lisboa. (Portuguese).
- Miyagi, P. E., Furukawa, C. M., Maruyama, N. and Tsuzuki, F. S. G. (1989). Manufacturing Systems: Modeling, Analysis and Design trends. Proceedings of X Brazilian Congress on Mechanical Engineering, pp. 491-494. Rio de Janeiro, Brasil. (Portuguese).
- Miyagi, P. E. (1993). Discret Event Systems Control. VI Escuela de Robótica. CINVESTAV. (Portuguese).
- Murata, T. (1989). Petri Nets: Properties, Analysis and Applications. Proc. IEEE. V. 77, Nº.4, pp. 541-480.
- Pellicer, J., Colombo, W. and Carelli, R. (1993). Software for structural validation of Flexible Manufacturing Systems model by means of Petri nets. Proceedings of the V Workshop on Information Processing and Control, pp. 494-501, Tucumán, Argentina. (Spanish).
- Pellicer, J. and Carelli, R. (1995). Coordination control of an experimental setup of an overhead travelling crane by means of Petri nets. Proceedings of the VI Workshop on Information Processing and Control, pp. 126-131. V. 1. Bahia Blanca, Argentina. (Spanish).
- Silva, M. and Velilla. (1982). Programmable Logic Controllers and Petri Nets: A Comparative Study. IFAC Software for Computer Control. Madrid, España.
- Silva, M. (1985). The Petri nets in the informatic and automatic systems. AC Ed. Madrid, España. (Spanish).
- Silva, R. and Miyagi, P. (1995). PFS/MFG: A High Level Net for the Modeling of Discrete Manufacturing Systems. BASYS '95. Balanced Automation Systems. Ed. Chapman and Hall. pp. 107-120. Brasil.
- Treviño, A. R. (1990). Task modelling and controllers specifications in robotics cells. MSc Thesis. (in Spanish).
- Velilla, S. and Silva, M. (1984). Design of a real time controller 'safety for operation' II IFAC National Symposium on automation in the Industry. pp. 303-310. Zaragoza.
- Villarreal Salcedo, J. L. (1992). ADA implementation of concurrent applications specified by means of Petri nets. Departamento de Ingeniería Eléctrica e Informática. Universidad de Zaragoza.

BIOGRAPHY

JORGE EDUARDO PELLICER

Jorge Eduardo Pellicer, was born in San Juan, Argentina in 1966. He graduated as Electronic Engineer (with honours) from the National University of San Juan (Argentina). At present he is pursuing the Master Degree in Control Engineering, on Coordination Control of Discrete Event Systems. He holds a grant from the National Council for Scientific and Technical Research (CONICET). His current research interests are in the fields of Petri Nets, Discrete Event Systems Control and Manufacturing Systems. Mr. Pellicer is a student member of The Institute of Electrical and Electronics Engineers (IEEE).

RICARDO CARELLI

Ricardo Carelli received the Electrical Engineering degree (with honours) from the National University of San Juan (Argentina) in 1976 and the PhD degree in Electrical Engineering from the National University of Mexico (UNAM) in 1989. At present he is a professor of Automatic Control at the National University of San Juan (UNSJ) and research-worker of the National Council for Scientific and Technical Research (CONICET). His current research interests are in the fields of robot control, manufacturing systems and artificial intelligence applied to systems control. Dr. Carelli is a member of IEEE and AAECA, the Argentine NMO of IFAC.