Available online at www.sciencedirect.com

SCIENCE DIRECT®

Robotics and Computer-Integrated Manufacturing 21 (2005) 241–247

Robotics and Computer-Integrated Manufacturing

www.elsevier.com/locate/rcim

Describing an IMS by a FNRTPN definition: a VHDL approach

C.A. Graciós Marín[a,*], E. Vargas Soto[b], A. Díaz Sánchez[a,c]

[a]Instituto Tecnológico de Puebla, Mechanical Industrial Engineering, Av Tecnológico No. 420, Colonia Maravillas Puebla, Pue. 72220, México
[b]Centro de Ingeniería y Desarrollo Industrial (CIDESI), Av Pie de la Cuesta #709 U. Hab. Des. San Pablo, Santiago de Querétaro, Qro. México
[c]Inst. Nal. de Astrofísica, Optica y Electrónica, Luis Enrique Erro No. 1 Tonantzintla, Puebla, Pue. P.O. Box 52, México

Received 8 September 2004

Abstract

A new Petri net extension and a novel method describing the structure and behaviour of an intelligent manufacturing system (IMS), using a VHDL tool, is proposed. The PN extension is defined as Fuzzy Neural Real Time Petri Net (FNRTPN), where the fuzzy part let the intelligent scheduling of the tasks for the IMS, and the neural part calculates the estimation of the parameters of the set point for each resource in the system. At last, VHDL is proposed as an Unified Modelling Language (UML) to express the model of the manufacturing process. The extension and VHDL are used to apply a fuzzy neural control scheme to a FMS.
© 2004 Elsevier Ltd. All rights reserved.

Keywords: Intelligent manufacturing systems; Modelling intelligent process; Simulation; Fuzzy neural control strategy

1. Introduction

The design, modelling, simulation, fault analysis and control of flexible manufacturing systems, have interested in the last 30 years to academic and industrial communities. Despite several solutions have been proposed, the actual requirements of the manufacturing systems, implying the insertion of intelligence in the elements and devices included in the process [15]. In that way, the active interaction of the human with software and hardware, agility, fault tolerance and the adaptability in general, are decisive characteristics that any intelligent manufacturing system should satisfy [1]. Therefore, the new methods and tools for design, simulation and control, must include a unified modelling language that establish a direct translation between the parameters of the process and the different strategies of intelligent control.

The present work begins with a brief description of the state of the art for modelling, simulation and control of FMS, where real-time Petri nets (RTPN) are modified to introduce a new extension of PN, and strategies for intelligent control applied in manufacturing systems. After that, fundamental theorems for the new PN extension are expressed. Finally, an example of modelling, simulation and control of an Intelligent Transportation, into an IMS, is described. VHDL will be used for the modelling and control of the IMS, to show the requirements improvement in an actual manufacturing process.

2. Petri nets definitions

2.1. Real-time PN model

Nowadays, PN have been used as an option for modelling, simulation, analysis and control for manufacturing systems. According to Venkatesh et al. [2], RTPN not only models the manufacturing process, but obtains the direct digital control system, described by a Ladder Logic Diagram (LLD).

*Corresponding author. Tel.: +52 222 229 88 24; fax: +52 222 222 21 14.
E-mail addresses: cgracios@itpuebla.edu.mx (C.A.G. Marín), emilio@mecatronica.net (E.V. Soto), adiazsan@inaoep.mx (A.D. Sánchez).

0736-5845/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.rcim.2004.09.001

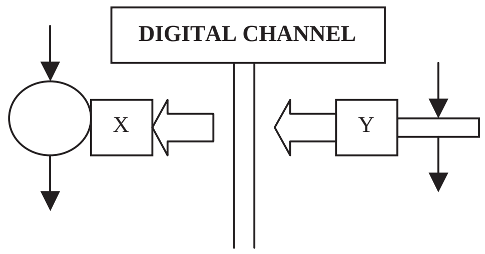


Fig. 1. Input and output vectors in RTPN extension.

A RTPN can be obtained by associating timing, and I/O (input/output) set of information to the untimed PN, and it can be defined as

$$\text{RTPN} = \{P, T, I, O, m, D, X, Y\},$$

where $P = \{p_1, p_2, \ldots, p_m\}$ is a finite set of places, $T = \{t_1, t_2, \ldots, t_n\}$ is a finite set of transitions with $P \cup T = 0$ and $P \cap T = 0$, $I{:}P \times T \to N$ is an input function that defines the set of directed arcs from P to T where $N = \{1, 2, \ldots\}$, $O{:}P \times T \to N$ is an output function that defines the set of directed arcs from T to P, $m_i{:}P \to N$, is a marking vector whose ith component represents the number of tokens in the ith place. An initial marking vector is denoted by m_0, $D{:}T \to R+$, is a firing time function where $R+$ is the set of non-negative real numbers, $X{:}P \to \{-,0,1,2,\ldots K\}$ and $X(pi) \neq X(pj), i \neq j$, is an input signal function, where K is the maximum number of input signal channels, and "-" is the dummy attribute indicating no assigned channel to the place (See Fig. 1.) $Y{:}T \to L$, is an output signal function, and L is a set of integers.

RTPN properties

1. Timing vector (D) is intended to associate time delays to transitions modelling the activities in the system.
2. Input signal vector (X) reads the state of the input signals from digital input interface. X associates attributes to every place. $X_i = X(p_i)$, where p_i represents the number of channels (bits) of the input p_i. The contents of any X_i are either 0 or 1.
3. Output signal vector (Y) is intended to send output signals through digital output interface. Y associates attribute to every transition. $Y_i = Y(t_i)$ is the attribute associated to transition t_i which represents the number that is to be sent to the digital output interface.

While the program is executed, RTPN writes the decimal number corresponding to the output channel to digital output interface when a transition is fired.

Using the X and Y vectors, a new extension, based on the inclusion of decision strategy by intelligent control, and the parameter acquisition of the process for the regulation the best estimation can be obtained by using a neural network with back propagation training.

In addition, a new variation for hardware description languages, using direct translation for the control expressed in a ladder logic diagram, is available. That new alternative compacts the design and development for the intelligent control in a programmable device, which can be applied at the process modelled [3,4].

2.2. *The fuzzy neural RTPN model*

Several contributions on the application of intelligent control to solve specific problems can be found in recent literature. Some propositions, based on intelligent control strategies, allow the timing, sequencing and scheduling of tasks, adding adaptability to process parameter variations, such that velocity, position and used materials [5,6].

Fuzzy logic, neural networks (NN), genetic algorithms and hybrid systems are actually applied in intelligent manufacturing system. In fact, fuzzy and NN are the most common structure used to control the activities, like robots, conveyor belts and CNC machines, of specific manufacturing process. Moreover, the inclusion of fuzzy and NN in PN definition to solve the short-term control, are included in recent works [16].

The present work describes a neurofuzzy control scheme, where neural part measures the signal error on the coupled sensors in each one for the regulation of the control parameters, and allows evaluating the task sequencing. Since a fuzzy task scheduler, which determinates the activation of resources according to disposability, can be defined, the goal of intelligent parameter adjusting of each resource of the process, based on previous fuzzy decisions, improves the adaptability.

3. VHDL and intelligent control scheme

Fig. 2 shows the FNRTPN extension, where the Y channel add the fuzzy logical decision on the PN firing, which simulates the decision taken to activate the resources of the real process. The NN part is applied to estimate the condition of the resources using a

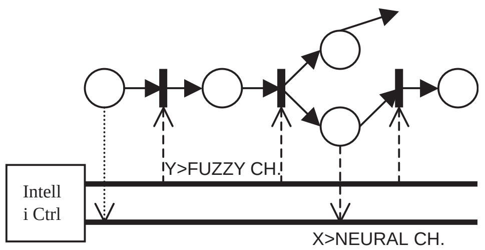


Fig. 2. FNRTPN extension.

back-propagation scheme. Since the condition of the whole process is known, the availability of the resources and the firing decision of the tasks can be controlled.

In order to include the fuzzy and NN schemes to the original definition of the RTPN, a two-mapping function must be firstly defined.

If a new firing vector of the PN F is defined, restricted to the fact that scheduler knows the state of the places in any time of the evolution of the process. The definition of F requires a mapping to translate the fuzzy logical operation values of the scheduler into converted digital values for the original RTPN. That fuzzy scheme provides an intelligent firing procedure for one or more transitions to enable activities (places) of the resources modelled. In other words, F is indirectly mapped by the firing vector D to FIRE the change transition of the RTPN actual states. To evaluate the actual state activities, the neural part is connected to the X digital channel, which recognises the values of the sensors [13,14,17].

NN can be applied to adjust the parameters for a soft variation in a typical discrete PID control scheme [7]. However, the X channel is used to modify the condition of the transitions firing (enabling the activities). Since the output of the final neuron convert the real value of the sensor connected in the structure, to a fuzzy value which is evaluated for the scheduler to determine the actual condition of each recourse.

Consequently, the NN identifies the actual values of the resources (position, velocity, pressure, etc.) to be known by the scheduler to decide the next activity. Therefore, if $Z = \{z_1, z_2, \ldots, z_m\}$ is the finite set of neurons connected to each X channel, $Z{:}X \Rightarrow P$, where the digital values of the sensors are evaluated by the NN to refine the decision of the scheduler and the values to be adjusted in the actuators of each resource, and m is the number of places, indicating the status of all the resources. Therefore, the number of sensor must be connected to verify the actual condition of each resource.

For the present work, a single-slide back-propagation NN structure is proposed, because its efficiency in two-slides scheme [7,12]. The neural scheme is shown in Fig. 3. That structure will be only used in the auto tuning strategy for the discrete and analogue actuators of the resources. The final decision on the parameter values will depend on the scheduler.

3.1. VHDL description for FNRTPN

VHDL is a suitable tool to describe and model DES by the structural and behavioural definitions in the entity and architecture schemes [9]. Several authors propose VHDL to model structures of intelligent control, incorporating fuzzy, NN, or other kind of

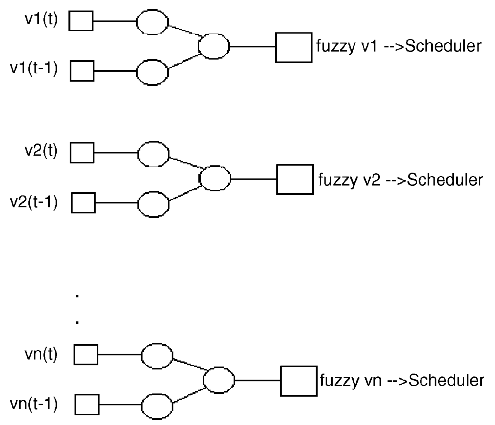


Fig. 3. Neural scheme.

schemes applied in FMS for sequencing, scheduling and control.

Using typical command description for discrete systems, a good level definition can be obtained to describe the structure and behaviour of the FNRTPN model.

3.2. Fuzzy neural control scheme

The use of a hardware description language is proposed as a tool of simulation and analysis for the FNRTPN model to develop a neurofuzzy control scheme to be applied in the real process. VHDL allows a concurrent design, which uses a unified modelling language with a unique database, generated and addressed at any step of the method [10].

Despite VHDL is mainly used in the design of complex digital circuits, it can be used in FNRTPN to describe processes like DES. At last, the simulation, analysis and design of the control can be evaluated using a typical discrete event evolution diagram, where transition signals express the discrete behaviour of the process in terms of the scheduling, timing and sequencing of task.

If the simulation and analysis have been exhaustively realised without find problems in feasibility, the next step for the synthesis of the intelligent control can be made.

Using the report generated for VHDL tool, an Inverse Logistic Algorithm (ILA) can be proposed to obtain the structure of the control scheduler. Therefore, the fuzzy vector F, and the neural set NN allows the interconnection between the real process and the intelligent control scheme (see Fig. 2).

The structure and design for the scheduler was proposed by Salapura [8], where the fuzzifier, the defuzzifier, the set of rules and the inference machine are coded in VHDL. Moreover, the behavioural information is obtained by the ILA algorithm applied to the FNRTPN model of the process.

The ILA procedure is as follows:

Each t_i (process) is marked as p_j (FNCS).
Each p_i (process) is marked as t_j (FNCS).
$I \Leftrightarrow O$ for each state in the structure of the FNRTPN.
$A(\text{FNCS}) = A-1$ (process).
Evolutionary method (Dadone).
ASM description for the structure and behavioural obtained.
Coupling step between the CORE defined for typical resources restricted for CADYC method proposed and the ASM description.
Definition in VHDL.
Testing until complete satisfaction.

That algorithm is based on the scheme suggested by Dadone in 1997 [16], where an evolutionary programming method is presented. The method offers improvements in the use of VHDL, as a unified modelling language, for all the steps required in control schemes if the process is always described like a DES, and the control strategy is implemented in a programmable device.

4. An example—intelligent transport

Fig. 4 shows the layout of the Transport system, which is a part of a FMS analysed in Instituto Tecnológico of Puebla. That system is used for distribution and inspection of the parts manufactured in the FMS. The neurofuzzy control scheme, allows an intelligent adjust for the process. Two Mitsubishi$^{\text{TM}}$ 5 degrees of freedom robots are included as resources of manipulation of the pieces. The first robot includes a camera vision for the detection of the pieces and the piece can be taken. Two conveyor belts (cv), driven by, servo motors are used for transportation. Additional sensors and actuators are included, as shown in Fig. 5.

Fig. 5 is an extended layout of the system to be modelled and controlled, where the X and Y parameters

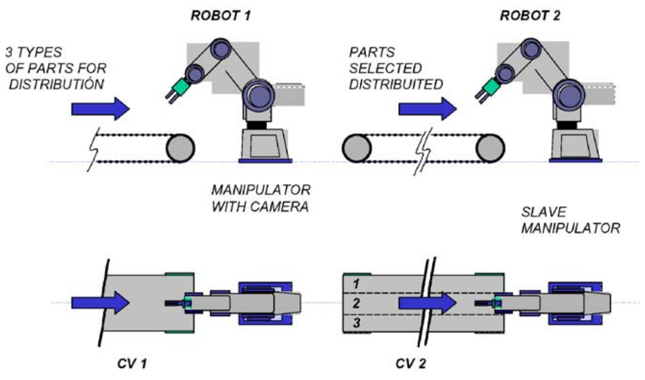


Fig. 4. System to be modelled.

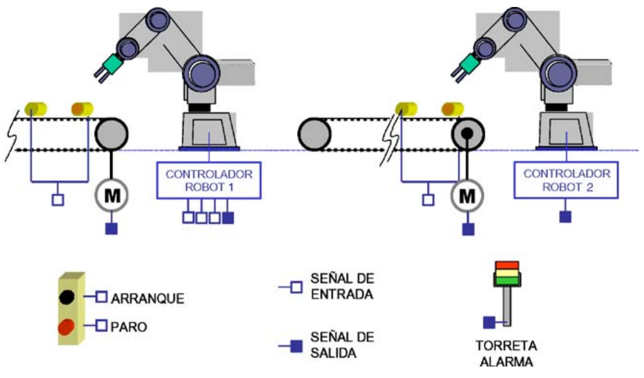


Fig. 5. Signals for the system.

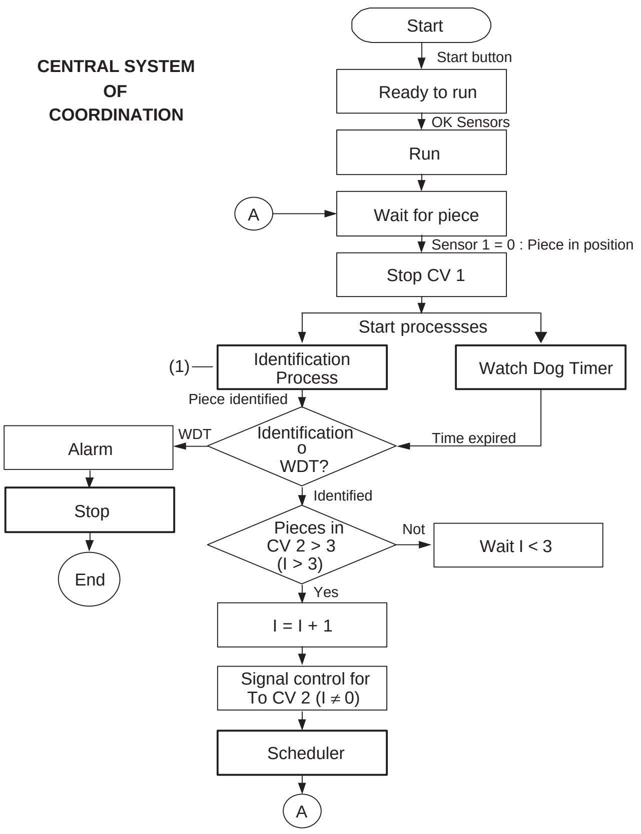


Fig. 6. TAD description.

for the FNRTPN are obtained. By using that information, the structure for the FNRTPN can be created.

Fig. 6 shows the TAD of the transport system. The description is obtained by the application of Woi and Bundell method proposed in [5], and the tool developed in [11]. The ASM description and the FNRTPN model are obtained, and results are shown in Figs. 7 and 8.

The obtained ASM description describes the process task sequence and their conditional decisions, where intelligent control can be applied. Fig. 8 shows the final FNRTPN model, where the specific tasks of the process are defined and the transitions are finite. Each transition

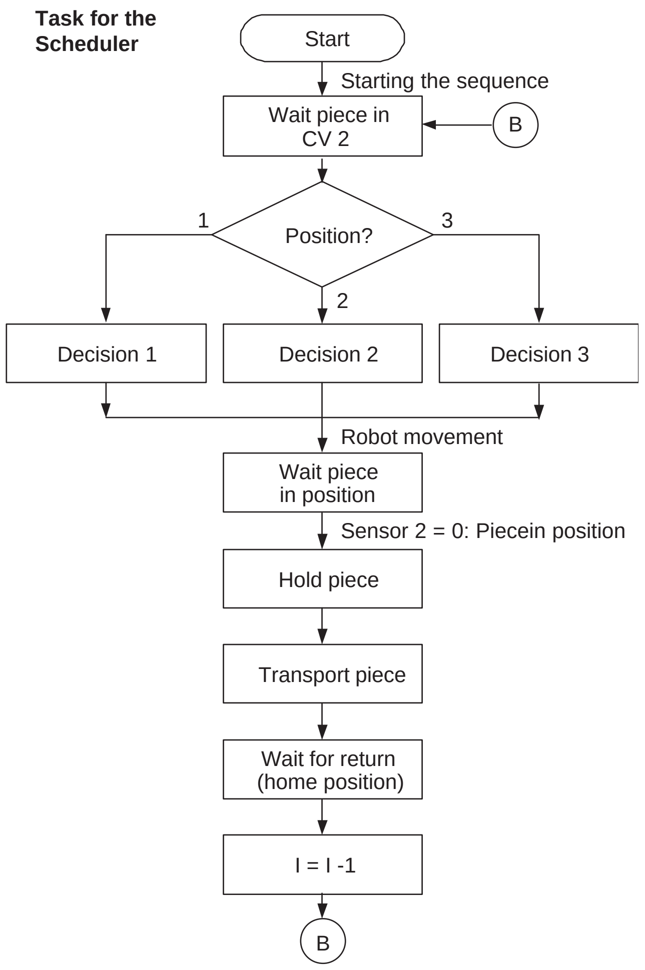


Fig. 7. ASM description.

receives a fuzzy channel f_i for the scheduler, in order to choose the evolution and adequate sequencing, according to the estimated values of the system by the neural part.

Similar transportation tasks for each type of piece can be noticed because they have been defined a priori in a VHDL CORE. The discrete model for the servomotors, cv's and robot dynamics, were modelled using linear discrete state variable equations. The student version BASELINE 10.7 of ALTERA^{TM} was used, and the programming scheme was developed in C language for MS-DOS^{TM}, using a NETLIST definition. Fig. 9 shows the synthetic structural and behavioural description for the FNRTPN model.

The directive entity is used to define the input signals (fuzzy,) and output signals (neural), which are expressed in the PORT command, where all the resources allowed states, and fixed times assigned to specific tasks are given.

Fig. 10 shows the neurofuzzy control description obtained using the proposed method of ILA applied at the FNRTPN model, where the X and Y transposition can be observed and the D's state are determined.

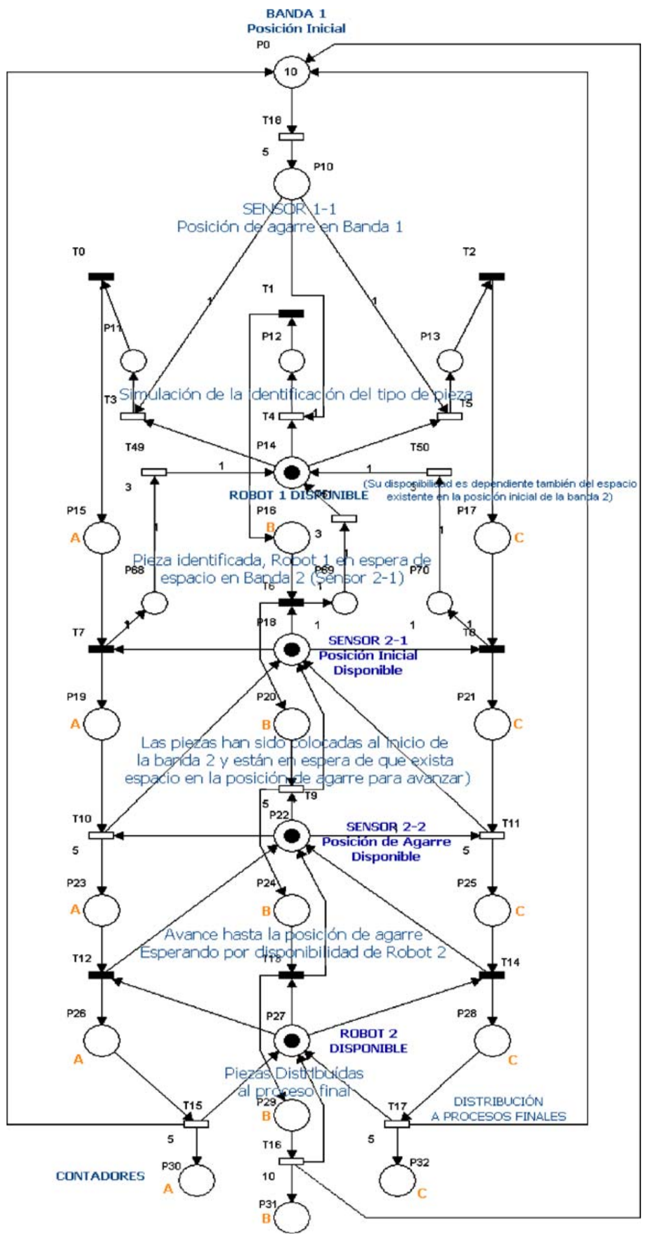

BANDA 1
Posición Inicial
P0
10
T18
P10
SENSOR 1-1
Posición de agarre en Banda 1
T0
T1
T2
P11
P12
P13
Simulación de la identificación del tipo de pieza
T3
T4
T5
T49
P14
T50
ROBOT 1 DISPONIBLE
(Su disponibilidad es dependiente también del espacio existente en la posición inicial de la banda 2)
P15
P16
P17
Pieza identificada, Robot 1 en espera de espacio en Banda 2 (Sensor 2-1)
P68
P69
P70
T6
T7
T8
P18
SENSOR 2-1
Posición Inicial
Disponible
P19
P20
P21
Las piezas han sido colocadas al inicio de la banda 2 y están en espera de que exista espacio en la posición de agarre para avanzar)
T9
T10
T11
P22
SENSOR 2-2
Posición de Agarre
Disponible
P23
P24
P25
Avance hasta la posición de agarre
Esperando por disponibilidad de Robot 2
T12
T13
T14
P26
P27
P28
ROBOT 2
DISPONIBLE
Piezas Distribuidas al proceso final
T15
P29
T17
DISTRIBUCIÓN
A PROCESOS FINALES
T16
CONTADORES
P30
P31
P32
A
B
C

Fig. 8. FNRTPN model.

Despite the direct neural control is defined a priori by the CORE, the final decision to change the PID parameters of the actuators resides on the scheduler. Similar values for the Neural PID were pre programmed and obtained as in [12] for the servo motors and the robots. To evaluate the performance of the FNRTPN and the real evolution of the system, a discrete simulation is showed in Fig. 11.

The adequate evolution between the real performance of the FMS and the control decisions of the scheme obtained by our method can be noticed. The evolution of the real system was evaluated in other environments like ARENA^{TM}, OOPN tools, but the observance of the model and the control in the same platform, was hard to

```
ENTITY fms1 IS
            PORT(
            robot2_ctrl,robot1_ctrl,start,cam_ctrl,cv_m_1,cv_m_2,
            paro,clk: IN std_logic;
            cv_sensor1,cv_sensor2,robot1_sensors,robot2_sensors,
            act_states,cam_signal: OUT std_logic);
END fms1;

ARCHITECTURE description OF fms1 IS
-- Aquí se asignan tiempos a cada transición:
 CONSTANT T1: integer:=1;      -- Estos tiempos han sido
            ...
 TYPE estados IS (p11,p12,p13,p14,p15,p16); -- Def. de estados
 SIGNAL presente: estados:=p1;
 SIGNAL robot2_ctrlaux,camaux,robot1_ctrlaux: std_logic:='0';
 SIGNAL tiempo: integer RANGE 0 TO 16#1FFFF# :=0;
 SIGNAL subtiempo: integer RANGE 0 TO 1023 :=0;
BEGIN
 fms:
 PROCESS(clk)
 BEGIN
  IF clk='1' THEN
   CASE presente IS
   WHEN p1=>
    IF start='11' THEN presente<=p12; END IF;
                        IF paro='1' THEN presente<=p16;
                        END IF;
    cv_sensor1<='0'; cv_sensor2<='0'; robot1_sensors<='1';
    act_states<='0'; robot2_sensors<='0'; camaux<='0';
    tiempo<=0;
    subtiempo<=0;
   WHEN p12=>
    robot1_sensors<='0';
    IF cam_ctrl='0' THEN cv_sensor1<='1';

          ...-- Continua descripcion

   END CASE;
  END IF;
 END PROCESS fms;

 PROCESS(robot1_ctrl)
 BEGIN
  IF robot1_ctrl='1' THEN robot1_ctrlaux<=NOT robot1_ctrlaux;

          END IF;
 END PROCESS;
```

Fig. 9. VHDL description of the FNRTPN model.

realize. Obtained improvements allow the analysis of the discrete dynamics of the process model, and the control decisions can be evaluated. However, some problems related to the resources discrete models were found.

5. Conclusions and future work

The results of the application of the proposed method show improvements in control simplicity, and a better performance when a FMS is modelled, simulated and designed in its control strategy. The intelligent control applied allows to the system self deterministic behaviour, increasing its adjusting and fault tolerance.

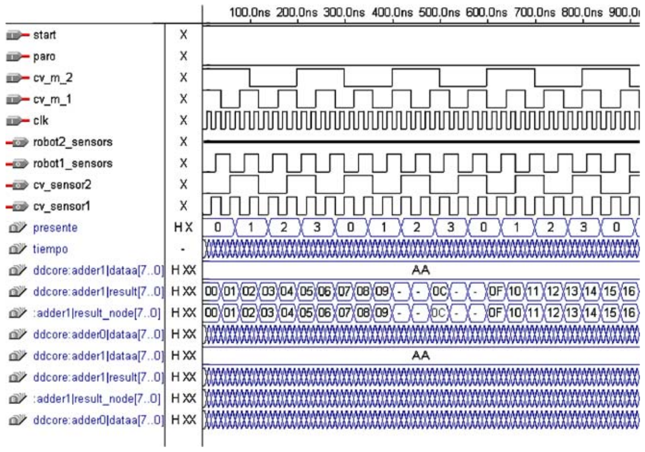
```
ENTITY intctrl IS
PORT (robot2_ctrl,robot1_ctrl,start,cam_ctrl,cv_m_1,cv_m_2,paro,clk: OUT
    std_logic;
    cv_sensor1,cv_sensor2,robot1_sensors,robot2_sensors,act_states,cam_si
    gnal: IN std_logic);
END intctrl;

ARCHITECTURE description OF intctrl IS
-- Scheduler definition
 CONSTANT finite_time: integer:=100;

 TYPE decisiones IS (d1,d2,d3,d4,d5,d6); -- the logistic decisions
 SIGNAL presente: decisiones:=d1;
 SIGNAL robot2_ctrlaux,robot1_ctrlaux,camaux,centriaux: std_logic:='0';
 SIGNAL tiempo: integer RANGE 0 TO 16#1FFFF# :=0;
SIGNAL aux_param: integer RANGE 0 to 256:=0 –for a byte word lenght
BEGIN
 intctrl:
 PROCESS(act_states)
 BEGIN
  IF act_states="101001" THEN
   CASE presente IS
   WHEN d1=> -- the set of rules
          clk<='1';
    IF cv_sensor1='0' THEN presente<=d2; END IF;
          IF cv_sensor2='0' THEN presente<=d6; END IF;
    cv_m_1<='0'; cv_m_2<='0'; robot1_ctrl<='0';robot2_ctrl<='0';
    tiempo<=0;
    subtiempo<=0;
   WHEN d2=>
    robot1_sensors<='0';
    IF cam_signal='0' THEN cam_ctrl<='1';
    ELSE  fuzzifier(paro);
    END IF;
    ...
   END CASE;
  END IF;
 END PROCESS ctrl;
```

(a)

```
 PROCESS(robot1_sensors)
 BEGIN
  IF robot1_sensors='1' THEN centriaux<=NOT centriaux;

    END IF;
 END PROCESS;
                        PROCESS(cv_sensor1)
 BEGIN
  IF robot2_ctrl='1' THEN robot2_ctrlaux<=NOT robot2_ctrlaux;

    END IF;
 END PROCESS;

 PROCESS(fuzzifier[aux_param])
 BEGIN
  Signal in_var:INTEGER 0 to 256:=0;

BEGIN
In_var< = aux_param;
 IF reset = '1' THEN
memb_var <= (Others => ' 0 '); --All the bits are set to zero
ELSIF clock'EVENT and (clock = '1') THEN
       IF (in_var <tm1m) THEN
            tmp := conv_std_logic_vector(unsigned(unsigned(in_var) -
               unsigned(start))*unsigned(delta), 8);
               memb_var(7 downto 0) <= tmp;
              memb_var (set_nr*8-1 downto 8) <= (Others => '0');
       ELSIF (in_var <tm2m) THEN
              tmp := conv_std_logic_vector(unsigned(unsigned(in_var) -
                 unsigned(tm1m))*unsigned(delta), 8);
             memb_var(7 downto 0) <= conv_std_logic_vector(1-
unsigned(tmp),8);
             memb_var(15 downto 8) <= tmp;
             memb_var (set_nr*8-1 downto 16) <= (Others => '0');
             ELSIF (in_var < tm3m) THEN
                          ...
END PROCESS;
END IF;
 END PROCESS;
END description;
```

(b)

Fig. 10. Partial VHDL description of the neurofuzzy control scheme (a). Partial VHDL description of the neurofuzzy control scheme (b).



100.0ns 200.0ns 300.0ns 400.0ns 500.0ns 600.0ns 700.0ns 800.0ns 900.0
start
paro
cv_m_2
cv_m_1
clk
robot2_sensors
robot1_sensors
cv_sensor2
cv_sensor1
presente
tiempo
ddcore:adder1|dataa[7..0]
ddcore:adder1|result[7..0]
:adder1|result_node[7..0]
ddcore:adder0|dataa[7..0]
ddcore:adder1|dataa[7..0]
ddcore:adder1|result[7..0]
:adder1|result_node[7..0]
ddcore:adder0|dataa[7..0]
AA

Fig. 11. Simulation.

Therefore, the method proposed a new extension of PN, with a better link between the real FMS and the intelligent control scheme. The project goals have been fulfilled, and the complete environment is working in the CIM-2000 of the Instituto Tecnológico de Puebla, where the intelligent control system applied shows an improved performance for the programmed activities.

The proposed method demonstrates the feasibility of the application of novel control strategies, which can be restricted by the use of discrete event models of process using discrete state variables for the elements and devices. The robustness and stability of the intelligent control generated must be analysed in future projects.

Acknowledgements

The authors want to express thanks to Dr. Dewi Jones and the Instituto Tecnológico of Puebla Laboratories, for their valuable help. The present work was financially supported by the *Asociación Nacional de Universidades e Instituciones de Educación Superior*, the Consejo del Sistema Nacional de Educación Tecnológica and the *Dirección General de Institutos Tecnológicos*.

References

[1] Shen W, Norrie DH. Agent-based systems for intelligent manufacturing. A state-of-the-art survey. Knowledge Inform Syst Int J 1999;1(2):129–56.

[2] Gregor E. A universal modelling language. 21st International conference on applications and theory of Petri nets, Aarthus, Denmark, June 26–30, 2000.

[3] Zhou M, Venkatesh K. Modelling, simulation and control of FMS—a Petri net approach. Singapore: World Scientific; 1999.

[4] Zimmermann A. Modelling of manufacturing systems and production routes using colored Petri nets. International Proceedings of the Third IASTED, International conference on robotics and manufacturing, Cancún, México, 1995, p. 380–3.

[5] Woi LA, Bundel GA. Analysis of a flexible manufacturing system task controller software model using hierarchical timed Petri nets. Information Systems Engineering Research Group, University of Western Australia, 1999.

[6] Van der Aalst. Modelling and analysis of production systems using a Petri net based approach. Eindhoven University of Technology, The Netherlands, 1998.

[7] Aguado BA. Temas de Identificación y Control Adaptable. La Habana, Cuba: ICIMAF; 2000.

[8] Salapura V, Hamann V. Implementing fuzzy control systems using VHDL and statecharts. Austria: Technische Universitat Wien; 2001.

[9] Gupta P. Hardware–software codesign. IEEE Potentials 2001/2002.

[10] Bundell GA. An fpga implementation of the Petri net firing algorithm. Information Systems Engineering Group, University of Western Australia, 2000.

[11] Torres GHH. Análisis y modelado de un sistema de manufactura flexible usando redes de Petri Jerárquico Temporizadas. MSc thesis. Instituto Tecnológico de Puebla, 2003.

[12] Gracios C, Muñoz G, Estévez J, Torres S. Neuro-PID control system with gravity compensation. Proceedings of the third IEEE international symposium on robotics and automation, Toluca, Edo. Mexico, 2002.

[13] Jiang Ch, Zheng Y. Fuzzy reasoning based on Petri nets. New York: IEEE Press.

[14] Ojala T. Neuro–Fuzzy systems for control. MSc thesis. Tampere Technical University, Finland, 1995.

[15] Myong-Gyun R, Sang-Eun H. Control and monitoring of factory automation system using Fuzzy Petri nets. IEEE, International symposium on industrial electronics, Pusan, Korea, 2001.

[16] Dadone P. Fuzzy control of flexible manufacturing systems. MSc thesis. Blacksburg, VA, 1997.

[17] Chen P, Forward K. Fuzzy Petri nets. First international conference on knowledge-based intelligent electronic systems, Adelaide, Australia, 1997.