Available online at www.sciencedirect.com

ScienceDirect

IFAC-PapersOnLine 49-12 (2016) 1008–1013

Task-based Coordination of Flexible Manufacturing Cells using Petri Nets and ISA standards

E.G. Hernández-Martínez*, Erika S. Puga-Velazquez*, Sergio A. Foyo-Valdés*, J.A. Meda Campaña**

**Engineering Department, Universidad Iberoamericana Ciudad de México, C.P. 0121, México, D.F.(e-mail: eduardo.gamaliel@ibero.mx, erika_selenep@hotmail.com sergiofoyo@live.com.mx).*
*** Mechanical Engineering Department, Sección de Estudios de Posgrado e Investigación, ESIME Zacatenco, Instituto Politécnico Nacional, C.P. 07738 México, D.F (e-mail: jmedac@ipn.mx)*

Abstract: This work presents an approach to describe the event-based coordination of standard Flexible Manufacturing Cells using Petri Nets and its translation to local controllers with computer-based supervision. The plant model is constructed by the interconnection of individual Petri Net models describing the relationships of equipment and storages with process tasks and their logical precedence restrictions, according to the suggestions of the ISA-95 standard. The modeling is generic and scalable containing the possible restrictions about concurrent production routes, equipment availability, storage limitations, sharing resources, etc. A procedure about the codification of the Petri Net model into a software application is presented. The result is a hierarchical setup, where the process tasks can be programmed in local controllers, like PLC's networks. These tasks are communicated and coordinated by a software application using the Petri Net model. A sequence of tasks is obtained by the firing of transitions of the Petri Net model and the system becomes flexible producing different and concurrent products. The approach is tested in a prototype of automated manufacturing cell using a PLC and the development of a software application in Matlab®, which can be extended to other industrial manufacturing cells.

Keywords: Petri nets, Automated Manufacturing Cells, ISA standard, PLC, Supervisory Control.

1. INTRODUCTION

Flexible Manufacturing Cells (FMC's) is a set of equipment, storages and material-handling devices working together in order to assemble different and concurrent products (Groover (2014)). According to the standard ISA-95 (Mazak (2015)), the coordination layer termed Supervisory Control and Data Acquisition must realize the strategic execution of low-level tasks to make different products concurrently using all the available resources. The ISA-95 recommends the separation of the plant model and the sequence of tasks for a specific product given by a product recipe. The main challenge of the coordination control is to increase the productivity exploiting the flexibility of the FMC. The coordination must consider the precedence conditions of tasks, storage limitations, pre-programmed routines of machined and assembly stations, the transportation routes of workpieces or final products and the online communication with the network of local controllers.

The Petri Net (PN) formalism has been widely studied in the modelling and control of industrial automated systems (Li (2014)). Firstly, in the design of sequential routines of electrical, pneumatic and hydraulic devices and its translation to Programmable Logical Controllers (PLC's) (Estrada-Vargas (2011) and Basile (2013)). Secondly, in the coordination layer studying the concurrency, blocking, fault detections and time optimization of routines in (Hu (2013) and Hu (2015)). Since the most of works focus the PN to specific FMC configurations or the mathematical analysis of simple examples, formal methodologies about the systematic modelling in combination with industrial standards has been little explored. These combinations constitute new research issues for the industrial field, since the guidelines of industrial standards inspire and impose new features and properties of PN models. Some preliminary efforts are given in (Petin (2007)), where the supervisory synthesis procedure is combined with the standard IEC61499. The PN modelling of disassembling process of electronic products is presented in (Dutta (2008)). Batch production scheduling with PN is given in (Sanchez (2010) and Gradišar (2012)) focus the combination of finite state automata and the ISA-88 standard in batch production. In all the previous works, the modelling applies to specific setups of automated systems and they do not address the procedure to construct systematically the plant model for any FMC and its application to automation software.

This paper presents a methodology to model a general FMC inspired on the ISA-95 standard through the definition and interconnection of individual PN models. These models contain the information about the availability and capacity of storages and equipment mixed with the process tasks and its logical precedence conditions. The methodology was preliminarily presented in (Hernandez-Martinez (2013)) and applied for the case of AGV´s coordination in our previous work in (Hernandez-Martinez (2015)). The main contribution of this paper is to present a new case of study and to complete the approach with the steps to generate the PN models and the codification of the incidence matrix in a software of

10.1016/j.ifacol.2016.07.574

supervision. Then, the PN model and the sequence of firing of transitions according to product recipes, become in a computer-based supervision with online communication to the network of local controllers. Therefore, the approach encompasses since the formal modelling until the real implementation of the supervision, providing the PN advantages to the engineering practice. The preliminary software is the fundamental element in the development of a PN-based integration software of FMS. The approach is tested in a prototype of FMC controlled by PLC with OPC (OLE for Process Control) server communicated to a software application developed in Matlab®.

2. FMC MODELING FRAMEWORK

According to the ISA-95 standard, the plant model is obtained from the definition and interconnection of basic models of sets of equipment (E), process tasks (PT), storage (A) and precedence restrictions (DL).

The equipment is the set $E = \{E_1, \dots, E_n\}$ of entities responsible of executing process tasks, such as robots, conveyor belts, machines, assembly stations, etc. Each $\mathrm{E_i}$, $i = 1, \dots, n$ is available in a $\mathrm{C(E_i)}$ quantity. The set $H(E_i) = \{T_{i1}, \dots, T_{ik_i}\}$ contains the possible k_i tasks performed by the equipment $\mathrm{E_i}$. Thus, the set $PT = H(E_i) \cup \dots \cup H(E_n)$ encompasses all the tasks in the system. The start and end of each task T_{ij} are given by the transitions sT_{ij} and fT_{ij}, respectively. These transitions are also grouped in the general sets $sT = \{sT_{ij}\}$, $\forall T_{ij} \in PT$ and $fT = \{fT_{ij}\}, \forall T_{ij} \in PT$.

In the other hand, the set of storages is given by $A = \{A_1, \dots, A_h\}$, where the capacity of the storage A_ℓ, $\ell = 1, \dots, h$ is given by $C(A_\ell)$ and the sets of transitions for the load and unload of the storage A_ℓ are given by $U_{in}(A_\ell)$ and $U_{out}(A_\ell)$, respectively. Depending on the type of transitions contained in $U_{in}(A_\ell)$ and $U_{out}(A_\ell)$, three types of storages are obtained:

a) Manual load-Automatic unload (ML-AU) storages, like dispensers of raw material, where $U_{in}(A_\ell) \in mT_{in}$, with $mT_{in} = \{mT_{in_1}, \dots, mT_{in_q}\}$ is the set of (entry) manual transitions and $U_{out}(A_\ell) \in sT$.

b) Automatic load-Manual unload (AL-MU) storages, like the storages of final product, where $U_{in}(A_\ell) \in fT$, $U_{out}(A_\ell) \in mT_{out}$, with $mT_{out} = \{mT_{out_1}, \dots, mT_{out_\eta}\}$ as the set of out manual transitions.

c) Automatic load-Automatic unload (AL-AU) storages, for intermediate storages where the subparts are placed temporally for the material-handling equipment, where $U_{in}(A_\ell) \in fT$, $U_{out}(A_\ell) \in sT$.

According to the PN structure defined in Cassandras (2008), the whole PN model can be constructed by

$$PN = (P, T, F, W, M_0) \quad (1)$$

where

- $P = E \cup PT \cup A \cup DL$ is the set of places.
- $T = sT \cup fT \cup mT_{in} \cup mT_{out}$ is the set of transitions.
- $F = (E \times sT) \cup (fT \times E) \cup (A \times sT) \cup (fT \times A) \cup (mTin \times A) \cup (A \times mT_{out}) \cup (sT \times PT) \cup (PT \times fT) \cup (fT \times DL) \cup (DL \times sT)$ is the set of arcs connecting places to transitions and vice versa.
- $M_0 = [M_0(E), M_0(PT), M_0(A), M_0(DL)]$, is the initial marking.

As will be presented below, the set DL constitutes all the logical dependence restrictions of tasks according to the correct functional flow of the FMC. They establish the necessary precedence, concurrence or divergence restrictions according to the physical conditions of the system. The general scheme of the FMC plant model is presented in the Fig. 1, where the places of equipment, the three type of storages, process tasks, precedence restrictions, and the possible arcs defined in (1) can be visualized. Note that availability of equipment and the capacities of the storages are represented by tokens, and the storages are assumed as initially full or empty. In a global perspective, the PN does not contain self-loop, is non-ordinary, it has finite capacity and bounded with has vivacity and reversibility properties guaranteeing the dynamical evolution of the FMS.

The PN models of equipment, storages and process tasks are given in the Fig. 2. The equipment E_i (Fig. 2a) is a place bounded with capacity $K(E_i) = C(E_i)$, and initial marking $M_0(E_i) = C(E_i)$, that represent the available quantity of items in the system. The start or end transitions of the task contained in its set $H(E_i)$ take or return tokens respectively, in E_i modifying the availability of the equipment. Similar case is the model of the storages in Fig. 2b, where the transitions of the set $U_{in}(A_\ell)$ and $U_{out}(A_\ell)$, put or take tokens in the place respectively, according to the type of storage. The weights w_{in} or w_{out} of the arcs establishes the number of pieces that arrive or leave the storage. The capacity of the place A_ℓ is given by $\mathrm{K}(\mathrm{A}_\ell) = C(A_\ell)$ and the initial marking is $\mathrm{M_0}(\mathrm{A}_\ell) = 0$, when the storage is initially empty or $M_0(A_\ell) = C(A_\ell)$ when is initially full.

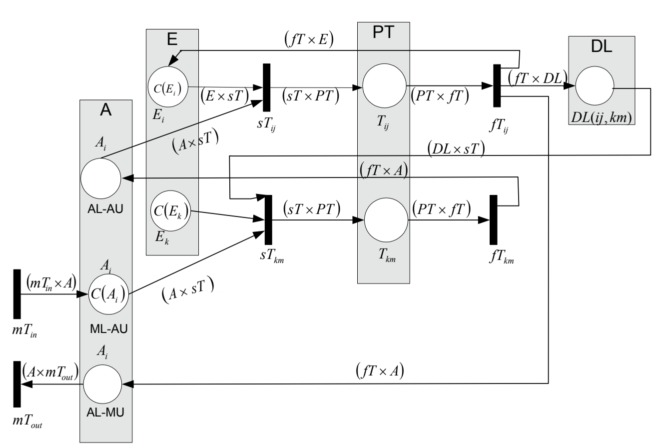
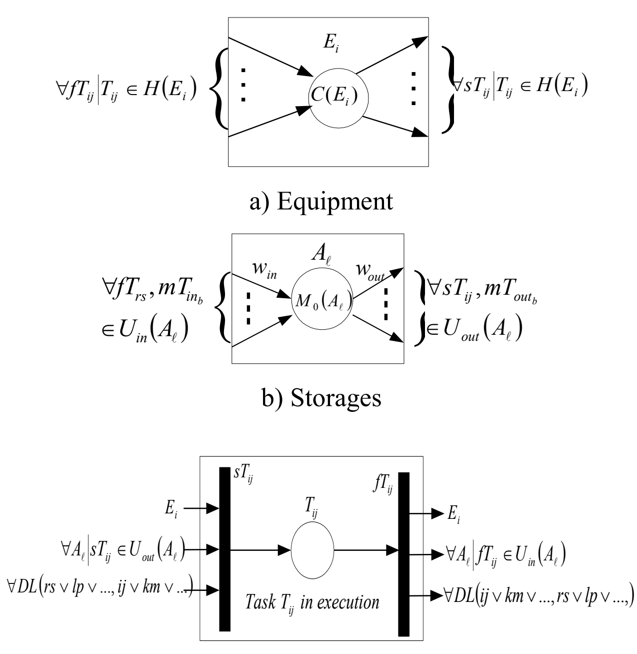


Fig 1. General PN plant model

The central actor of the PN plant model is the process task model shown in the Fig. 2c. The firing of the start-task transition is conditioned by the availability of the equipment, resources in the storages, and the compliance of the precedence tasks. If the task is being executed, a token is put in the place T_{ij}. At the end of the task, the end-transition is firing, and some tokens can be returned to the availability of the equipment. In some cases, new tokens are carry out to the storages and the end-task transition enables the next logical dependence conditions for a future task. A task T_{ij} can be executed concurrently, according to the number of equipment E_i in the system, therefore the capacity of the place T_{ij} is $K(T_{ij}) = C(E_i)$, and the initial marking is $M_0(T_{ij}) = 0$.

The models of DL are presented in the Fig. 3. According to the ISA-95 standard, can exist two kinds of precedences. Direct logical precedence conditions (DL-direct Fig. 3a), occurs when the final of a task T_{ij} enables the beginning of a subsequent task T_{rs}, in the normal functional flow. The diagram represents that the *nf* amount of end-tasks of T_{ij} enables the *ns* starts of T_{rs}. On the other hand, Inverse logical precedence restrictions (DL-inverse Fig 3b), appears when the finish of a posterior task enables again the start of a prior task. For example, when the end of a lathe machining enables the feeding of a new part again. Note that the DL-direct or DL-inverse are identified by continuous or dotted lines respectively, and the initial marking of the DL-inverse is different from zero, in order to enable the condition in a first time in the functional flow. The DL-direct and DL-inverse shown in the Fig. 3, can be extended to the precedence of multiple tasks. For example, multiple tasks could enable the activation of a single task (convergence), or the end of a single task could enable the start of multiple tasks (divergence). In the example of the next section, the possible multiple precedence conditions are shown for the sake of clarity.



a) Equipment

b) Storages

c) Process task

Fig. 2. PN models of equipment, storages and process tasks

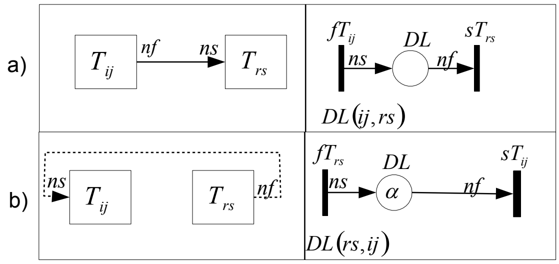


Fig. 3. PN models of a) DL-direct b) DL-inverse

In order to a illustrate the PN modelling framework, in the next section the case of study is described and modelled to obtain the PN plant model.

3. DESCRIPTION AND MODELING OF THE FMC

Fig. 4 presents the FMC of the case of study. It is a electrical-pneumatic training system for PLC manufactured by Fishertechnik® that reproduce in small-size scale the behaviour of an automated system. The FMC is composed by four interconnected modules that moves the workpieces by conveyor belts and a cylindrical robot to four possible processes: punching (P1), milling (P2), drilling (P3) and inspection (P4). The products differ in the sense that each workpiece can be carry out by one or more of these manufacturing operations. The local controller includes a Siemens PLC S7-1200 with Ethernet communication, using the OPC server called Kepserver, to enable the communication of tags to Matlab® and other software under the Windows OS.

The system is classified from E_1 to E_7 equipment modules according to the Fig. 4. E_1 contains the raw material storage A_1, a turning table (*TT*), the process P_1 and the conveyor feeder CF_1 of the conveyor belt B_1. The module E_2 is dedicated to move the workpieces by the conveyor belts B_1 and B_2 and the conveyor feeder CF_3. The module E_3 uses the conveyors B_3, B_4 and B_5 and CF_5 to move pieces near of the process P_2 and P_3. The modules E_4 and E_5 contain the elements to realize the operations of P_2 and P_3, respectively. The module E_6 is composed by a cylindrical-type robot manipulator to move the pieces between the conveyors B_5 and B_6 to B_7 or B_8, where the final product conveyors leaves the system (in fictional output storages A_2 and A_3). Table 1 summarized the equipment modules and their respective tasks, which are depicted in the Fig. 4, where the transportation tasks are represented by arrows.

For the correct functional flow in the FMS, some logical precedence restrictions (D-direct or D-inverse) need to be established between the tasks defined in the Table 1. They can be represented in the precedence diagram shown in the Fig. 5. The processes P_1 to P_4 are highlighted in blue colour. For example, the D-direct *D2=D(21,31)* (continuous red colour) establishes that the task T_{31} can not be executed until task T_{21} has finished (Conveyor B_3 requires the presence of a piece to start the transportation). Other example is the D-inverse *DI4=DI(33,32)* (dotted red colour) where the end of the task T_{33} enables again the task T_{32} (Conveyor B_4 is enabled to move a new piece in front of P_3, until the previous piece has been removed).

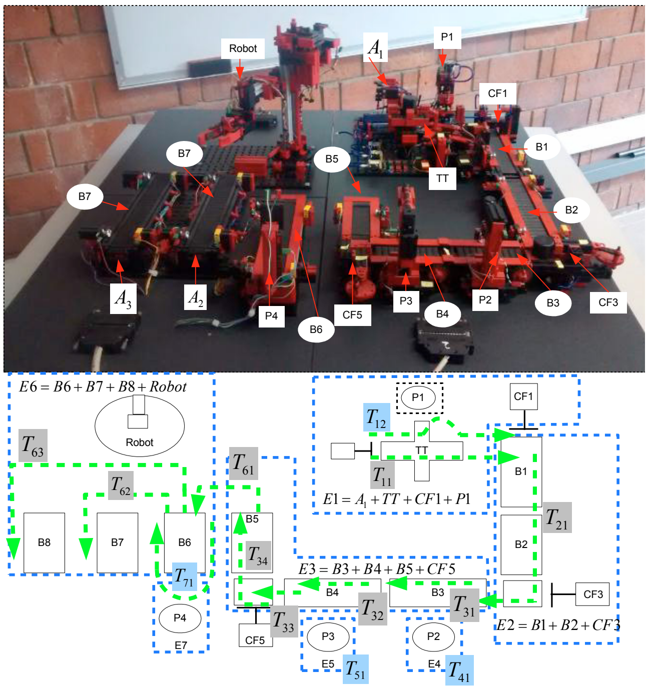


Fig. 4. Photo and scheme of the FMC

Table 1. Equipment modules and description of tasks

Eq.	Task	Description
E_1	T_{11}	TT moving from A_1 to CF_1
	T_{12}	TT moving from A_1 to CF_1 realizing P_1
E_2	T_{21}	B_1, B_2 and CF_3 move a piece to B_3
E_3	T_{31}	B_3 moves a piece to P_2
	T_{32}	B_3 and B_4 move a piece from P_2 to P_3
	T_{33}	B_4 moves a piece from P_3 to CF_5
	T_{34}	A piece is moved from CF_5 to the end of B_5
E_4	T_{41}	Milling process P_2
E_5	T_{51}	Drilling process P_3
E_6	T_{61}	Robot moves a piece from B_5 to B_6
	T_{62}	Robot moves final product from B_6 to B_7
	T_{63}	Robot moves final product from B_6 to B_8
E_7	T_{71}	B_6 moves pieces to the process P_4 and return

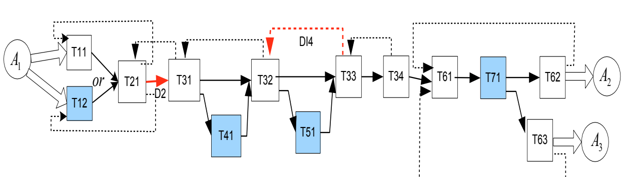


Fig. 5 Precedence diagram of the case of study

Applying the translation method to PN models of the Section 2 and considering the precedence restrictions in Fig. 5, the PN plant model of the FMC is depicted in the Fig. 6. Note that the models of storages, equipment, tasks and precedence restrictions are ordered and easily detected. Fig. 6 identifies by red boxes some examples of models of equipment (a), task (b), D-inverse (c), D-direct (d), ML-AU storage (e) and AU-ML storage (f). The places related to the processes P_1 to P_4 are highlighted in blue colour.

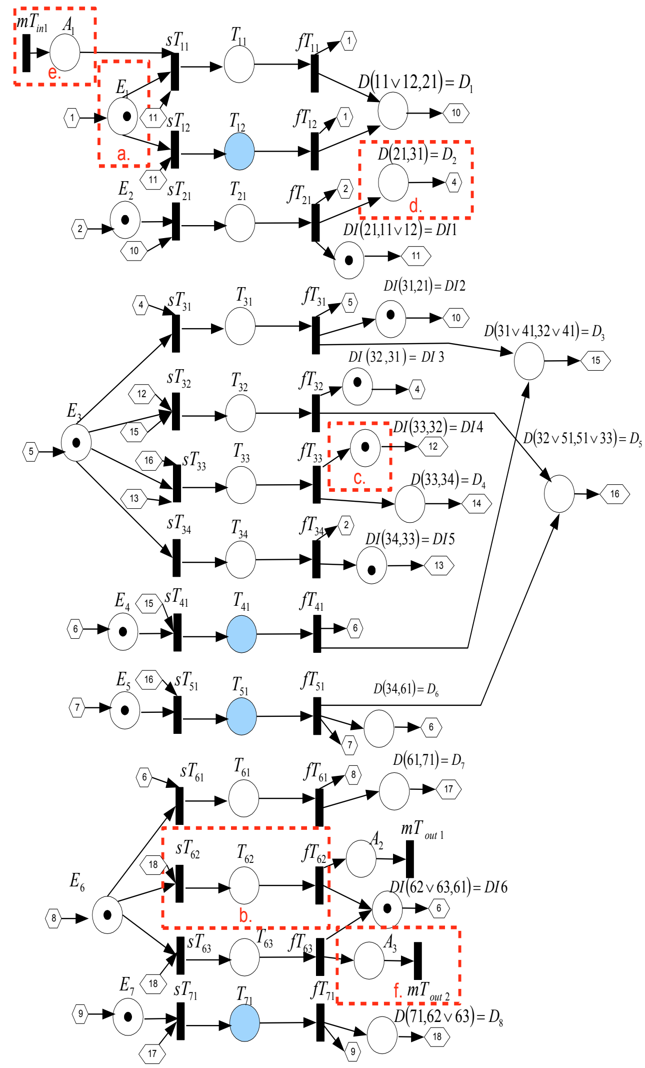


Fig. 6. Final PN of the case of study

4. IMPLEMENTATION IN SOFTWARE APPLICATION

The PN plant model obtained in the Section 2 can be implemented in a software application that becomes the fundamental item of a SCADA. The main purpose is to communicate online the firing of the PN transitions to the network of local controllers. Fig. 7 summarizes the blocks of the software implementation. In a first step, in the block a) the user generates an equipment and task decomposition according to the ISA-95 standard, providing the information about the storages, equipment and tasks in the FMC. Also, a precedence diagram is defined to restrict the logical order of the process flow, like the example shown in the Fig. 5.

The previous information can be coded in the block of the Fig. 7b in a configuration file called "Petri Net File" (PNF).

As expected, the PNF can be typed in three sections, according to the Fig. 8. The first section (Fig. 8a) is related to the equipment and their respective tasks. The second section of the PNF (Fig. 8b) is related to the information about storages, its number of pieces and the tasks that load and unload each of them. Finally, the D-direct and D-inverse precedence restrictions are coded in the third section (Fig. 8c) between the pair of tasks in each precedence condition.

Returning to the Fig. 7, the information of the PNF is used to construct automatically the incidence matrix in the block c) of the Fig. 7, of the PN plant model, according to the individual PN models and their possible interconnections described in the Section 2. A general scheme about the elements in the incidence matrix is presented in the Fig. 9. The places constitute the rows and the transitions generate the columns. For example, the first block is related to the arcs joining the start-task transitions with the task's places in the PN. Note that all the matricial blocks shown in the Fig. 9 coincide with the types of arcs shown in the Fig. 1.

On the other hand, the user can construct a list of desired tasks of the FMC in order to manufacture a product. This information can be saved in a "REC file" (Fig. 7e), which can be converted to the list of desired transitions to be fired in the PN plant model. Therefore, using the obtained incidence matrix (Fig. 7d) and the recipe of product, the block of the Fig. 7f implements the possible firing sequences using the standard rules of enabling and firing of PN (Cassandras (2008)).

The user can visualize the possible transitions online through a HMI (Human Machine Interface). Thus, according to the product recipe (or in manual activation), the start-task transitions can be enabled by the user whereas the end-task transitions are only read when a task has finished modifying the dynamics of the PN.

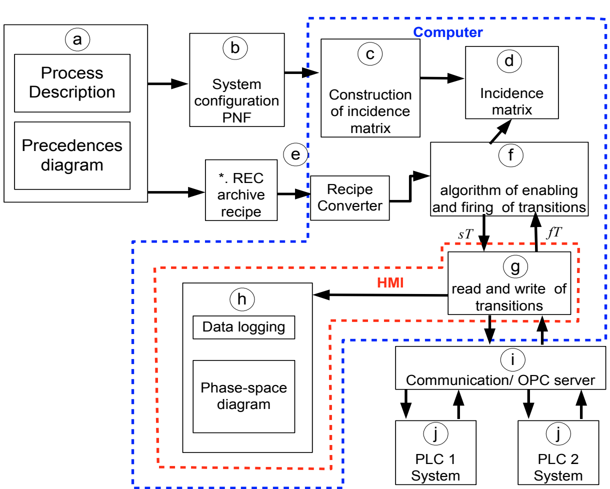


Fig. 7. General scheme of the approach implementation

Data logging about the transitions, tasks, product status and visualization in monitor of the phase-space diagram (record of the executed tasks for a product) can be displayed in the computer (Fig. 7h). Finally, using an OPC server (Fig. 7i and 7j), the transitions can be communicated to the local controllers, where the tasks are being previously programmed, communicating the status of each tasks to the software. The OPC communication allows the connection to different platforms of industrial controllers under Windows OS using different communication protocols used in real FMC's, like Ethernet, Modbus, CANbus, Profibus, etc.

Equipment	Quantity	belonging tasks
$E_i \in E$	$C(E_i)$	$T_{i1} \in H(E_i)$
		$\vdots$
$i = 1 \ldots n$		$T_{ik_i} \in H(E_i)$

a) Equipment section

Storage	capacity	Input tasks	Number of pieces entry	Output tasks	Number of pieces departure
$A_\ell \in A_1$ case AL-AU	$C(A_\ell)$	$T_{ij} \mid fT_{ij} \in U_{in}(A_\ell)$	ω_{in}	$T_{ij} \mid sT_{ij} \in U_{out}(A_\ell)$	ω_{out}
$A_\ell \in A_2$ case ML-AU	$C(A_\ell)$			$T_{ij} \mid sT_{ij} \in U_{out}(A_\ell)$	ω_{out}
$A_\ell \in A_3$ case AL-MU	$C(A_\ell)$	$T_{ij} \mid fT_{ij} \in U_{in}(A_\ell)$	ω_{in}		

b) Storage section

precedence	Input tasks	W input	output tasks	W output
direct $DL(ij,km)$	$T_{ij} \mid fT_{ij} \in fT$	ns_x	$T_{km} \mid sT_{km} \in sT$	nf_y
inverse $\overline{DL}(ij,km)$	$T_{ij} \mid fT_{ij} \in fT$	ns_x	$T_{km} \mid sT_{km} \in sT$	nf_y

c) Precedence conditions section

Fig. 8 Information in the PNF

	sT	fT	mT_{in}	mT_{out}
PT	$(sT \times PT)^T$	$(PT \times fT)$	0	0
E	$(E \times sT)$	$(fT \times E)^T$	0	0
$AL-AU$	$(A \times fT)$	$(fT \times A)^T$	0	0
$ML-AU$	$(A \times fT)$	0	$(mT_{in} \times A)^T$	0
$AL-MU$	0	$(fT \times A)^T$	0	$(A \times mT_{out})$
DL	$(DL \times sT)$	$(fT \times DL)^T$	0	0
$\overline{DL}$	$(\overline{DL} \times sT)$	$(fT \times \overline{DL})^T$	0	0

Figure 9. Blocks in the incidence matrix

For the case of study, the PN implementation was programmed in Matlab®, using the OPC server toolbox in Simulink®, linked to Kepware OPC, developed by Kepware Technologies®. It was online communication with the Siemens S7-1200 PLC, the local controller of FMC. The developed HMI is shown in the Fig. 10. It includes sections of the PNF loading, which displays the information about the FMC of the case of study, and the list of transitions in the current time instant. The user can execute some start-task transition select it from the list. Finally, an example of a product sequence is presented in the Fig. 11, where the Matlab® application displays the record of the executed tasks in the FMC.

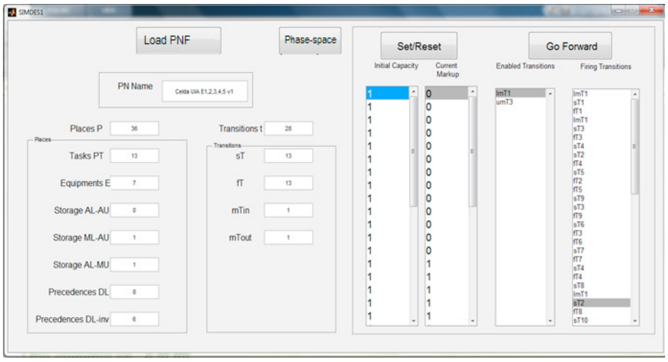


Figure 10. HMI of the software application

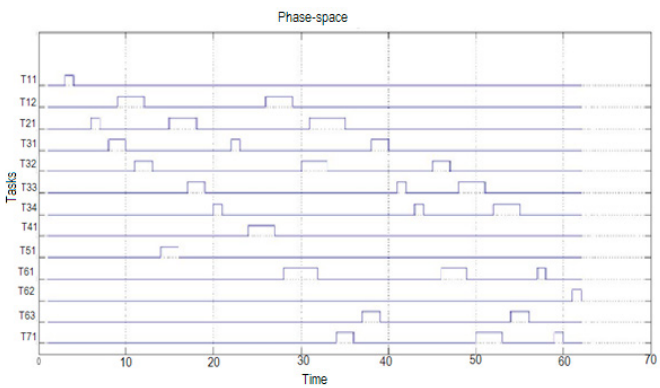


Figure 11. Phase-space diagram of the case of the study.

6. CONCLUSIONS

This work presents an approach for the modelling and supervision implementation of the FMC's coordination using PN. An equipment classification and task assignment is realized according to the ISA-95 standard. Then, some generic PN models about storages and equipment availability, process tasks and their logical precedence restrictions are defined and interconnected to obtain a PN plant model. The framework allows the possibility to modify the amount of equipment or storage limitations without changes in the network topology, preserving its static properties. The product recipes can be translated to sequences of transitions allowing the concurrence of different products.

The paper includes the procedure to translate the incidence matrix of the PN in a software application, where the user provides only the general information of the FMC and the product recipes, and the software codifies the PN using the incidence matrix to enable the tasks according to the product recipe. The software is communicated to a network of local controllers of the FMC, where the tasks are being previously programmed. The approach is tested in a prototype of FMC and the software application is programmed in Matlab® with OPC communication to a PLC. The software application serves as the base of an integration software of automated systems. The approach is a systematic method to close the concepts of Discrete-event systems in an engineering manufacturing context.

REFERENCES

Basile F.; Chiacchio P.; Gerbasio D. (2013), On the Implementation of Industrial Automation Systems Based on PLC, *IEEE Transactions on Automation Science and Engineering*, Volume 10, No 4, pp 990-1003.

Duta L.; Filip F.G. (2008), Control and Decision-making Process in Disassembling Used Electronic Products, *Studies in Informatics and Control*, Volume 17, No. 1, pp 17-26.

Estrada-Vargas A.P.; Lesage J., Lopez-Mellado E. (2011), Stepwise identification of automated discrete manufacturing systems, *IEEE 16th Conference on Emerging Technologies & Factory Automation,* pp 1-8.

Gradišar D.; Mušič G. (2012), Petri Nets - Manufacturing and Computer Science, *InTech,* edited by Pawel Pawlewski, ISBN 978-953-51-0700-2, pp. 5-26.

Groover M.P; (2014), Automation, *Production Systems and Computer Integrated Manufacturing*, 4th Ed., Pearson.

Hernandez-Martinez E.G.; Foyo-Valdés S.A.; Puga-Velazquez E.S.; Meda Campaña J.A. (2015), Motion Coordination of AGV's in FMS using Petri Nets, *IFAC INCOM,* pp. 196-201.

Hernandez-Martinez E.G.; Puga-Velazquez E.S; Foyo-Valdés S.A.; Meda Campaña J.A. (2013), Modeling Framework for Automated Manufacturing Systems based on Petri Nets and ISA Standards, *Studies in Informatics and Control*, Volume 22, no. 2, pp. 163-174.

Hu H.; Zhou Z.; Li Z.; Tang Y. (2013), Deadlock-Free Control of Automated Manufacturing Systems With Flexible Routes and Assembly Operations Using Petri Nets, *IEEE Transactions on Industrial Informatics*, Volume 9, Issue 1, pp 109-121.

Hu H.; Zhou M. (2015), A Petri Net-Based Discrete-Event Control of Automated Manufacturing Systems With Assembly Operations, *IEEE Transactions on Control Systems Technology*, Volume 23, Issue 2, pp 513-524.

Li J.; Zhou M.; Guo T.; Gan Y.; Dai X. (2014), Robust control reconfiguration of resource allocation systems with Petri nets and integer programming, *Automatica* Volume 50, Issue 3, pp. 915–923.

Mazak A; Huemer C; (2015), HoVer: A modelling framework for horizontal and vertical integration, *IEEE International Conference on Industrial Informatics*, pp 1642-1647.

Petin J.F.; Gouyon D.; Morel G. (2007), Supervisory synthesis for product-driven automation and its application to a flexible assembly cell, *Control Engineering Practice*, Volume 15, Issue 5, pp 595-614.

Sanchez A.; Aranda-Bricaire E.; Jaimes F.; Hernandez E.; Nava A. (2010), Synthesis of product-driven coordination controllers for a class of discrete-event manufacturing systems, *Robotics and Computer-Integrated Manufacturing*, vol. 26, no. 4, pp. 361-369.