Technical University of Cluj-Napoca
ACTA TECHNICA NAPOCENSIS

Series: Applied Mathematics, Mechanics, and Engineering
Vol. 58, Issue III, September, 2015

CONTROL OF FLEXIBLE MANUFACTURING SYSTEMS

Liana MIC, Raluca FAT, Tiberiu LETIA

Abstract: *Many applications are of hybrid type, which means that they consist of two parts: a discrete events and a discrete time (continuous) part. The control synthesis for these kinds of systems is difficult because they involve interactions of different types of models belonging to different approaches. The control components have to react to internal or external asynchronous discrete events as consequences of the changes of the environment. The control of these systems must be constructed such that the system fulfills its specifications: reach or avoid some stated, avoid the deadlocks execute cyclically or parallel sequences of events with the shortest periods of time. The discrete events part is modeled using Timed Petri Nets (TPN) and the continuous part using Discrete Time Systems (DTS) and Fuzzy Logic Control (FLC). New methods are presented that are able to link the TPN and FLC models in order to describe the hybrid control system, resulting Enhanced TPN (ETPN) models of systems. The search means to automatically find a controller modeled by ETPN that fulfills some specified requirements. The control synthesis is made employing the techniques of Genetic Programming (GP) which uses the tree representations (LISP expressions) of the ETPNL descriptions of the hybrid system. The parameters of the controller are synthesized using the techniques of Genetic Algorithms (GA). The individuals of the populations taking part to the evolutionary process of GP are organized into species based on some parameters like the isomorphic distance between them.*
Keywords: *flexible manufacturing system; automatic program synthesis; control specification; controller modeling; search methods; genetic programming*

1. INTRODUCTION

The continuously increasing requirements demands for efficiency and the quality led to the development of new production platforms and methods for products manufacturing. One of the highest classes of production systems are the Flexible Manufacturing Systems (FMS). The biggest advantage of the FMSs relative to the other systems is their capacity of reconfiguration, which had a major impact on improving the industrial processes. This led to the introduction of a new term: Reconfigurable Manufacturing Systems. [1]. A FMS is composed of a set of multi-functional machines (and/or robots), an automatic transport system and of control system that controls everything in the FMS (machine status, operations execution). Each machine has a program that can be loaded into it (like a microcontroller) that coordinates, plans the order of the execution of operations. This possible execution of different operations gives the attribute “flexible” to the name of these kinds of systems. A FMS can have between 5 and 20 machines and thousands of operations that involve complex interactions. The transportation of the parts to different machines is performed by an automatic conveyor system. The building of products on these manufacturing platforms is achieved by constructing the component parts of the products and then assembling the parts to compose the whole product. This process of matching components is called product configuration. [3]

Hybrid applications have in composition a discrete events part (DE) and a discrete time (or continuous - DT) part that involve the interaction of different model types. The control synthesis of these systems is a difficult task because they contain models that belong to different approaches. In the current study the Discrete Events System (DES) is modeled using Time Petri Nets (TPN). The Discrete Time System (DTS) is modeled using an enhanced version of TPNs that can link the TPN models with Fuzzy Logic models, and is named Enhanced TPN (ETPN), and thus the entire hybrid system can be described.

The current article presents a method that leads to an efficient production configuration and to a supervisory control of the system that implements some formal product descriptions. The control system of a FMS consists of a set of distributed controlled resources and a central system supervisor player with the coordination role. For manufacturing efficiency some operations are executed concurrently: part processing on different machines, handling (loading/ unloading), palletizing. Other operations are controlled asynchronously due to their access to shared resources [4].

2. RELATED WORKS

Ones of the major problems in the industrial processes were solved by the reconfigurable systems. FMSs are reconfigurable systems. The system's rapid adaptation is very important in planning, scheduling and execution of new production. This led to the appearance of another acronym: Reconfigurable Manufacturing Systems (RMS). The capacity of reconfiguration is achieved by two factors: plant flexibility and control software adaptability. In other words this means modular control software with parameters that can be modified depending of the plant functioning modes.

There are two types of flexibility depending on the duration of the period it is applied. The long term flexibility represents the introduction of new product families into the manufacturing system.

The short term flexibility represents the handling concurrently a large variety of products into the manufacturing system at a given time [4], [5].

2.1. Automatic Program Synthesis

Automatic program synthesis (or construction) represents the generation of executable code and is sometimes associated with program verification techniques. The program synthesis is performed by agents called *inductive inference machines.* These synthesizer programs may use as input different types of data: an abstract description of the problem or an explicit description of the behavior of the desired program.

There are multiple approaches for the automatic program synthesis.

In [1] is proposed a method of synthesis which uses as input data a complete listing of the program that must be synthesized. The synthesis is performed by inference agents.

In [2] is presented a method of program synthesis based on deductive learning. It shows how computer code can be efficiently generated mechanically from declarative specifications.

In [4] is defined the generative programming which has goal to build software systems families, highly customized and optimized products using particular requirements.

In [7] is shown how to use generative techniques to synthesize programs of aspect-oriented programming style.

In [8] is proposed a template-based technique for programs transformation with the aim to improve program performance.

In [9] is presented an approach for synthesis of loop-free programs based on a combination of oracle-guided learning and constrained-based synthesis.

2.2. Enhanced TPN

The DESs can be modeled (described) using Petri Nets (PNs).

There are different types of PNs used to model temporal behavior of DES (Discrete Event Systems):

- Time Petri Nets (TPN): each transition has associated a delay

- Delay TPN (DTPN): the time delays are associated to transitions and to extraction and injection of tokens
- Place TPN (PTPN): the delays are associated to places

In [13] is proposed a method to solve the problem of modeling large scale systems using Petri Nets. Compositional TPN models composed of components and connectors are introduced. After that is defined a set of rules, each of which transforms a TPN component into a very simple one while the net's external observable timing properties are maintained.

In [14] monitors are added to the TPN model of a plant and the linear programming techniques are used to prevent the deadlocks.

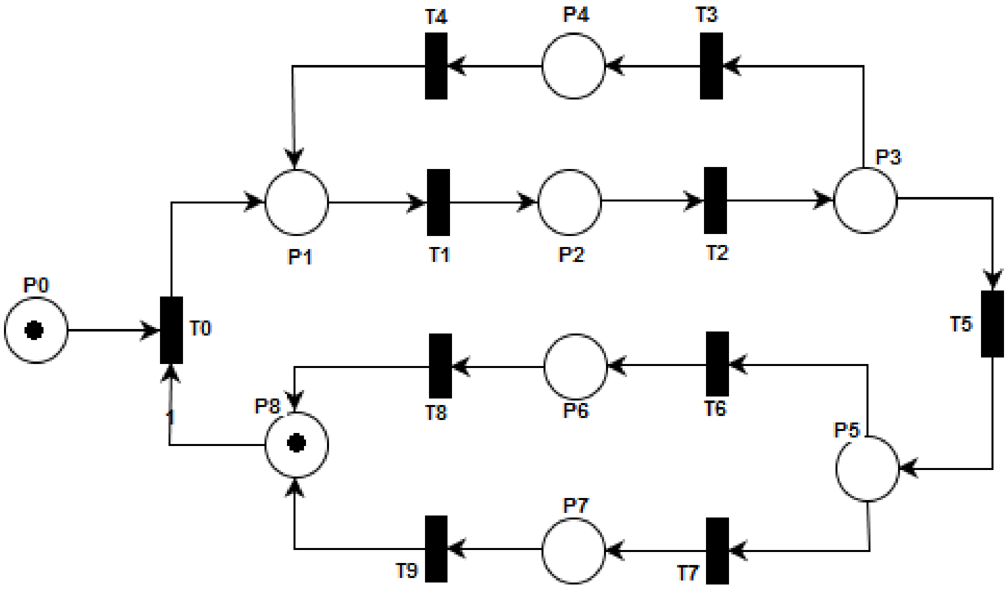

Fig.1: An example TPN

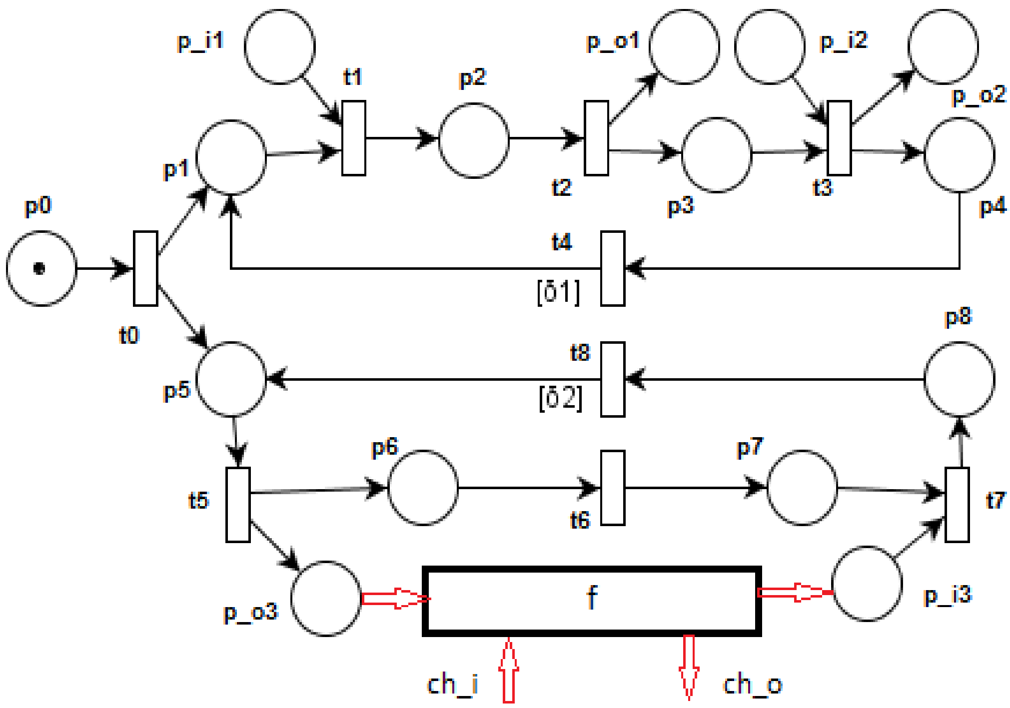

Fig.2: The ETPN controller model of a hybrid system

As above mentioned, the purpose of the system in the current study is a hybrid one (consisting of a DES and a DTS part). The DTS part of a system cannot be modeled using TPNs (as can be seen with DES), which is generally used to model a closed system.

To model a hybrid component that interacts with its environment Enhanced Time Petri Nets are employed (ETPN). An ETPN consists of a TPN enhanced with an interface that contains a set of input places and a set of output places.

Figure 2 presents an ETPN corresponding to the hybrid control of a cyber-physical system. A place from the input set I_{np} can be injected from the outside of the model and the component reacts to this signaled event using a place from the output set O_{np}. A box labeled **f** endowed with an analog input channel and an analog output channel has the role to model the interaction of the controller with external world. The sequence T_1*T_2 shows an example of the program reaction to an input event. The sequence T_5*T_6*T_7 and the function **f** shows how the program reacts to analog inputs. When the transition T_5 fires, the place P_{o3} is reached and then the execution of the function **f** starts. At the end of the execution of the sequence a token is injected in the place P_{i3}. The lower loop is executed with the period $\square_2$.

For the formal description of TPN behavior the TPNL language is used. The TPNL uses the following operators to describe the relations between transitions:

- '*' – sequence
- '#' – loop composition
- '+' – selection
- '&' – concurrent execution

For the example TPN in Fig.1, the following TPNL expression describes it:

$$E_1=T_0*(((T_1*T_2)\ \#\ (T_3*T_4))n\ *T_5*\ ((T_6*T_8)\ +\ (T_7*T_9))). \quad (1)$$

The drawback of TPNL is that it cannot communicate with DTSs. It is generally used to model closed systems.

An extension of the TPNL language to describe the links of a controller with its external environment is the Enhanced TPN language (ETPNL). In ETPNL the operands (transitions) are extended to have 2 arguments (arity 2). They can be of the following form: $t_i[c_i,r_i]$ or $t_i[\square_i,c_i]$. The arguments significances are:

- $\square_i$ – the relative delay of transition t_i
- c_i – the link to an input place (channel) or a time delay
- r_i – the link to an output place (channel)

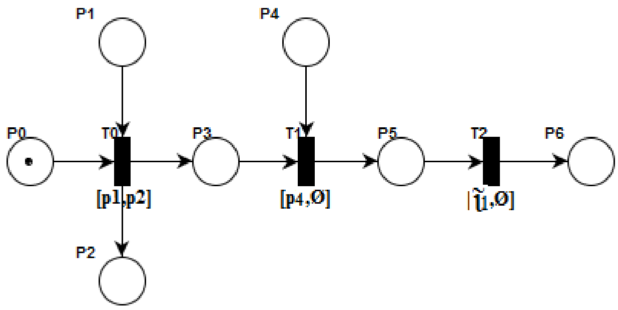

Fig.3: ETPN model of sequential a controller

The ETPN model presented in Fig.3 can be described by the ETPNL expression:

$$E_1= T_0\,[p_1,p_2]*T_1[p_4,ø]*T_2[\ \ _1,ø]. \quad (2)$$

The symbol **Ø** represents the absence of a controlled place. The controller expression can also be written (due to associativity property):

$$E_1= T_0\,[p_1,\ p_2]*(T_1\,[p_4,\ ø]*T_2\,[\ \ _1,\ ø]). \quad (3)$$

This last description is used to transform the ETPN into the LISP expression:

$$E_L = (*\ T_0[p_1,\ p_2],\ (*\ T_1[p_4,\ ø],\ T_2[\ \ _1,\ ø]) \quad (4)$$

The LISP expression is used to construct the tree representation of the controller ETPN model, as it is shown in Fig.4.

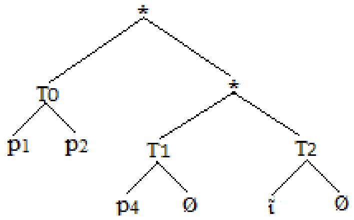

Fig.4: Tree representation of the sequential controller

2.3. Genetic Programming

The search of the controller ETPN model is performed by a GP method. The GP methods use tree representations for the genome.
Expression (1) can be transformed into the Lisp expression:

$E_2=(*\ T_0,*(\#(*T_1,T_2),(*T_3,T_4)),(*T_5,(+(*T_6,T_8),(*T_7,T_9))))).$
(5)

The corresponding tree of the above Lisp expression is shown in Fig.5.

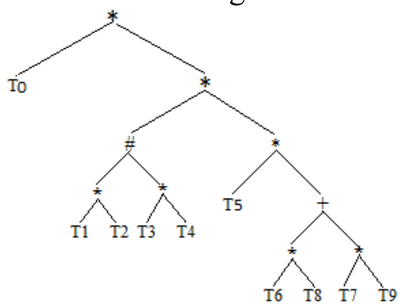

Fig.5: Tree representation of the ETPN model

3 CONTROLLER SYNTHESES

In [15] are approached the problems of product specification and manufacturing system specification. The current research tries to find solutions to the problems of control system of the manufacturing process.

Figure 6 shows the components of the synthesizer and their interaction. The Structural Evolution acts on the ETPN tree of the species and the Adaptation Evolution acts on the parameters Par_i of each individual I_i.

Controller Populations contains in each stage j a set:
$L_x^j = \{I_1^j, I_2^j,\ldots\}$ of individuals that are assigned to each class C_x from the class set $\{C_1^j,C_2^j,\ldots\}$. A class corresponds to a species.

A controller can be described by:
- an ETPN for the structure and the DE behavior
- a set of functions $F=\{f_1,f_2,\ldots\}$ for the DT behavior activated by the ETPN
- a set of parameters $Par=\{par_1,par_2,\ldots\}$ characterizing the ETPN delays or the functions from the set F.

The genome of an individual of the controller is:

$$I_i = (ETPN_i, Par_i). \quad (6)$$

The search of the *Par* set values is implemented by a Genetic Algorithm (GA). The traits are set at the beginning of the synthesis but the evolutionary algorithm can modify them.

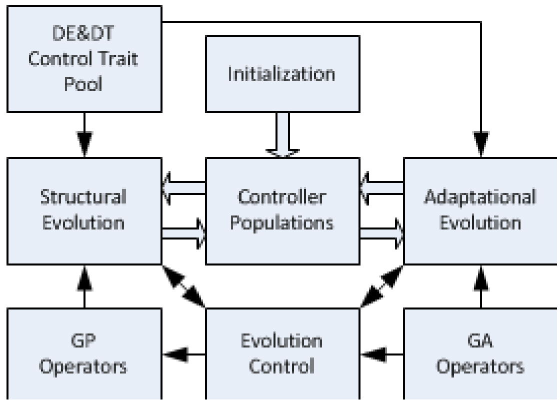

Fig.6: Structure of the control synthesizer

The individuals of the same species have the same structure, that is, in this case, the same ETPN model. They have instead some transitions with different delays or DT functions with different parameters.

In different stages Controller Populations contains different species. When going from one stage to another, some species could disappear and new ones could appear.

The genome of a species C_x has the form:

$$C_x = (ETPN_x, Par_x). \quad (7)$$

Where $ETPN_x$ is the class tree and Par_x is the set of delays and function parameters. The tree $ETPN_x$ describes the set of functions involved for the construction of the individuals of each class.

The Control System receives the specifications of the activities that must be done from the Control Manager and executes the required operation on the corresponding machine.

The model of a FMS, previously discussed and similar to the one described in [2] is shown in Figure 7. This kind of FMS contains two robots (R1, R2), five machines (M1,…M5) controlled by the control system, one deposit (D) and a conveyor (C). The machines and robots have their proper microcontrollers that signal the execution of the required operations.

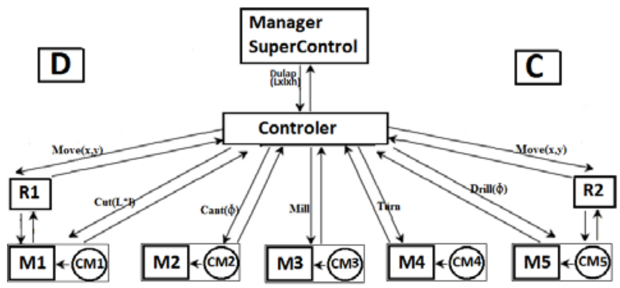

Fig.7: Basic architecture of a FMS

It is the central unit that controls each component of the plant. It controls the movements of the robot in a 3D environment, the activities of the machines (start/stop, operations execution) involved in the manufacturing. It communicates with the Plant by means of communication-reaction channels in which sends control signals to the plant (start/stop, execute operation) and receives the answers from it (execution finished successfully/unsuccessfully, some deadlocks appeared).

The Plant is composed of all the physical components involved in the process of manufacturing of the products (machines, Beyond the activities of controlling the parts of the FMS, the Control System is also involved in controlling the continuous behavior of the manufacturing system (for example the movement of a component on a curve trajectory, such as the knife of cutting machine, the borer of drilling machine.) The Control System receives the coordinates of the targets (points, radiuses, angles of movement) and utilizes Fuzzy Logic Control (FLC) to move the controlled component (knife, borer) as close as possible on the desired trajectory.

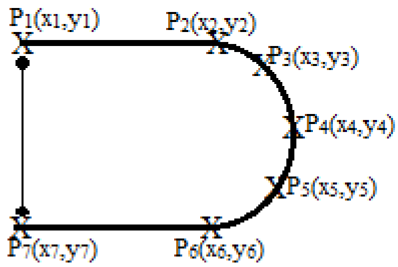

Fig.8: Generation of a curvilinear trajectory

Figure 8 shows the demand to generate a curvilinear trajectory for moving one of the components specified by a set of points.

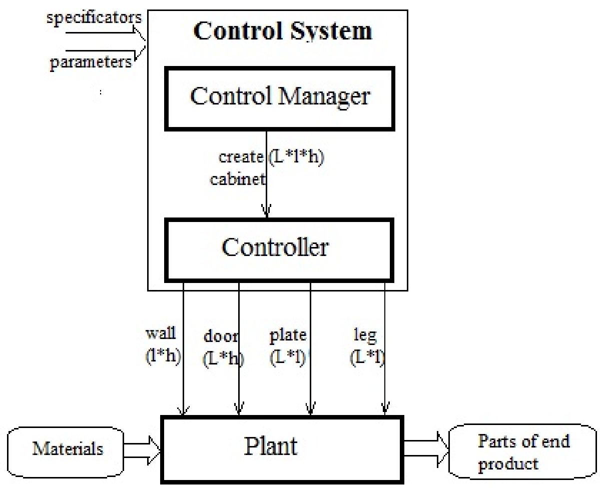

Fig.9: The interaction of the components

The control Manager can be thought of as a graphical interface (software level) created on top of the Controller (hardware level) which transmits it the specifications of the product that must be manufactured (name, parameters etc.).

Figure 9 shows the model of the components interaction for manufacturing a certain product:

- The Control Manager sends the specifications of the product to be processed to the Controller (e. g. name, dimensions, color, material etc.).

- The Controller dispatches this demands into a sequence of control signals that it sends to each component of the plant (robot, machine, conveyor system etc.).

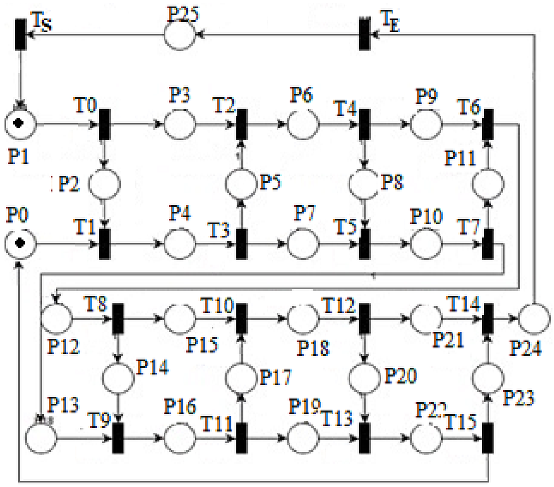

Fig.10: ETPN model of the Controller-Plant interaction

By means of this communication-reaction mechanism, the control system receives back the results of operations executions and their status (success, acknowledgement, failure, certain errors occurred etc.).

The ETPNL expressions that describe the diagram from Fig. 10 are:

$$\sigma_{E1}=(T_0[ø,p_2]*T_2[p_5,ø]*T_4[ø,p_8]*T_6[p_{11},ø]*T_8[ø,p_{14}]*T_{10}[p_{17},ø]*T_{12}[ø,p_{20}]*T_{14}[p_{23},ø])\#(T_E*T_S). \quad (8)$$

$$\sigma_{E2}=T_1[p_2,ø]*T_3[ø,p_5]*T_5[p_8,ø]*T_7[ø,p_{11}]*T_9[p_{14},ø]*T_{11}[ø,p_{17}]*T_{13}[p_{20},ø]*T_{15}[ø,p_{23}]. \quad (9)$$

The corresponding LISP expressions are:

$$\sigma_{L1}=(\#(*T_0[ø,p_2],(*T_2[p_5,ø],(*T_4[ø,p_8],(*T_6[p_{11},ø],(*T_8[ø,p_{14}],(*T_{10}[p_{17},ø],(*T_{12}[ø,p_{20}],T_{14}[p_{23},ø])))))),(*T_E,T_S)). \quad (10)$$

$$\sigma_{L2}=(*T_1[p_2,ø],(*T_3[ø,p_5],(*T_5[p_8,ø],(*T_7[ø,p_{11}],(*T_9[p_{14},ø],(*T_{11}[ø,p_{17}],(*T_{13}[p_{20},ø],T_{15}[ø,p_{23}]))))))). \quad (11)$$

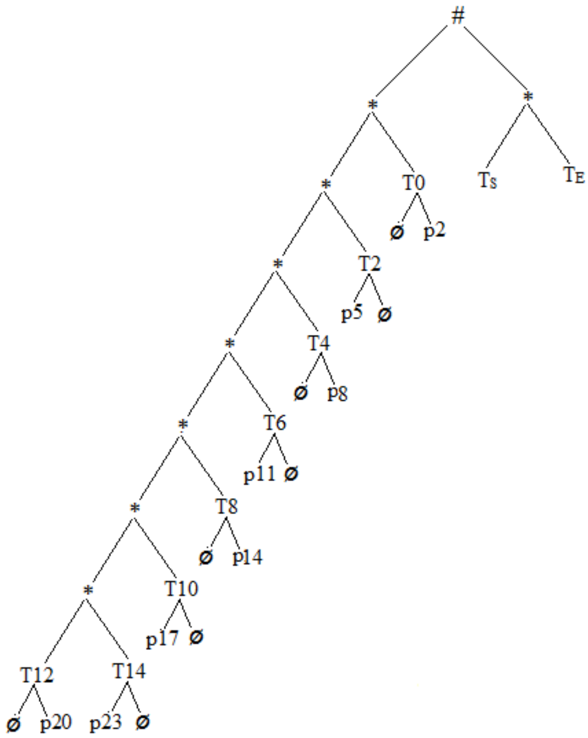

Fig.11a): Tree representation of the controller ETPN model

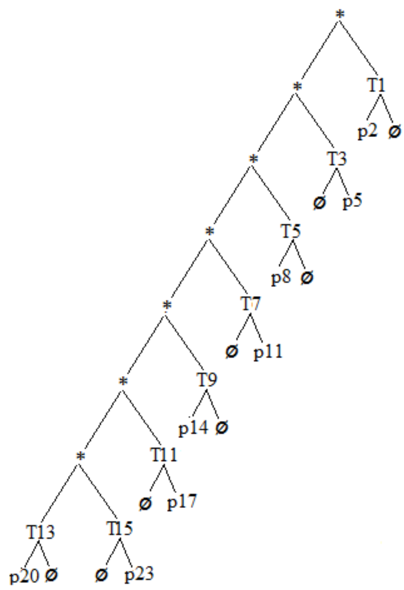

Fig.11b): Tree representation of the plant ETPN model

Each machine performs on the assigned piece of material a series of operations. For example, machine 1 (severing):

- catch the piece of material set by the robot
- position the piece of material in a certain place for a good processing
- mark the material with the dimensions for cutting

- cut the material
- border the severed piece
- release the cut piece of material for further processing on the other machines

The machines that compose the system have the following operations:

- machine 1: cutting the raw piece of material on some given dimensions
- machine 2: insert the cant channel into the piece
- machine 3: mill the piece of material
- machine 4: turn the piece of material
- machine 5: drill the piece of material

All the operations executed by all machines are performed under the supervision of the machine controller. This concept is shown in Figure 12.

The embedded controller takes care of the correct execution of the operations of that machine signaling the operations that have to be executed through the communication channels and receiving the responses from the machine about the results of executions through the reaction channels. This communication is achieved by means of an interface in the same manner as the communication between Controller, Robot and Plant.

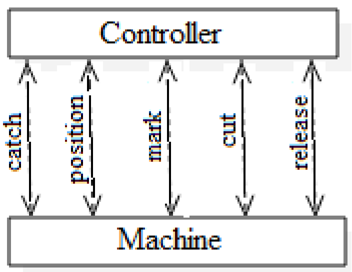

Fig.12: The communication signals between Machine1 and its own controller

Figure 13 shows the ETPN model of the communication between a machine and its own controller. There the interface for communication consisting of the sets of input channels and output channels can be seen.

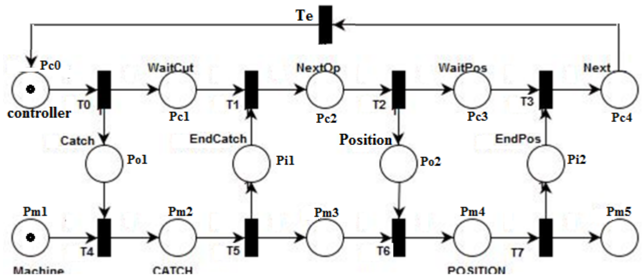

Fig.13: The ETPN model of communication between the machine and its own controller

The communication interface, as can be seen in Fig.13, is constituted from the external places through which signals are sent and received. The transitions that have an external place either incoming or outgoing are called hybrid.

The ETPNL expressions that formally describe the models of machine and controller from Fig.13 are:

$$\sigma_{E1}=(T_0[\text{Ø},Po1]*T_1[Pi1,\text{Ø}]*T_2[\text{Ø},Po2]*T_3[Pi2,\text{Ø}])\#Te. \qquad (12)$$

$$\sigma_{E2}=(T_4[Po1,\text{Ø}]*T_5[\text{Ø},Pi1]*T_6[Po2,\text{Ø}]*T_7[\text{Ø},Pi2]).(13)$$

The corresponding LISP expressions are:

$$\sigma_{L1}=(\#(*(T_0[\text{Ø},Po1],(*T_1[Pi1,\text{Ø}],(*T_2[\text{Ø},Po2],T_3[Pi2,\text{Ø}]))),Te). \qquad (14)$$

$$\sigma_{L2}=(*T_4[Po1,\text{Ø}],(*T_5[\text{Ø},Pi1],(*T_6[Po2,\text{Ø}],T_7[\text{Ø},Pi2]))) \qquad (15)$$

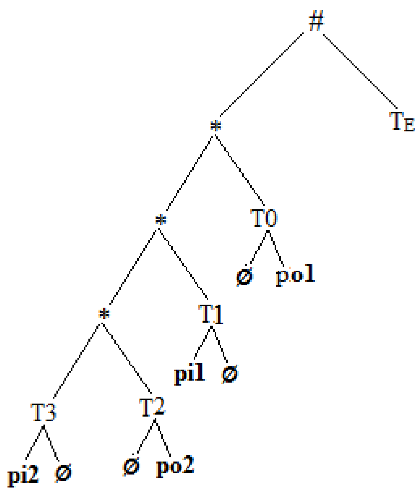

Fig.14a): Tree representation of the controller

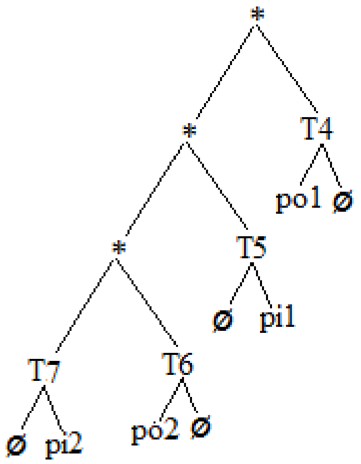

Fig.14b): Tree representation of the machine

The proposed genetic algorithm for the synthesis of the controller uses a tree representation of the controller and an interface matrix as genotype. The genotype contains the controller tree and the interface matrix. The synthesis algorithm takes as input some of the set of characteristics: number of individuals in a generation, number of generations, number of classes in a generation, evaluation functions, adaptation functions etc. The output of the algorithm is the best individual in the form (ETPN, F, Par).

3. CONCLUSIONS

The current approach continues the line of researches about Flexible Manufacturing Systems from the previous works. This approach sustains the FMS optimal design and optimal control synthesis.

The goal of the current study is to demonstrate the possibility to formalize:

- the manufacturing
- the specification
- the FMS capabilities
- the controlling and interaction of components to manufacture the required products

The presented FMS architecture is composed of many modules that have the benefits to design, to construction and to functionality. The formal descriptions of the systems, components and their interactions are obtained by means of modeling tools like:

- UML component diagrams
- TPN diagrams
- TPNL and LISP-S languages

All these open the way to automatically synthesize the control system for simultaneously manufacturing of different products and further to automatic program construction. Avoiding the human intervention to different aspects of system manufacturing removes the appearance of faults and errors and increases the production efficiency in many aspects (time, money etc.). The presented method can be used to verify if the FMS is capable to fulfill the product specifications or to design a FMS for future demands.

For future works the intention is to develop methods based on techniques used in Artificial Intelligence for an improved automatic synthesis of the Control System. The most important technologies for this kind of automation are Genetic Programming (GP) and Genetic Algorithms (GA).

REFERENCES

[1] J. Erschler, G. de Tersac, Flexibilite et role de l'operateur humain dans l'automatisation integree de production Rapport LAAS, no. 88137, 1988.
[2] B.J. Lageweg, J.K. Lenstra, and A.H.G. Rinnooy Kan, Job-shop scheduling by implicit enumeration, Management Science, vol. 24, pp. 441-450, 1977.
[3] E.J. Lee, N. Dangoumau, A. Toguyeni, A Petri net based approach to design controllers for Flexible Manufacturing Systems, Proceeding of 17th IMACS World Congress Scientific Computation, Applied Mathematics and Simulation-IMACS 2005, France, 2005.
[4] Y. Koren, U. Heisel, F. Jovane, T. Moriwoki, G. Pritshow, A.G. Ulosy and H. Van Bruseel, Reconfigurable Manufacturing Systems, Annals of the CIRP, Vol.48, pp. 527-540, 1999.
[5] Y. KOREN, Reconfigurable Manufacturing and Beyond, In the summary of keynote Speech of CIRP05 3rd International Conference on Reconfigurable Manufacturing, Ann Arbor, Michigan, USA, 2005.
[6] M. Silva, R. Vallete, Petri Nets and Flexible Manufacturing, European Workshop on Applications and Theory in Petri Nets, pp.374-417, Spinger, 2009.
[7] A. Voronov, On Formal Methods for Large-Scale Product Configuration, Phd. Thesis, Chalmers University of Technology, Goteborg, Sweden, 2013.
[8] General Motors, Manufacturing Automation Protocol (MAP), a communication network for Open Systems

Interconnection (OSI), Document specification, 1985.
[9] E.Y.T. Juan, J. P. Tsai, T. Murata and Y. Zhou, Reduction methods for real-time systems using delay time Petri nets, IEEE Transactions on Software and Engineering, vol.27, no. 5, pp. 422-448, 2001.
[10] P. Merlin and D. Faber, Recovery of communication protocols implication of a theoretical study, IEEE Trans. Communications, vol. 24, no. 9, pp. 381-404, 1976.
[11] W. Khansa, J. P. Denath and S. Collart. Fault P-time Petri nets for manufacturing systems, Proceedings of WODES96, pp. 19-21, Ediburgh, UK. 1996.
[12] T. Letia, M. Hulea, O. Cuibus, Controller Synthesis Method for Discrete Event Systems IEEE International Conference, Automation Quality and Testing Robotics (AQTR), pp. 85- 90, Cluj-Napoca, Romania, 2012.
[13] J. Wang, Y. Deng, M. Zhu, Compositional Time Petri Nets and Reduction Rules, IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics, Vol.30, No.4
[14] R. Zhu, A Deadlock Prevention Approach for Flexible manufacturing Systems using with Uncontrollable Transitions in their Petri Net Models, Asian Journal of Control, ISSN: 1934-6093, Wiley Library 2011
[15] T. Letia, L. Mic, R. Fat, M. Santa, From Product Specification to Manufacturing Control, IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR), DOI: 10.1109/AQTR.2014

Controlul Sistemelor Flexibile de Fabricatie

Rezumat: Cele mai multe aplicatii sunt de tip hybrid, ceea ce inseamna ca sunt formate din 2 parti: una de evenimente discrete si una discrete de timp(continua).Sinteza controlului pentru acest tip de sisteme este dificila deoarece ele contin interactii intre diferite tipuri de modele care apartin de tehnologii diferite. Componentele controlului trebuie sa reactioneze la evenimente asincrone din interior sau exterior care sunt consecinte ale schimbarilor petrecute in mediu. Controlul acestor sisteme trebuie construit astfel incat sistemul sa indeplineasca specificatiile: atingerea sau evitarea unor stari, evitarea blocajelor, executia ciclica sau in parallel a secventelor de evenimente in cele mai mici perioade de timp. Partea de evenimente discrete este modelata cu ajutorul Retelelor Petri de Timp (TPN) iar partea continua cu Sisteme Discrete de Timp (DTS) si Logica Fuzzy(FLC). In acest studio sunt prezentate noi metode care leaga modelele TPN de cele FLC pentru a descrie sistemul de control hybrid, si acestea sunt modelele Enhanced TPN (ETPN). Cautarea reprezinta metoda de gasire automata a unui controller modelat printr-un ETPN care indeplineste cerintele specificate. Sinteza controlului este realizata cu ajutorul tehnicilor Programare Genetici (GP) care foloseste reprezentarea sub forma de arbori ale expresiilor ETPNL ale sistemului hybrid. Parametrii controlerului sunt sintetizati cu ajutorul tehnicii de Algoritmi Genetici (GA). Indivizii populatiilor care iau parte la procesul evolutionar al GP sunt organizati in specii in functie de niste parametrii precum distanta izomorfica dintre ei.

Liana MIC, drd, Technical University of Cluj-Napoca, Faculty of Automation and Computer Science, liana.mic@yahoo.com
Raluca FAT, drd, Technical University of Cluj-Napoca, Faculty of Automation and Computer Science, ralufat@yahoo.com
Tiberiu LETIA, prof. univ. dr., Technical University of Cluj-Napoca, Faculty of Automation and Computer Science , tiberiu.letia@aut.utcluj.ro