http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 4 (3): 233-238, 2005
ISSN 1812-5638

Modelling and Analysis Flexible Manufacturing Systems

[1]Myriam Noureddine and [2]Patrick Martineau
[1]Department of Computer Science, University of Science and Technology of Oran, BP1505,
El M'naouer, 31000, Oran, Algeria
[2]Laboratory of Computer Science, Polytechnic School of the University of Tours,
64 Avenue J. Portalis, 37200, Tours, France

Abstract: This study presents a new approach to model and analyse flexible manufacturing systems. This work takes place in a widest project of an automatic generator of control software. Our approach is divided into four steps: the first one deals with the physical description of the manufacturing system that is described by a specific notation. The second step builds the conceptual model with series of entities linked together. The following step models each link using controllable outputs Petri nets. A sequence of these Petri nets is obtained and the last step described their integration. An analyser-supervisor controls this integration operation. The results and interpretations given by the analyser provide information on the state of the system, more specifically on identification of potential deadlocks.

Key words: Flexible manufacturing system, modelling and analysis, Petri net

INTRODUCTION

Current production systems have to reply to production demands of small and medium series of different products. This evolution induces often a conflict between the necessity of an important flexibility and a high level of productivity. The notion of flexible workshop or Flexible Manufacturing Systems (FMS) presents a good compromise between flexibility and productivity.

Modelling and analysis tools offer a conceptual framework for the study of FMS. In a broad sense, modelling concerns representations that describe the FMS in concise forms. Analysis concerns techniques that describe the FMS behaviour.

Among the wide spectrum of the FMS methods, Petri Nets (PN) are certainly one of the most important. Using PN tool presents many advantages[1] in the modelling of manufacturing systems. However, the modelling methods are concentrating on just one or several aspects of modelling FMS, -not all aspects. Currently, the methods that focus on the description of the system structure do not include generally the control system, that causes errors and design problems.

According to these remarks and considerations, we propose a complete approach of modelling and analysis of FMS, integrating a control system and using as tool of analysis.

MATERIALS AND METHODS

The proposed approach begins with the FMS description until the analysis of the obtained final model, following four sequential steps, grouped in two modules: Generation of conceptual model and Integration of sub PN.

Generation of conceptual model: The generation of a conceptual model is obtained through two steps: physical description of the FMS and building the conceptual model.

Physical description: This step allows the physical description of the workshop following graphical and textual way. The graphical representation gives exactly the FMS model and the textual representation gives the structure using a detailed formalism. These two representations are obtained by the DeSAP software, developed in the research laboratory of Tours (France).

The textual representation adopted is a generic notation, called notation 5, that describes the FMS structure organized into combination of cells (at least two processing resources served by a single resource of transportation) and lines (at least two processing resources with several resources of transportation). Specific informations are specified for each resource

Corresponding Author: Myriam Noureddine, Department of Computer Science, University of Science and Technology of Oran, BP1505, El M'naouer, 31000, Oran, Algeria

following syntactical rules[2,3]. In our modelling context, we are only concerning by some informations:

- For processing resource: name of the resource (Mi) and its interaction (Input and/or Ouput, or None).
- For transportation resource: name of the resource associated with a sequence of couples (Mi,Mj) modelling transport from Mi to Mj.

Each resource (processing or transportation) is considered like an operational entity associated to a storage entity[2]. Two adjacent entities communicate through the generic model of synchronization following a dialogue insuring the transfer of a piece (or a product) from an entity to the next entity. The generic model of synchronization is based on four primitives[2,4]: Verify Delivery (VD), Deliver (D), Verify Store (VS) and Store (S). To model the process activity of an operative entity, a new primitive called treatment (Trt), must be added.

The conceptual model: The conceptual model is obtained from the generic notation and from the routings (manufacturing processes i.e. sequence of processing resources visited by the product). The basic elements of the conceptual model are the entity and the link. The link allows the representation of the synchronization between two adjacent entities of the conceptual model.

The conceptual model is finally a linear succession of entities connected by links: Ei . Lk . Ej. So, the conceptual model is a new writing of the generic notation, clearer, with only necessary informations for our approach.

The building of the conceptual model is made through two internal functions: the first one translates elements of the notation 5 in an intermediate language and the second one builds effectively the conceptual model from elements obtained by the preceding function.

- The translation of the notation 5, is described by next definitions:
  We define for each processing resource α, a sequence u_α that u_α represents the environment of the resource α following three entities: E_α (input), α, S_α (output) connected by two links.

$$So, u_\alpha = E\_\alpha \,.\, Li \,.\, \alpha \,.\, Li+1 \,.\, S\_\alpha$$

We define for each arc between two processing resources α and β, a sequence $a_{\alpha,\beta}$ that represents the transportation following a resource θ between α and β. The entity θ is connected by two links: a link Lk connects S_α with θ and a link Lk+1 connects θ with E_β . So, $a_{\alpha,\beta}$ = S_α . Lk . θ . Lk+1 . E_β

- The building of conceptual model consist in connecting the elements u_α and $a_{\alpha,\beta}$, that are concerned in a given routing. So, for each routing we build r_i where r_i is a linear representation of the conceptual model and the complete conceptual model is the union of sub models, for all routings.

Formally, the conceptual model SC is SC= $\cup_i r_i$

The partial linear representation obtained for a given routing uses the notion of environment (u_α) of a processing resource α. The complete linear representation is obtained simply by substituting u_α by its value.

The obtained conceptual model shows a clear and linear vision of the FMS, maintaining a high abstraction level, following the natural approach of modular modelling (decomposition of the whole system into subsystems). A finite automaton can represent the conceptual model.

Integration of PN boxes: The section describes the integration of PN boxes following two steps: PN modelling and integration of sub nets.

PN modelling: The PN have been chosen as tool of analysis of the model because PN allow modelling the dynamic behaviour of a system that can be characterized by the qualitative properties of PN corresponding. These properties are liveness, boundedness, reversibility and consistency. The liveness allows to ensure that blocking will never occur, the boundedness guarantees that the number of input processes is upper bounded, the reversibility allows the system to come back to its initial state whatever state it reaches;) the consistency is a necessary condition for the reversibility that is a difficult property to establish.

In the context of a modular modelling, a class of PN is well suited, which is the class of Controllable-Output Petri Nets (COPN) or CO nets[5,6], because sub COPN can be integrated, guarantying the qualitative properties. A COPN is an ordinary PN with input transitions (source transitions, without input places) and output transitions (sink transitions, without output places), where the restricted subnet is an acyclic graph without isolated nodes. A COPN have the above qualitative properties.

The integration operation[5,6] allows obtaining the final model of the manufacturing system according to a process that preserves the properties above. The process consists in the integration of all COPN that are linked together. This is done through a set of interface places and inter-modules arcs and the integrated PN is a COPN with the qualitative properties.

Design of a COPN box: We propose to model the conceptual model by the modelling of links because links

insure the synchronization between adjacent entities of the obtained conceptual model. In the context of modular modelling, each link represents one sub system to integrate.

The link between Ei and Ej is modelled following a COPN[4] with primitives of the generic model: three primitives Trt, VD, D on Ei and three primitives Trt, VS, S on Ej. The four primitives VD, D, VS, S are used to ensure the transfer of a piece from Ei to Ej. The obtained COPN with three input transitions to start the PN functioning and three output transitions to show the end functioning of the PN, has the qualitative properties described previously. We model this COPN by a PN black box[4], concerned only by these input/output transitions and this PN box (Fig. 1) is a generic COPN with:

- te1 = 'Treatment demand on Ei'
- te2 = 'Deliver demand on Ei'
- te3 = 'Treatment demand on Ej'
- ts1 = 'End of Treatment on Ei'
- ts2 = 'End of Store on Ej
- ts3 = 'End of Treatment on Ej'

Integration of sub PN boxes: Each link of the conceptual model is modelled as previously. We obtain thus a sequence of COPN boxes that can therefore be integrated following the principle described previously.

So, we define the next integration rule[4]: each output transition 'End of Store' (ts2) on a box connects to the input transition 'Deliver demand' (te2) of the adjacent box. The realization of the integration rule introduces one interface place and inter-modules arcs and allows to obtain a new PN box, where are concerned only input transitions and output transitions.

Following this specification, two cases may appear, that we identify as the case of linear integration and the case of none-linear integration (introduction of cycles). In the case of linear integration, the obtained PN by the integration operation is a COPN with the qualitative properties. The case of none-linear integration is defined when the integration introduces cycles. In this case, the obtained PN by the integration operation is a PN where there exists at least a cycle that contrary the definition. The obtained PN by the integration operation is not therefore a COPN.

Note that the study of the conceptual model allows identifying the immediate existence of cycles by the presence of more than an occurrence of an entity or a link in the linear representation. The cycles can be identified through the corresponding finite automaton. However, the presence of a cycle does not give always a deadlock but does not allow warning it.

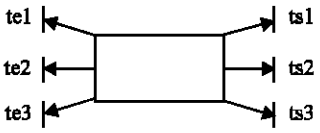

Fig. 1: COPN box for a link (Ei-Ej)

The control layer: We associate a control layer to the integration principle defined previously. The control layer plays the role of an analyser-supervisor and it drives the integration process.

The principle of the control layer is based on an interpretation of the integration formal rule. This interpretation introduces the checking of the availability of entities concerned by the integration.

When an entity is not available, a blocking is identified and the production process is then put on standby until release of the resource concerned. A deadlock intervenes if one or more process awaits a set of resources that it will be able to never obtain. As the functioning of the control layer is interactive, when a first deadlock is identified the integration process stops. In this case, all other potential blockings and/or deadlocks will not be identified and the final PN is not built completely. We add the concept of the mutual exclusion[7] to the principle of the control layer, to build the final PN entirely and to identify all blockings and/or deadlocks.

The control layer can be showing the boxes that induce a deadlock. This information will be exploited for adding instructions in the control software to manage the deadlocks.

So, if a potential blocking is identified during integration, messages will be sent to the control software to identify the resource concerned and to manage this situation (for example, to decide that the process is put on standby). In the same way, if a first potential deadlock is identified, it is signaled and messages will also be sent to the control software (Fig. 2). Integration continues, in order to identify other cases of blockings or deadlocks.

Strategies of supervision: The control layer drives the integration process through two strategy cases: mono production and multi production[7].

The case of the single production strategy considers only one routing with only one product to be manufactured. In this case, there are no deadlocks because all the resources are used in the sequential manner. The case of the multi production strategy considers one routing with several products to be manufactured or several routings with each one at least a product to be manufactured. In this case, one or many deadlocks can appear.

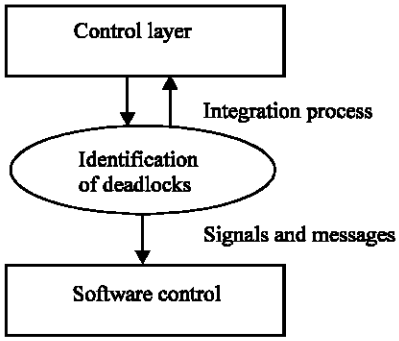

Fig. 2: Supervision of the integration process

EXAMPLE

The next figure presents an example of FMS[5] with four processing resources (machines M1, M2, M3 and M4, grouped in two cells) and three transportation resources (two carriages and an Automated Guided Vehicle-AGV). The first cell is M1 (which is input and output of the system) and M2, with the carriage 1 as a transport resource (C1). The second cell is M3 and M4 (which is input and output of the system), with the carriage 2 as a transport resource (C2). These cells are in relation through an AGV called A, from M3 to M2 and from M2 to M4. There is two routings[5] called G1 and G2 where

- G1 is the routing M1 M2 M4
- G2 is the routing M4 M3 M2 M1

Application of the methodology: The four steps of the methodology are applied.

Physical description: The generic notation (notation 5) provided by the software DeSAP for this FMS is:
"2C, 1L {**M1** TR1 NDIO(:-,1:-,:-),**M2** TR1 EDN(:-,1:-,:-), / **C1** (1) [] (,,) // (**M1,M2**) (-,) ; (**M2,M1**)(-,)} {**M3** TR 1 EDN (:-,1:-,:-), **M4 TR** 1 ED**IO** (:-,1:-,:-), /**C2** (1) [] (,,) // (**M3,M4**) (-,); (**M4,M3**) (-,)} [**M2** TR1 EDN (:-,1:-,:-), **M3** TR 1 EDN (:-,1:-,:-), **M4** TR1 ED**IO** (:-,1:-,:-), / **A**(1) [] (,,) // (**M2,M4**) (-,); (**M3,M2**)(-,)]".
We put in bold informations used in our context.

The conceptual model of the FMS: The application of the translation principle gives therefore the environment of each processing resource and also all sequences representing the transportation between these processing resources.

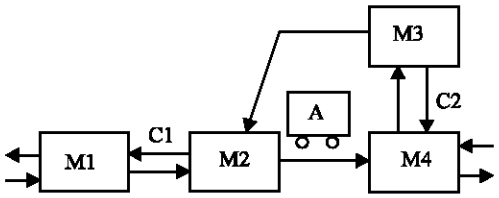

Fig. 3: Example of FMS

For example, the translation gives

- u_{M1} = E_M1 . L1 . M1 . L2 . S_M1
- u_{M2} = E_M2 . L5 . M2 . L6 . S_M2
- $a_{M1,M2} = u_{M1}$. L7 . C1 . L8 . u_{M2}

More, as the input (E) and the output (S) of the FMS is made by M1, the translation gives:

- $a_{E,M1}$ = E . L3 . u_{M1}
- $a_{M1,S} = u_{M1}$. L4 . S

After the translation step, the application of the construction principle gives the conceptual model considering the two routings G1 and G2.

The complete linear representation for the FMS is:

$$SC = r_1 \cup r_2 \text{ with } r_1 \text{ for G1 and } r_2 \text{ for G2.}$$

We can consider only the partial linear representation (environment for each processing resource). For a more easy reading, we give the environment name as the name of the processing resource. This partial representation can be illustrated by a finite automaton *A* (Fig. 4), where states are the entities (resources), with E as the initial state and S the final state. The state transition function allows moving from state to another through links.
For example, r_1 is : L3 . L7 . L8 . L21 . L22 . L16

Note that the obtained automaton allows deducing other potential routings, with their respective linear representations. For example:

- r_3 for G3: M1 M2 M4 M3 M2 M1
- r_4 for G4: M1 M2 M1 M2 M1
- r_5 for G5: M4 M3 M2 M1 M2 M4

The PN boxes: A PN box models each link Li of SC, given by the above finite automaton.

For example, consider the link L7 between the entities M1 (or environment of M1) and C1 (the first carriage).

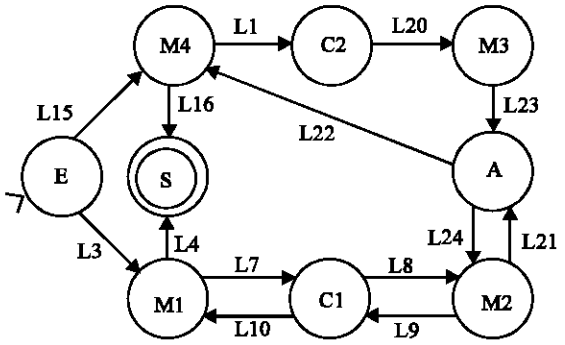

Fig. 4: Finite automaton of the conceptual model

The COPN of this link is obtained from the generic COPN box, described in Fig.1, replacing Ei by M1 and Ej by C1. The transitions of the COPN box of the link L7 is deduced immediately.

So, a sequence of COPN boxes is obtained and these PN boxes are integrated following the formal rule of integration.

Integration and analysis: We give some examples through the two cases of production strategies.

Single production strategy: As seen previously, in this case there are no deadlocks because all the resources are used in the sequential manner. Consider the routings G1 or G2: the integration of PN boxes in their corresponding representations r_1 or r_2 can be done without deadlocks. If there is a path giving a cycle in the automaton (G3 or G5), the integration will be done also without deadlocks. In the same way, for the routing G4 (r_4) that despites a cycle) there is no deadlock and the integration becomes linear (Fig. 5) because there is only one product to manufacture.

Multi production strategy: As seen previously, in this case there are two situations: there are no deadlocks or there are one or many deadlocks.

- No deadlocks. For example, consider the routings G1 (r_1) and G3 (r_3). The control layer supervises the integration process by putting in mutual exclusion the PN box L3. One of these two processes begins and corresponding information will be transmitting to the control software. For example, a put in wait of the other process. So, there are no deadlocks.
- Identification of deadlocks. Consider the routing G4 (r_4), used two times (two products with G4), called P1 and P2. We put in mutual exclusion the PN box L3 in P1 and P2. So, signals and messages will be send to the control software, like 'put in wait P2' if P1 begins.

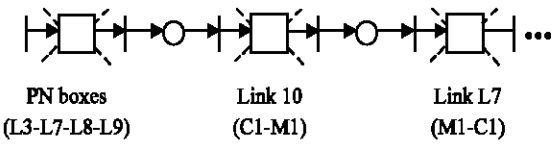

Fig. 5: Integration of PN boxes for G4

P1 continues with transmission of signals to the software control. For example when L3 is passed, the entity (resource) M1 is released and P2 can begin.

Note than sometimes, the mutual exclusion principle is not sufficient because there is a cycle. For example, a potential deadlock will be identified induced by the COPN box L7 that appears four times (two in the first process and two in the second process). The control layer will send these informations to the control software that will suggest the instructions to be added to resolve this deadlock.

CONCLUSIONS

This study is situated in the methodological framework of FMS modelling. The adopted approach covers the steps of representation, modelling and analysis of FMS and each step is described with a formalism.

Our approach uses a modular modelling, that allows integrating the different models. These models are represented in the form of Controllable-Output Petri nets. The obtained models are analyzed and we define a control layer, playing the role of supervisor that drives the integration operation of obtained sub.

Our method is articulated around two aspects: a formalism aspect, the conceptual model and a control aspect of the integration. The analysis aspect is fundamental because it allows guaranteeing the validity of the proposed approach. Indeed, we have identified cases where problems are underline (cycles and deadlocks) during the integration step; these cases cannot be suppressed in the model because they are induced by the even organization of the considered workshop. The analysis of the final PN gives elements of solution to these problems and some cycles can be interpreted and some unresolved deadlocks could be localized.

REFERENCES

1. Moore, K.E. and S.M. Gupta, 1996. Petri nets models of flexible and automated manufacturing systems: A survey. Intl. J. Production Res., 34: 3001-3035.

2. Martineau, P., C. Tacquard and A. Rouillon, 1999. Génération automatique de la conduite d'un système flexible de production. J. Européen des Systèmes Automatisés, 33: 815-853.
3. Tacquard, C. and P. Martineau, 2001. Automatic notation of the physical structure of a flexible manufacturing system. Intl. J. Production Econom., 74: 279-292.
4. Noureddine, M. and P. Martineau, 2000. Towards a modeling methodology of manufacturing systems. Conference Management and Control of Production and Logistics MCPL'2000 (IFIP/IFAC/IEEE), Grenoble (France), CD-Rom, session P2-P34, pp: 6.
5. Proth, J.M. and X. Xie, 1995. Les réseaux de Petri pour la conception et la gestion des systèmes de production. Masson Editeur, Paris (France), pp: 293.
6. Proth, J.M., L. Wang and X. Xie, 1997. A class of Petri Nets for manufacturing system integration. IEEE Transactions on Robotics and Automation, 13: 317-326.
7. Noureddine, M. and P. Martineau, 2003. Approches supervisées pour la détection des interblocages dans les systèmes de production. Conférence Internationale sur la Productique CIP'2003, Algiers (Algeria), CD-Rom, session B6-Art.4, pp: 7.