SUPERVISION AND CONTROL OF HETERARCHICAL DISCRETE EVENT SYSTEMS: THE LAAS APPROACH

MARCOS R. DA SILVEIRA [1] [2] *, MICHEL COMBACAU [1]

[1] *LAAS-CNRS - 7 Avenue du colonel Roche, 31077 TOULOUSE - cedex 04 France*
Université Paul Sabatier - 118 Route de Narbonne, 31044 TOULOUSE - cedex 04 France
[2] *Pontifícia Universidade Católica do Paraná PUCPR*
Laboratory of Automation and Systems - R. Imaculada Conceição, 1155 - 80215-901 CURITIBA – Brazil
* *Financial Support also given by CNPq (Centro Nacional de desenvolvimento Científico e Tecnológico)*
E-mail: dasilveira@laas.fr, combacau@ laas.fr

Resumo: Este artigo apresenta o estado atual das pesquisas desenvolvidas sobre supervisão e controle de sistemas discretos. As definições de alguns termos usados por pesquisadores franceses na área de sistemas de produção são introduzidas. Uma suscinta descrição sobre os trabalhos efetuados anteriormente no LAAS é apresentada, a qual descreve soluções para a supervisão e o tratamento de falhas em sistemas hierárquicos. A continuidade destes trabalhos nos leva a analisar as características de supervisão de um sistema distribuído. Assim, neste artigo é proposto uma técnica formal para distribuir um modelo centralizado (baseado em redes de Petri) a fim de obter um conjunto de submodelos os quais apresentam uma redundância parcial de seus dados.

Abstract: This paper presents the current state of researches about supervision and control of discrete event systems. The definitions of some terms used by French researchers of the production systems community are introduced. A brief description of previous works developed at LAAS is presented in order to depict the solutions for process failure supervision in a hierarchical architecture. Presently, our researches focus on supervision based on distributed architectures. In this paper, we propose a formal technique to split up a centralized Petri net (the model of the process) into a set of sub models with partial redundancy.

Keywords: Supervision, monitoring, distributed models, discrete event systems, architectures.

1 Introduction

Modern industries use the automatic systems to improve the quality of its products. To achieve the market goals, the production systems should increase the adaptation capacity and reduce the production costs. It is a complex problem involving the production control and management. Many industries implement a distributed workshop and distributed computation to solve these problems, facilitating the reconfiguration of its production line. The continuous researches for the bests solutions motivate the collaboration between scientists and managers to develop optimized architectures to control and supervise the production line by organizing the cooperation between devices that are geographically distributed. Some difficulties are identified in this case: The weak link between devices (that can be a dynamic link or partially specified in the models) result a complex problem for accessing, controlling and for even having visibility of data, events and general communication objects (Fuertes at al, 1999; Forrester, 1998).

Modeling a production line according to the industrial priorities needs a wide knowledge about the production process. Different modeling techniques exist to aid the modeling of the system, among them Petri net and automata theory are the most used to describe discrete events. Petri nets were first introduced by Peterson to model and analyze the manufacturing systems (Peterson, 1981). Other works enriched it, as Ramadge and Wonham (Ramadge, 1987) that had presented a proposition of modeling and analysis of DES (discrete events systems) with controllable and uncontrollable events that can be turned on and off by the supervisor. Lin and Wonham (Lin, 1990) introduced a coordinator to local supervisors in order to improve the closed-loop behavior under decentralized supervision. The automata theory is referenced by Smith (Smith, 1993), who related control commands generation to state transitions. According to Du (Du, 1999), the two major gaps between the existing models and the supervisory control practice are the simplistic definition of system states and the information contained in each message exchange. The information propagation in complex systems is referenced in Zamai (Zamai at al, 1997) who proposed split the whole control process into simpler subparts of a hierarchical structure. Each subpart is composed by a process model and by a generic control, monitoring supervision functions, made to react in normal or abnormal situations. A communication mechanism allows levels to communicate by taking control and monitoring requirements into account. Duffie (Duffie, 1996) describes the attractiveness of, what he named, the heterarchical architecture as alternative to conventional hierarchical control architecture. In the approach of Cho (Cho at al, 1999), he had suggested mixing centralized and decentralized supervisory control when pure decentralized control is not appli-

cable, and he had presented the consequences in view of computational complexity.

We can notice that a great number of researchers propose advanced methods and tools to analyze the distributed models, but these models are, in many cases, created by empirical methods based on the knowledge of experts and customized for a selected application.

In a sense, our goal is present a generic idea of previous works developed at LAAS about the supervision of discrete event systems and the current stage of this researches involving the distributed supervision and control systems.

The second section introduces the definitions of some terminologies. The motivation to use distributed systems is presented in the section three and the previous works developed at LAAS is described at section four. The fifth section details the systematic of distribution of a centralized model and the mathematical formalism is presented in the sixth section. Finally, the communication and the data update problems are introduced in the section seven.

2 The used terminology

This section presents some definitions taken from (Combacau et al., 2000), a collaborative paper of French researchers of the production systems community. Control, monitoring and supervision are first defined and then the definition of terms used in this paper is given.

Control: It triggers the execution of a set of operations by giving orders to the process actuators. It may be:

- A set of operations corresponding to the manufacturing sequence of the product.
- A set of operations executed to restore the process functionality offered during normal execution.
- Actions with a high priority level engaged in order to protect the shop workers and to prevent catastrophic evolutions.
- Some checking, tuning or cleaning operations executed in order to maintain the process in its operational state.

It means that our definition of control includes all the functions actually acting on the process.

Monitoring: It collects data from the process and from the controller, it determines the actual state of the controlled system and it makes the inferences needed to produce additional data (historic, diagnosis, etc.). Monitoring is limited to data processing and has no direct actions on the models or on the process.

Supervision: It computes and sets the parameters of the control sequence to be executed according to the state of the control system and to the state of the process. It includes normal and abnormal operations.

During normal operation, the supervision takes the decisions to raise the indecision in the control system (real-time scheduling, optimisation, control sets and switching from a control law to another one).

When a process failure occurs, supervision takes all the decisions necessary to allow the system to resume its normal operation (rescheduling, recovery actions, emergency procedures, etc.).

It must be noticed that supervision takes place in a hierarchical structure (with at least two levels). At the lowest level of the structure only the control and monitoring functions are generally implemented - no real decisions have to be taken.

Some generic terms need to be defined. Some of them can be found in (Laprie, 1992).

Fault: Action, voluntary or not, that does not take all the specifications into account.

Defect: Difference between the actual value of a parameter and its nominal value.

Error: Part of a model not exactly matching the specifications of the physical system. Logically, an error is a consequence of a fault.

Latent error: The error is qualified as latent as long as the erroneous part of the model has not been used. After using the erroneous part of the model, the error becomes effective.

Failure: Event characterizing a situation in which an operation is not executed by a resource because its state does not correspond to the nominal specifications any longer.

Breakdown state: State of a resource from which the system cannot provide the specified service. This state is a consequence of a failure.

Symptom: Event or data through which the detection system identifies an abnormal operation of the process. The symptom is the only information the monitoring system knows at the detection step.

Recovery point: State reachable from the breakdown state in which the system must be driven to resume the normal operation.

Recovery sequence: Set of ordered actions executed to bring the process back from the breakdown state to the recovery point.

According to these basic concepts, we can define the elementary functions of the supervision and monitoring system. Between brackets the letter *M*, *S* or *C* indicates to which previously discussed group (Monitoring, Supervision, Control) the function belongs.

Detection (M): determines the normality or abnormality of the functioning system. Two classes of abnormal operations are considered :

- The first one includes situations in which basic operating constraints of the process are violated (collisions for instance).
- The second one groups together situations in which the part routing (control law) is not respected (fabrication delays for instance).

Follow (M): maintains a history of treatments executed and a trace of events observed by the control/supervision system.

Diagnosis (M): looks for a causality link between the observed symptom, the failure and its origin. Classically, three sub-functions are distinguished:

- Localization determines the subsystem responsible for the failure,
- Identification identifies the causes of the failure,
- Explanation justifies the conclusions.
- *Prognosis* (M): foresees the consequences of a failure on the future operation of the system. The consequences can be immediate ones (resource unavailable) or induced ones (faulty parts in the workshop).

Decision (S): determines the state that must be reached to resume to the normal operation, then determines the sequence of corrective actions to be performed to reach this state.

Recovery (C, S): acts on the process by changing the states of the resource or equipment and on the control system by changing the control laws, the part routing, etc. Three classes can be defined:

- Minor, only the control laws are adapted,
- Significant, other resources are reallocated,
- Major, reallocated resources need to be prepared to execute the recovery.

3 Distributed Systems

Rapid technological advances in computing and communications have made it possible to consider a wide range of possibilities in the design of control architectures. We can observe many basics architectures proposition, but centralized, hierarchical and "heterarchical"(as well as hybrids, holonic, dynamics, etc.) are the most accepted (Dilts, 1991; Brennan, 2000; Forrester, 1998). In this paper, we will use the terminology *heterarchical* to indicate an architecture that doesn't have the same characteristics of centralized or hierarchical ones. The heterarchical architecture is adopted in order to pursuit a full local autonomy, which the global information is minimized or eliminated. This implies that: (1) there are no external higher levels of control to coordinate any cooperative process, (2) the communication between entities will not have a master/slave relationship as found in the others architectures, (3) we can introduce a new entity or modify the existent ones without significant structural changes.

However, the complexity associated to the process distribution as well as the coherence between entity models and the faulty treatment propagation no have satisfactory solution yet, and constitute a large domain of research.

The hierarchical architecture solves the coherence problem introducing different levels of control/supervision. At LAAS, the hierarchical architecture was firstly analyzed architecture and important results were obtained. The most part of these researches is also valid to the distributed cases and are presented briefly in the next section.

4 The LAAS approach

Previous works have defined a hierarchical and modular approach of monitoring and supervision (Combacau, 1991; Chaillet, 1995). It must be noticed that only process failures are taken into account. The control and monitoring system is considered as error free. When a process failure occurs, the corrective actions to be executed are performed according to the process resources used, to the kind of manufactured products and to the production strategy specified by the user.

The failure processing is not limited to the classical sequence (detection, diagnosis, decision and recovery) (Zamaï et al., 1998).

The Acquisition/Routing block manages all these functions. This block is based on an algorithm directing incoming messages to the most suitable functions according to the nature of the data and to the state of the monitoring system. Moreover, this algorithm maintains the state of this model and triggers the suitable monitoring, control and/or supervision functions (Combacau et al., 1998).

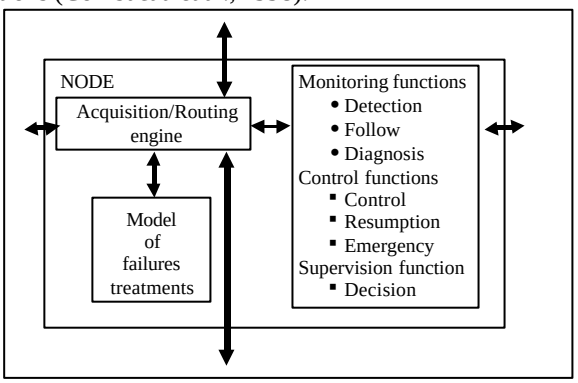

Figure 1: A generic module for control, supervision and monitoring

From a specification point of view, this approach is based on complementary tools. Petri nets with Objects are used to model control, recovery and emergency sequences and failures treatments. An extended entity relationship model provides a process representation (called Information System) in which data that are not easy to model by means of Petri nets (time notions like dates, duration, history to keep a track of data evolution, etc.) can be found.

Most of monitoring and supervision functions have already been described by other publications (Berruet et al., 1998; Dangoumau et al., 1999; Combacau et al, 2000). Our recent works propose a systematic procedure for distributing a centralized model of supervision and control. It is based on the Petri nets (PN) and it shows as advantage that each part of control process has, at least, the same proprieties of the whole model until all the good proprieties (boundedness, liveness, home state). This procedure is founded on the linear PN invariants theory and it results a set of control parts associated to a resource (or a set of them) and its interactions with others resources. Some parts have redundant information

about the systems, because the relation between resources must be represented in all entities that use its services. The sequel of the paper is dedicated to these recent results.

5 The systematic distribution

This section describes the methodology used to distribute a centralized model. The whole model is split in several subparts (sub-models), which represent all possibilities for one resource moves from one state to another and represent the relations with others resources. Normally a sub-model is associated to only one resource, nevertheless there are no restriction to group several resources into one sub-model. The main constraint is that all resources should have its activities represented in at least one process sub-model. In this work, the description of the production system is separate in two classes: the control and the process model.

The *control flow model* (or control model) represents all constraints associated to the transformation of raw parts to finished goods, called the operational constraints.

The *process model* represents the physical and functional characteristics of the process (mutual exclusions, sequence constraints, co-operation activity, etc.).

According to these informations, an entity composed by a control flow sub-model, a process sub-model and all functions of supervision and control constitute a *module*.

A module must have all information necessary to execute any modeled activities to express the industrial reality. Then the branch of the whole PN model is duplicated in several modules if it describes the relation between them. In other words, there is a *partial redundancy* of global information when common activities are described in different modules. The scope of this redundancy can be enlarged to incorporate *important information* in order to increase the reliability of the system. In this paper, all sequences of activities that can drive the system to a catastrophic situation is considered as important information. The duplication of information follows strict mathematical rules as described on the next section.

6 The mathematical formalism to distribute models with partial redundancy.

This section presents some basics concepts of Petri net to aid the comprehension of the model distribution methodology. The linear invariants are classed as place invariant and transition invariant. We are interest on a positive version of linear invariants because it represents a realistic evolution of the system.

Definition: Let a finite PN system with $P = p_1, p_2, ..., p_m$ and $T = t_1, t_2, ..., t_n$ be given.

A vector $v \in Z^m$ is called a positive P-invariant of a given PN, iff:

$v^T \bullet C = 0 \wedge v \neq 0 \wedge v_i \geq 0 \ \forall i = 1, 2, ..., m$

Where *C* is the incidence matrix.

A vector $s \in Z^n$ is called a positive T-invariant of the given PN, iff:

$C \bullet s = 0 \wedge s \neq 0 \wedge s_i \geq 0 \ \forall i = 1, 2, ..., n$

The systematic distribution proposed is founded on the positive p-invariant (pp-invariant) theory and generators, referenced in (Kruckenberg, 1987). In this paper, we will focus only on the third level of generators, where each positive invariant is a positive linear combination of generators, which are minimal standardized invariants.

The places that compose a generator and the input/output transitions associated to them establish the minimal set of places/transitions of a distributed sub-model. Others components can be included to give more autonomy and/or reliability to the module. They are classified in two different classes: (1) the ones directly (by one arc) connected to the transitions of the minimal set, and (2) the resulting of a reduced PN.

This systematic distribution is ordered in four steps:

First one, the places that compose a generator are identified.

Definition: A set of places of one generator v_k.

$$\begin{cases} v_k \left| v_k^T \bullet C = 0 \right. \\ Q_k = \{ p \in P \left| v_k(p) > 0 \right. \} \end{cases}$$

$$Q = \bigcup_{i=1}^{N} Q_i \ , \ N = \text{number of generators}$$

Second one, the input and/or output transitions of these places are grouped as a set of local transitions.

Definition: A set of input/output transitions of one generator

$$S_k = \left\{ t_j \in T \left| I(t_j, p) + O(t_j, p) \neq 0 \ \forall p \in Q_k \right. \right\}$$

$$S = \bigcup_{i=1}^{N} S_i \ , \ N = \text{number of generators}$$

Third one, all input and/or output places of these transitions, except the ones identified on the first step, are grouped as a set of external places.

Definition: The external places.

$$j_k = \left\{ p \in P \setminus Q_k \left| \exists t_j \in S_k \quad I(t_j, p) + O(t_j, p) \neq 0 \right. \right\}$$

$$j = \bigcup_{i=1}^{N} j_i \ , N = \text{number of generators}$$

Finally, all others places and transitions not considered by the previous steps are reduced to compose the *reduced PN* (PN^-). The PN^- is coupled with the sub-model as complementary information.

Some constraints can be imposed by the user to obtain the PN^-. For instance, critical parts of the sys-

tem (the important information) are not reduced to increase the global fault tolerance.

Others tools are used to optimize the model distribution, as well as the predicate/transition PN. In this case, all activities executed by a specific resource can be recognized. If these activities are specified in different generators, then they are coupled to describe the comportment of the resource. In other sense, the subnets that don't use the referenced resource can be reduced. The steps 3 and 4 of systematic distribution are applied on the coupled generator as one.

The

Figure 2 shows a single example of the distribution methodology. A centralized model describes the activities of two resources R_A and R_B. The places associate to positive p-invariant of the specified model ($Q_A = \{R_A, P_1, P_2, P_3\}$ and $Q_B = \{R_B, P_4, P_5\}$) are the minimal set of place of each sub-model, two sub-models are identified SM_A and SM_B respectively. The transitions associate to each set of places composes the minimal set of transitions ($S_A = \{t_1, t_2, t_3, t_4\}$ and $S_B = \{t_5, t_6, t_7\}$). The interaction between models is represented by the inclusion of places direct connected to the minimal set of transitions ($\varphi_A = \varphi_B = \{stck\}$) and by the inclusion of PN^-. $SM_A = (Q_A \cup \varphi_A) \times S_A \cup PN^-$ and $SM_B = (Q_B \cup \varphi_B) \times S_B \cup PN^-$

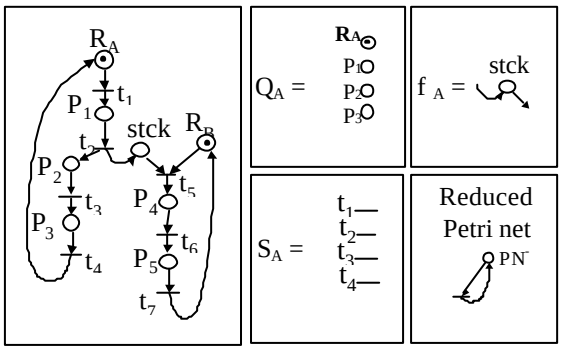

Figure 2: Centralized model

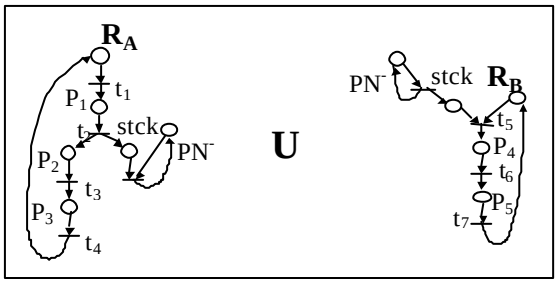

Figure 3: Distributed sub-models

The consequence of the model distribution is the increasing of complexity to treat data when there are synchronizations of distributed actions or competitiveness by one resource. Physical constraints impose some delays to update distributed information and cases of multiple reservation of one resource can appear. A communication protocol is proposed to maintain the data coherence.

7 The communication procedure

In the most part of manufacturing systems, the nature of the subsystems is asynchronous and their behaviors are independent of other subsystems. Thus, during execution of an activity, each subsystem assume its own unique state (reflected by the data value of the variable, etc.) and none has knowledge about the exact state of other subsystems. When activities that depend on several resources are required, then the modules that control these resource (as described in the previous section) have to communicate, in order to update their data and to synchronize its actions (intermodules communication). Any incoherence of data can result in a decisional or operational conflict and can drive the system to an erroneous state. Some authors propose different negotiation techniques between the modules that should cooperate, but the negotiation time is generally incompatible with the real time processes. We propose to centralize a part of the decisions to optimize this time. This alternative is applied when competitiveness and cooperation are identified. In this particular case, the decision is taken by only one module called centralized decider module (*CDM*), as shown in

Figure 4.

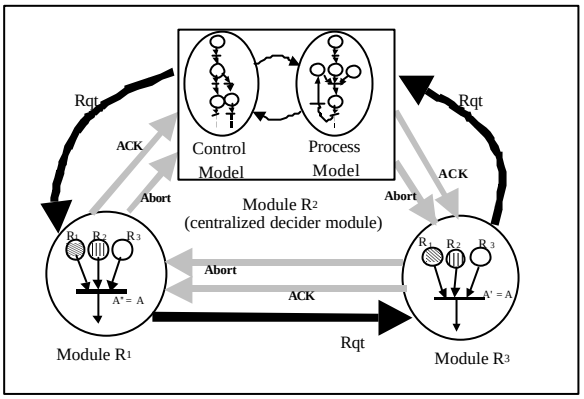

Figure 4: The verification sequence

Some specific roles such as physical characteristics, industrial priorities, security, etc. are adopted to choose the CDM. The CDM has different finality according to the model applied. The process model must solve physical conflicts coming from the hardware flexibility (for example, it chooses the resource that will execute an activity among a pool of resources), while the control model must solve conflicts between activities using decisional flexibility (for example, needing a shared resource, two activities are waiting for the end of a production process). When a conflict cannot be solved using the process model, then it must be solved according to the control model. If no solution is found a symptom is detected and the supervision functions are launched. It looks obvious that a communication between both models (intramodule communication) is necessary and that this kind of communication doesn't work as intermodules communication. In this paper, both (intramodule and

intermodules) communications are not detailed, we consider that the intramodule communication is a variety of client/server protocol where the client and the server are not fixed, they are dynamically chosen according to the need of the system. The intermodule communication is used to verify the pre-condition of a required activity.

8 Conclusions

A formal technique based on the Petri net Theory has bee presented in order to distribute a centralized model into a heterachical architecture. This systematic is based on some properties of Petri nets as the linear invariant theory. The distribution of a whole model results in a set of sub-models that describes the behavior of a resource (or a set of them) and the interactions with other resources. The distributed model has the same properties than the centralized one. These models are used by the control and supervision functions. The modules have autonomy to execute the activities described by its models since every pre-conditions are satisfied. The synchronization of actions executed by different modules is guaranteed by the intramodule protocol that supports the communication between them and maintains the coherence of distributed data. The perspective of these works concerns the quantification of the redundancy inserted by the distribution methodology.

References

Berruet P., Toguyeni A., Elkhattabi S., Craye E. (1999). Tolerance evaluation of flexible manufacturing architectures. *Journal of Intelligent Manufacturing*, Vol. 10, N° 6, December, p. 1-14.

Brennan, R. W. (2000). Performance comparison and analysis of reactive and planinning-based control architecture for manufacturing. *Robotics and Computer Integrated Manufacturing*, No. 16, pages 191 - 200.

Chaillet A. (1995). Multi-model approach for real-time control and monitoring of complex discrete events systems. *PhD thesis, University of Toulouse III*, December.

Cho, K., Lim, J (1999). Mixed centralized/decentralized supervisory control of discrete events dynamic systems. *Automatica*, No. 35, pages 121 - 128. Ed. Elsevier Science.

Combacau M., Berruet P., Charbonnaud, Khatab A. and Zamai E. (2000). Supervision and monitoring of production systems. *Proceedings of MCPL'2000*, Grenoble, 4-6 juillet.

Combacau M., Zamaï E., and Chaillet-Subias A. (1998), Monitoring Strategies as Control Structure of Monitoring Architectures Based on Discrete Event Systems. *Computational Engineering in System Applications*, Nabeul-Hammamet, Tunisia, 1-4 April.

Dangoumau, N., Elkhattabi, S. Craye, (1999). Design and Management of Flexible Manufacturing System's modes. *ACS'99*, Szczecin, Poland, 18-19 November, p. 417-422.

Dilts, D. M., Boyd, N. P., Whorms, H. H. (1991). The evolution of Control Architectures for automated Manufacturing Systems. *Journal of manufacturing systems*, vol 10, pages 79 - 93.

Du Hua, X., Zhou, C. (1999). Message-oriented decomposition for supervisory control in manufacturing systems. *Robotics and Computer Integrated manufacturing*, No. 15, pages 441 - 452. Ed. Elsevier Science.

Forrester, J. W. (1998). Designing the future. *Work presented at Universidad de Sevilla*, 15 dec, Spain.

Fuertes, J.M., Herrera, J., Arboleda, J.P., Heit, F., Casas, C., Company, J. (1999). Communication system for a distributed intelligent controller. *Microprocessors and microsystems*, No. 23, pages 89 - 93, Ed. Elseviers Science.

Ghosh S. (2001). Understanding complex, real-world systems through asynchronous, distributed decision-making algorithms. *The Journal of Systems and Software*, No. 58, pages 153-167, Ed. Elsevier.

Huang, Y., Jeng, M., Chung, S. (2001). Design, analysis and implementation of a real-world manufacturing cell controller based on Petri nets. *International Journal of Computer Integrated Manufacturing*, vol. 14, N° 3, pag. 304-318, Ed. Taylor & Francis, 2001

Kruckenberg, F., Jaxy, M. (1987). Mathematical Methods for Calculating Invariants in Petri Nets. *Advances in Petri Nets*, Springer, LNCS 266, ed. G. Rozenberg, pages 104-131.

Lin, F., Wonham, W. M. (1990). Decentralized supervisory control of discrete event systems with partial observation. *IEEE transaction in Automatic Control*, No. 35, pages 1330 - 1337.

Peterson, J. (1981). Petri net theory and the modeling of systems. Ed. Prentice-hall International, Englewood Cliffs, NJ.

Ramadge, P. J. G., Wonham, W. M. (1987). Supervisory control of a class of discrete event processes. *SIAM Journal Control Optimisation*, No. 25, pages 206 - 230

Smith, J. S., Joshi, S. B. (1993). Message-based part state graphs: a formal model for shop floor control. *IMSE Working paper series*, Pennsylvania State University.

Zamaï, E, Combacau, M., Chaillet-Subias, A. (1998). Models and Strategies for Monitoring of Flexible Manufacturing Systems. *9th Symposium on Information Control in Manufacturing*, Nancy-Metz, France, June 1998.

Zamaï, E., Subias, A. C., Combacau, M., de Bonneval, A. (1997). A hierarchical structure for control of discrete events systems and monitoring of process failures. *Studies in Informatics and Control*, vol 6, No. 1, March.