5

Fault Diagnosis in Discrete Event Systems using Interpreted Petri Nets

Jesús Arámburo-Lizárraga, Antonio Ramírez-Treviño,
Ernesto López-Mellado and Elvia Ruiz-Beltrán
CINVESTAV Unidad Guadalajara
México

1. Introduction

Diagnosability property and fault detection schemes have been widely addressed on centralized approaches using the global model of the *Discrete Event System* (*DES*). Roughly speaking, diagnosability is the property of determining if using the system model is possible to detect and locate the faulty states in a finite number of steps. In the works (Sampath, et al., 1995) and (Sampath, et al., 1996), a method for modeling a *DES* using finite automata is proposed; based on this model, a diagnoser is derived. The cycles in the diagnoser are used to determine when the *DES* is diagnosable.

Recently, fault diagnosis of *DES* has been addressed through a distributed approach allowing breaking down the complexity when dealing with large and complex systems (Benveniste, et al., 2003; O. Contant, et al., 2004; Debouk, et al., 2000; Genc & Lafortune, 2003; Jiroveanu & Boel, 2003; Pencolé, 2004; Arámburo-Lizárraga, et al., 2005).

In (Debouk, et al., 2000) it is proposed a decentralized and modular approach to perform failure diagnosis based on Sampath's results (Sampath, et al., 1995). In (Contant, et al., 2004) and (Pencolé, 2004) the authors presented incremental algorithms to perform diagnosability analysis based on (Sampath, et al., 1995) in a distributed way; they consider systems whose components evolve by the occurrence of events; the parallel composition leads to a complete system model intractable. In (Genc & Lafortune, 2003) it is proposed a method that handles the reachability graph of the *PN* model in order to perform the analysis similarly to (Sampath, et al., 1995); based on design considerations the model is partitioned into two labelled *PN* and it is proven that the distributed diagnosis is equivalent to the centralized diagnosis; later, (Genc & Lafortune, 2005) extend the results to systems modeled by several labelled *PN* that share places, and present an algorithm to determine distributed diagnosis. In (Qiu & Kumar, 2005) it is studied the codiagnosability property, this property guarantees that any faults occurred in the system must be detected by at least one local diagnoser in a finite number of steps using the local information, besides, a notion of safe-codiagnosability is mentioned to capture the fact that the system has a safe specification while the system performance is tolerable. (Arámburo-Lizárraga, et al., 2005) proposes a methodology for designing reduced diagnosers and presents an algorithm to split a global model into a set of communicating sub-models for building distributed diagnosers. The diagnosers handle a system sub-model and every diagnoser has a set of communication events for detecting and

Source: Advances in Robotics, Automation and Control, Book edited by: Jesús Arámburo and Antonio Ramírez Treviño,
ISBN 78-953-7619-16-9, pp. 472, October 2008, I-Tech, Vienna, Austria

locating the faults of the corresponding sub-model. (Arámburo-Lizárraga, et al., 2007) shows how to design low interaction distributed diagnosers reducing the communication among them and proposes a redundant distributed diagnoser scheme composed by a set of independent modules handling two kinds of redundancy (duplication or *TMR*).
This work considers the system modeled as an interpreted *PN* (*IPN*) allowing describing the system with partially observable states and events; the model includes the possible faults it may occur. In order to build such a model, this work presents a bottom-up modeling methodology in which the behavior of the system elements is decomposed into state variables; a range for each state variable must be settled. These ranges represent the possible values of state variables. Afterwards, these rages are coded into *IPN* (modules), where each value is represented by a different place. Then two composition operators for joining modules are used; the first one is named synchronic composition, which merges transitions according to certain rules. It is similar to the synchronic product presented in (Giua & DiCesare, 1994); the second one is named permissive composition, which uses selfloops for enabling transitions among modules; this allows constraining the model behavior into the actual system behavior. This new operator avoids the use of tuning phases of other modeling methods used in supervisory control (Giua & DiCesare, 1994). Based on the derived *IPN* model with the proposed methodology, the diagnosability property for *IPN* models is introduced. Roughly speaking, this property says that an *IPN* is diagnosable if it is possible to know both, when a faulty place is marked and which faulty place is marked. This property is closely related to the observability property (Aguirre-Salas, et al., 2002) and (Ramírez-Treviño, et al., 2003) and polynomial algorithms to test when an *IPN* is diagnosable are derived, avoiding the reachability analysis of other approaches. Also, a distributed diagnoser is presented; every distributed diagnoser uses the local information or communication among diagnosers for detecting and locating a system fault. The diagnosability property is preserved in the distributed architecture. Redundancy techniques could be applied to the distributed diagnosers to detect and locate a malfunction in the distributed diagnosers set.
The chapter is organized as follows: section 2 provides basic definitions of *PN*, *IPN* and the modeling methodology are presented. In section 3 the property of input-output diagnosability is defined and characterized, a diagnoser scheme devoted to detect and isolate failure states is also presented. Section 4 presents a procedure to build a reduced *IPN* model. Section 5 describes a method for model decomposition allowing interaction distributed diagnosers, also, it is presented a redundant scheme for reliable diagnosis applying redundancy to the distributed diagnosers. Finally, conclusions are given.

2. Basic notations and system modeling

2.1 Petri net basics

We consider systems modeled by Petri Nets and Interpreted Petri Nets. A Petri Net structure is a graph $G = (P, T, I, O)$ where: $P = \{p_1, p_2, ..., p_n\}$ and $T = \{t_1, t_2, ..., t_m\}$ are finite sets of nodes called respectively places and transitions, $I\ (O)$: $P \times T \rightarrow \mathbb{Z}^+$ is a function representing the weighted arcs going from places to transitions (transitions to places), where $\mathbb{Z}^+$ is the set of nonnegative integers.
The symbol $^\bullet t_j$ ($t_j^\bullet$) denotes the set of all places p_i such that $I(p_i,t_j)\neq 0$ ($O(p_i,t_j)\neq 0$). Analogously, $^\bullet p_i$ ($p_i^\bullet$) denotes the set of all transitions t_j such that $O(p_i,t_j)\neq 0$ ($I(p_i,t_j)\neq 0$) and the incidence matrix of G is $C=[c_{ij}]$, where $c_{ij} = O(p_i,t_j) - I(p_i,t_j)$.

A marking function M: $P \rightarrow \mathbb{Z}^+$ represents the number of tokens (depicted as dots) residing inside each place. The marking of a *PN* is usually expressed as an n-entry vector.
A Petri Net system or Petri Net (*PN*) is the pair $N=(G,M_0)$, where G is a *PN* structure and M_0 is an initial token distribution. $R(G,M_0)$ is the set of all possible reachable markings from M_0 firing only enabled transitions. In a *PN* system, a transition t_j is enabled at marking M_k if $\forall p_i \in P$, $M_k(p_i) \geq I(p_i,t_j)$; an enabled transition t_j can be fired reaching a new marking M_{k+1} which can be computed as $M_{k+1} = M_k + Cv_k$, where $v_k(i)=0$, $i \neq j$, $v_k(j)=1$.
Interpreted Petri Nets (*IPN*) (Ramírez-Treviño, et al., 2003) is an extension to *PN* that allow to associate input and output signals to *PN* models. An *IPN* (Q, M_0) is an Interpreted Petri Net structure where $Q = (G, \Sigma, \lambda, \varphi)$ with an initial marking M_0, G is a *PN* structure, $\Sigma = \{a_1, a_2, \ldots, a_r\}$ is the input alphabet of the net, where a_i is an input symbol, $\lambda: T \rightarrow \Sigma \cup \{\varepsilon\}$ is a transition labelling function with the following constraint: $\forall t_j, t_k \in T$, $j \neq k$, if $\forall p_i\ I(p_i,t_j) = I(p_i,t_k) \neq 0$ and both $\lambda(t_j) \neq \varepsilon$, $\lambda(t_k) \neq \varepsilon$, then $\lambda(t_j) \neq \lambda(t_k)$; ε represents an internal system event, and $\varphi: R(Q,M_0) \rightarrow (\mathbb{Z}^+)^q$ is an output function that associates to each marking an output vector, where q is the number of outputs. In this work φ is a $q \times n$ matrix. If the output symbol i is present (turned on) every time that $M(p_j) \geq 1$, then $\varphi(i,j)=1$, otherwise $\varphi(i,j)=0$.
A transition $t_j \in T$ of an *IPN* is enabled at marking M_k if $\forall p_i \in P$, $M_k(p_i) \geq I(p_i,t_j)$. If t_j is enabled at marking M_k, and $\lambda(t_j)$ is present, then t_j can be fired reaching M_{k+1}, i.e., $M_k \xrightarrow{t_j} M_{k+1}$; M_{k+1} can be computed using the state equation:

$$\begin{aligned} M_{k+1} &= M_k + Cv_k \\ y_k &= \varphi(M_k) \end{aligned} \tag{1}$$

where C and v_k are defined as in *PN* and $y_k \in (\mathbb{Z}^+)^q$ is the k-th output vector of the *IPN*.
The sequence $\sigma = t_i t_j \ldots t_k \ldots$ is a firing transition sequence of an *IPN* (Q,M_0) if $M_0 \xrightarrow{t_i} M_1 \xrightarrow{t_j} \ldots M_x \xrightarrow{t_k} \ldots$ According to functions λ and φ, transitions and places of an *IPN* (Q,M_0) are classified. If $\lambda(t_i) \neq \varepsilon$ the transition t_i is said to be manipulated. Otherwise it is non-manipulated. A place $p_i \in P$ is said to be measurable if the i-th column vector of φ is not null, i.e. $\varphi(\bullet,i) \neq 0$. Otherwise it is non-measurable. The following concepts are useful in the study of the diagnosability property.
The set $\pounds(Q,M_0) = \{\sigma = t_i t_j \ldots t_k \ldots \wedge M_0 \xrightarrow{t_i} M_1 \xrightarrow{t_j} \ldots M_x \xrightarrow{t_k} \ldots\}$ of all firing transition sequences is called the firing language of (Q,M_0).
A sequence of input-output symbols of (Q,M_0) is a sequence $\omega = (a_0,y_0)(a_1,y_1)\ldots(a_n,y_n)$, where $a_j \in \Sigma \cup \{\varepsilon\}$. The symbol a_{i+1} is the current *IPN* input when the output changes from y_i to y_{i+1}. It is assumed that $a_0 = \varepsilon$ and $y_0 = \varphi(M_0)$.
The firing transition sequence $\sigma \in \pounds(Q,M_0)$ whose firing actually generates ω is denoted by σ_ω. The set of all possible firing transition sequences that could generate the word ω is defined as $\Omega(\omega) = \{\sigma \mid \sigma \in \pounds(Q,M_0) \wedge$ the firing of σ produces $\omega\}$.
The set $\Lambda(Q,M_0) = \{\omega \mid \omega$ is a sequence of input-output symbols$\}$ denotes the set of all sequences of input-output symbols of (Q,M_0) and the set of all input-output sequences of length greater or equal than k will be denoted by $\Lambda^k(Q,M_0)$, i.e. $\Lambda^k(Q,M_0) = \{\omega \in \Lambda(Q,M_0) \mid |\omega| \geq k\}$ where $k \in \mathbb{N}$.

The set $\Lambda_B(Q,M_0)$ = {ω ∈ $\Lambda(Q,M_0)$ | σ∈Ω(ω) such that $M_0 \xrightarrow{\sigma} M_j$ and M_j enables no transition, or when $M_j \xrightarrow{t_i}$ then $C(\bullet,t_i)=\vec{0}$ } denotes all input-output sequences leading to an ending marking in the *IPN* (markings enabling no transition or only self-loop transitions).

An *IPN* (Q, M_0) described by the state equation (1) is event-detectable iff the firing of any pair of transition t_i, t_j of (Q, M_0) can be distinguished from each other by the observation of the sequences of input-output symbols.

The following lemma (Rivera-Rangel, et al., 2005) gives a polynomial characterisation of event-detectable *IPN*.

Lemma 1: A live *IPN* given by (Q,M_0) is event-detectable if and only if:

1. $\forall t_i, t_j \in T$ such that $\lambda(t_i) = \lambda(t_j)$ or $\lambda(t_i) = \varepsilon$ it holds that $\varphi C(\bullet,t_i) \neq \varphi C(\bullet,t_j)$ and
2. $\forall t_k \in T$ it holds that $\varphi C(\bullet,t_k) \neq 0$.

2.2 System modeling

We work with the modeling methodology proposed in (Ramírez-Treviño, et.al.,2007). The methodology follows a modular bottom-up strategy. After identifying the system components, a set of state variables is assigned to every component, each state variable behavior is modeled by a *PN* model, herein named module. Then the set of modules are integrated into a single model according to the appropriate relationships achieved through two module composition operations. This methodology builds binary *IPN* modules to represent the behavior of each component of the identified *DES* and the relationships between them. The model captures the normal and faulty behavior of the individual components of the system. Below it is described the detailed steps of the methodology.

2.2.1 Modeling methodology

1. System components.- The system components must be identified and named. Afterwards, a finite set System_Components= {sc_1, sc_2,…, sc_n} of these names must be created. A system component could be a valve, a motor, a system resource, etc.
2. State variables.- For each system component, the different variables needed to represent its behavior must be chosen. In other words, the finite set State_Variables$_i$= {sv^i_1, sv^i_2,…, sv^i_m} associated to the system component sc_i∈ System_Components must be built. These variables could represent the position (for instance a valve position), velocity, voltage, etc. of each system component, or could represent a task descriptor (for instance a machine state). There exists at least one state variable for each system component.
3. Set of values.- For each state variable sv^i_j ∈ State_Variables$_i$, the set $Value_{sv^i_j} = \{val^{ij}_1, val^{ij}_2, ..., val^{ij}_p\}$ of possible values of sv^i_j must be stated. Necessary faulty values should also be considered in this set. For instance, the variable "valve_position" may take four values: "Open", "Closed", "ErrorOnOpen" and "ErrorOnClosed", or the variable "task_machine1" may take three values: "loading", "processing", and "unloading".
4. Codification.- The values in each set $Value_{sv^i_j}$ must be represented in terms of *PN* markings. This can be easily achieved if binary places are used. Thus, for each sv^i_j ∈

State_Variables_i a set $P_{sv_j^i} = \{p_1^{ij}, p_2^{ij}, ..., p_n^{ij}\}$ of places such that $|Value_{sv_j^i}| = |P_{sv_j^i}|$ must be created. The marking of these places is binary and mutually exclusive. Then, $M(p^{ij}_z)=1$ means that the variable sv^i_j takes the value val^{ij}_z. Because of the existence of faulty values, the set of places can be partitioned into the subsets $P^F_{sv_j^i}$ and $P^N_{sv_j^i}$, representing the faulty and normal values respectively.

5. Event modeling.- For each pair of values val^{ij}_m, val^{ij}_n such that the state variable sv^i_j could change from value val^{ij}_m to a value val^{ij}_n, a transition t^{ij}_{mn} must be created. Then, one arc going from place p^{ij}_m to transition t^{ij}_{mn} and one arc going from transition t^{ij}_{mn} to place p^{ij}_n must be added.
6. Initial marking.- The initial marking is defined as: $M_0(p^{ij}_m)=1$ if the initial value of the variable sv^i_j is val^{ij}_m and $M_0(p^{ij}_m)=0$ otherwise.
7. Output.- The output of this algorithm is a set of isolated PN modules, each one modeling the behavior of a state variable sv^i_j.
8. Perform synchronous composition and permissive among modules *IPN* to obtain the global *IPN* model system (see Alcaraz-Mejía et al. 2003) and (Ramírez-Treviño et al. 2007).

In the *IPN* system model the sets of nodes are partitioned into faulty nodes (P^F, places coding faulty states, and T^F, transitions leading to faulty states) and normal functioning nodes (P^N and T^N); so $P = P^F \cup P^N$ and $T = T^F \cup T^N$. p^N_i denotes a place in P^N. Since $P^N \subseteq P$ then p^N_i also belongs to (Q, M_0). The set of risky places of (Q, M_0) is $P^R = {}^\bullet T^F$. The post-risk transition set of (Q, M_0) is $T^R = P^{R\bullet} \cap T^N$. (Q^N, M^N_0) denotes the embedded normal behavior of (Q, M_0), i.e., (Q^N, M^N_0) is the subnet induced by normal nodes.

Example 1. Consider the producer-consumer scheme depicted in figure 1.

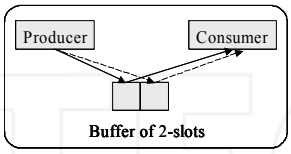

Fig. 1. Producer-Consumer with buffer of 2-slots scheme

The model consists of a producer unit (PU), a consumer unit (CU) and a buffer of 2-slots. The behavior of this system is the following. The producer unit PU creates and delivers products into the free buffer positions. The consumer unit CU retrieves products from the buffer when there is a product stored into a buffer slot. The producer unit PU could reach a faulty state from its producing state. Similarly, the consuming unit could reach a faulty state from its consuming state. We model each system component isolately (see figure 2), then the synchronous composition and permissive operations are applied to obtain the global system model of the producer-consumer scheme. In figure 3 is depicted the normal system behavior model; figure 4 includes the faulty behavior, in this case when PU stops its production and

CU stops its consumption. The places p_1, p_2, p_3 represent the normal PU behavior and p_{11} represents the faulty behavior. Places p_4, p_5, p_6 represent the normal CU behavior and p_{12} represents the faulty behavior. The places p_7, p_8, p_9 and p_{10} represent the 2- slots of the buffer. Function λ is defined as $\lambda(t_1)=a$, $\lambda(t_8)=b$; and $\lambda(t_i)=\varepsilon$ for others transitions. Measurable places are p_3, p_6, p_8, p_{10}, $P^R=\{p_3,p_6\}$, $T^R=\{t_1,t_8\}$, $T^F=\{t_9,t_{10}\}$ and $P^F=\{p_{11},p_{12}\}$.

3. Centralized diagnosability

The characterisation of input-output diagnosable *IPN* is based on the partition of $R(Q,M_0)$ into normal and faulty markings where all the faulty markings must be distinguishable from other reachable markings.

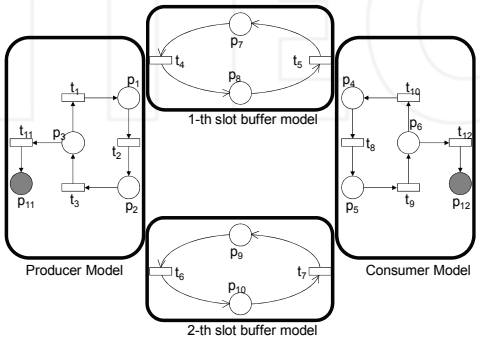

Fig. 2. *IPN* modules of the identified components

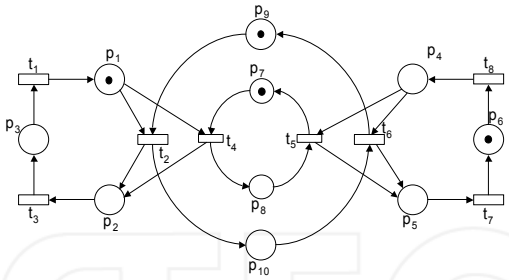

Fig. 3. Normal behavior of the *IPN* of the producer-consumer scheme

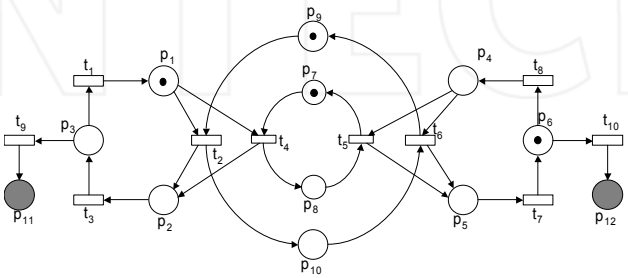

Fig. 4. Normal and Faulty behavior of the *IPN* of the producer-consumer scheme

Definition 1: An *IPN* given by (Q,M_0) is said to be input - output diagnosable in $k < \infty$ steps if any marking $M_f \in F$ is distinguishable from any other $M_k \in R(Q,M_0)$ using any word $\omega \in \Lambda^k(Q,M_f) \cup \Lambda_B(Q,M_f)$, where $F = \{M \mid \exists p_k \in P^F$ such that $M(p_k)>0, M \in R(Q,M_0)\}$.
The following result provides sufficient structural conditions for determining the input-output diagnosability of an *IPN* model.

Theorem 1: Let (Q,M_0) be a binary *IPN*, such that $\left(Q^N, M_0^N\right)$ is live, strongly connected and event detectable. Let $\{X_1,...,X_\tau\}$ be the set of all T-semiflows of (Q,M_0). If $\forall\, p_i^N \in P^N$, $\left(p_i^N\right)^\bullet \cap T^F \neq \varnothing$ the following conditions hold:

1. $\forall r, \exists j\ X_r(j) \geq 1$, where $t_j \in \left(p_i^N\right)^\bullet - T^F$,
2. $\forall t_k \in \left(p_i^N\right)^\bullet - T^F$, $^\bullet(t_k)=\{\ p_i^N\ \}$ and $\lambda(t_k) \neq \varepsilon$.

then the *IPN* (Q,M_0) is input-output diagnosable.

Proof: Assume that (Q,M_0) meets all conditions of the theorem. Since $\left(Q^N, M_0^N\right)$ is live, then it is live by places (see Desel & Esparza, 1995), i.e. $\forall p \in P$ there exists a marking M_k such that $M_k(p_i)>0$, previous observation is also valid for any $p_i^N \in P^N$, $\left(p_i^N\right)^\bullet \cap T^F \neq \varnothing$. Thus there exists σ_ω such that $M_0 \xrightarrow{\sigma_\omega} M_k$ and $M_k\left(p_i^N\right)=1$. The input-output symbol word generating σ_ω will be denoted by ω. Choose the longest transition firing sequence σ_ω not including transitions in T^F (could be the empty one). Such sequence must be marking a place $p_i^N \in P^N$, $\left(p_i^N\right)^\bullet \cap T^F \neq \varnothing$ since in the next step a faulty transition should be fired.

Since the initial marking is known, $\left(Q^N, M_0^N\right)$ is event detectable, and σ_ω contains no faulty transitions, then the marking M_k such that $M_0 \xrightarrow{\sigma_\omega} M_k$, can be computed using the *IPN* state equation.

Since p_i^N has a token at M_k, then a transition $t_i^F \in \left(p_i^N\right)^\bullet \cap T^F$ could be fired reaching a faulty marking M_j. Thus, we will prove that it is possible to detect that M_j is reached.

Assume that such transition t_i^F at M_k. Thus $M_0 \xrightarrow{\sigma_\omega} M_k \xrightarrow{t_i^F} M_j$ and M_j adds a token into a place $p_j^F \in \left(p_i^N\right)^{\bullet\bullet} \cap P^F$ and removes a token from the place p_i^N. Since the use of the modeling methodology assures that $\lambda\left(t_i^F\right)=\varepsilon$ and $\varphi\left(M_k\right)=\varphi\left(M_j\right)$, then the firing of t_i^F cannot be detected using the current sequence of input-output symbols, i.e. it is not possible to determine when the faulty marking M_j was reached. In other words, the input-output symbol word ω generates both σ_ω and $\sigma_\omega t_i^F$. In order to detect when M_j is reached, we proceed as follows.

Since it holds that $\forall r, \exists j\ X_r(j) \geq 1$, $t_j \in T - T^F$, then no T-semiflow can be fired without firing a transition in $\left(p_i^N\right)^\bullet - T^F$. Moreover, since the number of transitions in (Q,M_0) is finite, then the length of a sequence attempting to fire a transition $\left(p_i^N\right)^\bullet$ should be finite. Then a transition $t_k \in \left(p_i^N\right)^\bullet \cap T^N$ must be attempted to fire after the firing of a finite sequence.

Since the current marking in the net is the faulty marking M_j, then t_k cannot be fired. Since the condition 2 indicates that $\lambda\left(t_k\right) \neq \varepsilon$, then we can detect when the symbol $\lambda\left(t_k\right)$ is given to

the system. Moreover, since $^{\bullet}(t_k)=\{p_i^N\}$, if t_k cannot be fired, then certainly that $M_j(p_i^N)=0$ and that $M_j(p_j^F)=1$, where $p_j^F \in (p_i^N)^{\bullet\bullet} \cap P^F$. Thus the faulty marking M_j is detected, i.e. the *IPN* is input-output diagnosable. □

3.1 Diagnosability test

Determining when an *IPN* is input-output diagnosable is reduced to the following tests.

1. Binarity of (Q,M_0). It is fulfilled because of the modeling methodology.
2. Liveness of modules can be tested efficiently; however the property is not preserved during arbitrary module composition operators. Thus some constraints in the application of synchronous and permissive compositions are introduced to guarantee liveness during composition. Such constraints may follow the rules introduced in (Koh & DiCesare, 1991) for module composition.
3. Event detectability and strongly connectedness are determined in polynomial time as well as detecting places $p_i^N \in P^N$ such that $(p_i^N)^{\bullet} \cap T^F \neq \emptyset$. Then condition 2 of previous theorem is efficiently tested.
4. Finally, condition 1 can be verified in polynomial time. In this case we need to check that there exists no T-semiflow that does not include transitions in $(p_i^N)^{\bullet}$, $p_i^N \in P^N$ such that $(p_i^N)^{\bullet} \cap T^F \neq \emptyset$. Thus we need to check that the following linear programming problems have no solutions.

$\forall p_i^N \in P^N$ such that $p_i^N \in P^N, (p_i^N)^{\bullet} \cap T^F \neq \emptyset$

$\nexists X$

s.t.

$CX=0$

$x(j)=0, \forall t_j \in (p_i^N)^{\bullet} - T^F$

There exist different strategies for constructing the diagnoser-model, we presented two different ways: a centralized and a distributed diagnoser. A centralized diagnoser model is composed by a copy of the normal system behavior of (Q,M_0). Also a reduced diagnoser model can be built. Also, this work handles a distributed diagnoser which is very useful for large and complex system besides incorporate reliability to the system diagnosis process through redundancy techniques applied to the distributed diagnoser model. The different diagnoser models are defined in the following sections.

3.2 Centralized diagnoser design

Diagnosability theorem given above provides the basis for designing an on-line diagnoser (see figure 5). Such diagnoser can detect a fault and locate faulty markings reached by an *IPN*. The proposed scheme for diagnosis (Ramírez-Treviño, et al., 2004) handles a copy of the normal behavior model which must evolve similarly to the system; the outputs of both the system and the model are compared and, when there is a difference, a procedure is started to compute the faulty marking.

Example 2. The *IPN* system model depicted in figure 3 represents the normal behavior model for the producer-consumer diagnoser. The initial marking M_0 represents that all the

buffers are empty, the PU is waiting to deliver and the CU state is idle. Since this *IPN* is input-output diagnosable by theorem 1, then we can detect and locate the fault with the diagnoser. The system and its on-line diagnoser are depicted in figure 6, notice the diagnoser is a copy of the system. Now assume that the events represented by the sequence t_2t_3 are executed into the system, then the sequence t_2t_3 is fired in the diagnoser model. Thus both, the system and the diagnoser-model have the same output "producing" and "consuming". If the fault transition t_{11} is fired, then p_{11} is marked in the system and no system output change is detected. When the symbol of t_1, $\lambda(t_1)$ is given as input to the system, then the diagnoser-model evolves and its new output is "consuming", however the output of the system continues in "producing" and "consuming". Then the difference is the signal "producing"; thus the algorithm determines that p_{11} is marked, thus the fault is isolated.

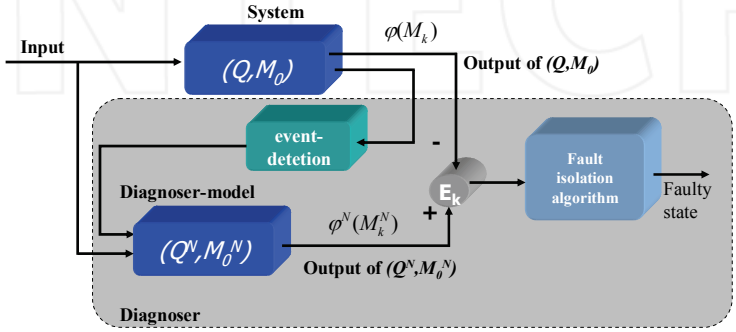

Fig. 5. On-line diagnose scheme

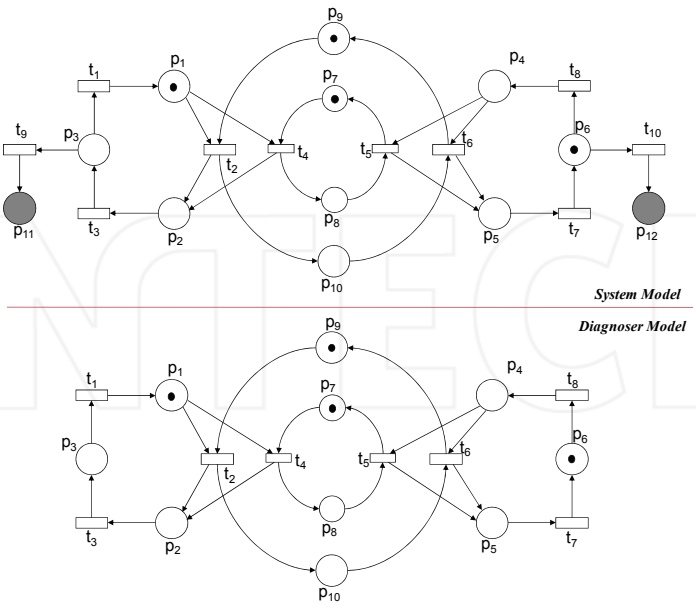

Fig. 6. On-line diagnoser model

4. Reduced diagnoser

4.1 Diagnoser model

Definition 2: The proposed diagnoser model structure for the system normal behavior (Q^N, $M_0{}^N$) is an *IPN* (Q^d, $M_0{}^d$) where the set of places is $P^d = \{p_d\}$ and the set of transitions is $T^d = T^N$, the incidence matrix C^d of (Q^d, $M_0{}^d$) is the following

$$C^d = B^T \varphi^N C^N \tag{2}$$

where B^T is a q×1 vector (q is the number of measurable places of ((Q^N,$M_0{}^N$))), φ^N is the output matrix of (Q^N, $M_0{}^N$), C^N is the incidence matrix of (Q^N, $M_0{}^N$).
The initial marking of the diagnoser model structure for one place is computed as:

$$M_0^d = B^T \varphi^N \left(M_0^N \right) \tag{3}$$

We propose a matrix B that is computed as follows:
Algorithm 1: Building B

Inputs: C-incidence matrix of an *IPN*,
l - number of places in the diagnoser-model,
q - number of measurable places in the *IPN*,
Outputs: A matrix B

1. The "base number" b should be computed. In this case $b=2max(abs(c_{ij}))+1$, where c_{ij} is an element of incidence matrix C.
2. Define a q × 1 vector.
3. $\left[b^0 \quad b^1 \quad \cdots \quad b^{q-1} \right]$

This procedure computes matrix B.
According to the way in which B was constructed, all columns of C^d must different from zero and different from each other.
If a transition $t_i \in T - (T^R \cup T^F)$ is fired in (Q,M_0) then it is fired in (Q^d, $M_0{}^d$) (it is possible since these transitions are event detectable, thus the output system information is enough to detect when one of these transition is fired).
If a transition $t_j \in T^R$ is enabled in (Q_d, $M_0{}^d$) and $\lambda(t_j)$ is activated in (Q,M_0), then t_j must be fired in (Q^d, $M_0{}^d$). Thus, if t_j is not fired in (Q,M_0), then (Q,M_0) reached a faulty marking. In this case the output of the system and the output of the diagnoser are different from each other.

4.2 Error computation and fault isolation

Definition 3. Error computation. The k-th error is computed by the following equation:

$$e_k = M_k{}^d - B^T(\varphi M_k) \tag{5}$$

Notice that e_k is computed from the diagnoser-model output and not from the marking M_k. It means that the proposed diagnoser is using the system output and not internal system signals (those signals that are non measurable).
Definition 4. Fault isolation. When $e_k \neq 0$, an error is detected, then a faulty marking was reached. The mechanism used to find out the faulty marking is named fault isolation. This work proposes the following algorithm to accomplish this task.

Algorithm 2: Fault isolation

Inputs: M_k , M_k^d , e_k.
Outputs: p(faulty place), M_f(faulty marking)
Constants: C^d is the IPN diagnoser structure incidence matrix
i = index of the column of C^d such that $C^d(1,i)= e_k$

- $\forall p \in \cdot t_i$, $M_k(p)=0$
- $\forall p \in t_i \cdot$, $M_k(p)=0$
- $\forall p^F \in ({}^\bullet t_i)^{\bullet\bullet} \cap P^F$, $M_k(p^F)=1$
- $M_f = M_k$
- Return (p, M_f)

Example 3. Consider the system of the example 1. The *IPN* depicted in figure 4 represents the behavior of the system of example 1. Since this *IPN* is input-output diagnosable by theorem 1, then a diagnoser can be built for this system. In this case we will use the reduced structure presented in this section.

$$C^N = \begin{bmatrix} 1 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & -1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \end{bmatrix}; \phi = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The base obtained to compute B is b=2*1+1=3; since we build B using algorithm 9. We obtain the following vector:

$$B^T = \begin{bmatrix} 1 & 3 & 9 & 27 \end{bmatrix}^T$$

Therefore C^d *is:*

$$C^d = \begin{bmatrix} -1 & 27 & 1 & 9 & -9 & -27 & 3 & -3 \end{bmatrix}$$

Hence, its associated *IPN* is depicted in figure 7.

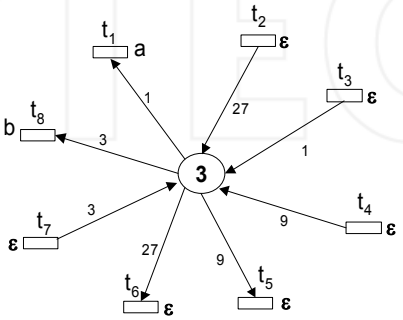

Fig. 7. The *IPN* reduced diagnoser-model

The initial marking of the diagnoser is M_0^d =[3]. In order to show how the diagnoser works, assume that the following sequence t_2t_3 is executed into the system, then this sequence is fired in the diagnoser. Thus the system output is $(\varphi(M_k))^T = [1 \quad 1 \quad 0 \quad 1]$, and the marking of the *IPN* diagnoser is M_k^d =[3 1]. Then $e_k = M_k^d - (B^T)(\varphi M_k)$ = [31] – [31] = 0, thus the system is in a normal state. Now if the faulty transition t_9 is fired, then p_{11} is marked, however no change in the output system is detected. If the symbol of t_1 ($\lambda(t_1)$=a) is given as input to the *IPN* of the system model and *IPN* diagnoser, then the diagnoser evolves, and $M_{k+1}{}^d$ =[30]. Then e_k = –1 indicating the existence of an error. The fault isolation algorithm (algorithm 2) detects that the column 1 of C^d is equal to e_k, thus t_1 was not fired in the system. Then the same algorithm detects the faulty marking and determines that the faulty place p_{11} is marked.

5. Distributed diagnoser

5.1 Model distribution

In order to build a distributed diagnoser, the *IPN* model (Q, M_0) can be conveniently decomposed into m interacting modules where different modules share nodes (transitions and/or places).

Definition 5. Let (Q,M_0) be an *IPN*. A module μ_k = (N_k, Σ_k, λ_k, φ_k) is an *IPN* subnet of the global model (Q,M_0), where:

- N_k = (T_k, P_k, I_k, O_k, I_k^C, O_k^C, M_{0k}) where:
  - $T_k \subseteq T$,
  - $P_k = P_k^L \cup P_k^C$; $P_k^L \subseteq P$; P_k^C *represents the communication places among modules; this set is a copy of some places* P_l^L *that belongs to other modules,* $l \neq k$. P_k^C *is the minimal places that is required for the transitions of the module are event-detectable.* $M(P_k^C) = M(P_l^L)$.
  - $I_k(O_k)$: $P_k^L \times T_k \rightarrow Z^+$, *s.t.,* $I_k(p_i,t_j) = I(p_i,t_j)$ $(O_k(p_i,t_j) = O(p_i,t_j))$, $\forall p_i \in P_k^L$ and $\forall t_j \in T_k$.
  - $I_k^C(O_k^C)$: $P_k^C \times T_l \rightarrow Z^+$, *s.t.,* $I_k^C(p_i,t_j) = I(p_i,t_j)$ $(O_k^C(p_i,t_j) = O(p_i,t_j))$, $\forall p_i \in P_k^C$ and $\forall t_j \in T_l$, $l \neq k$. $I_k^C(O_k^C)$ *are the input(output) arcs to from* $p_i \in P_k^C$ *to transitions of other modules.*
  - $M_{0k} = M_0 \mid_{Pk}$
- $\Sigma_k = \{\alpha \in \Sigma \mid \exists t_i, t_i \in T_k, \lambda(t_i) = \alpha\}$
- $\lambda_k : T_k \rightarrow \Sigma_k \cup \{\varepsilon\}$, s.t. $\lambda_k(t_i) = \lambda(t_i)$ and $t_i \in T_k$
- $\varphi_k : R(\mu_k, M_{0k}) \rightarrow (Z^+)^{qk}$, q_k is restricted to the outputs associated to P_k.

Definition 6. Let (Q,M_0) be an *IPN*. A distribution DN_i of (Q,M_0) is a finite set of m *modules*, i.e., DN_i={μ_1, μ_2,…,μ_m}. The distribution DN_i holds the following conditions:

1. $\bigcap_{k=1}^{m} P_k^L = \phi; \bigcup_{k=1}^{m} P_k^L = P$ (It is a partition over P_k^L).
2. $\bigcup_{k=1}^{m} T_k = T$ (It is not a partition)

The set of communication places $P_{com} = \bigcup_{k=1}^{m} P_k^C$ of a distribution represents the measurable places of each module $\mu_k \in DN_i$ needed to guarantee the event-detectability property of μ_k.

Definition 7. Let DN_i be a distribution of (Q,M_0) and $\sigma = t_it_j \ldots t_k \ldots$ be a sequence of £(Q,M_0). We define the natural projection PT_k of £(Q,M_0) over the languages of the modules $\mu_k \in DN_i$ of the following way:

$$PT_k: £(Q,M_0) \rightarrow £(\mu_k,M_{0k})$$

$$\forall t_it_j \ldots t_st_q \in £(Q,M_0)$$

$$PT_k(\varepsilon) = \varepsilon$$
$$PT_k(t_i t_j ... t_s t_q) = \begin{cases} PT_k(t_i t_j ... t_s) & if\ t_q \notin T_k \\ PT_k(t_i t_j ... t_s) t_q & if\ t_q \in T_k \end{cases}$$

There exists the input-output symbol projection over the input and output module symbols. Let ω = $(a_0,y_0)(a_1,y_1)...(a_n,y_n)$ be a sequence of symbols of (Q,M_0)

$P\Lambda_{\mu k}(\omega) = \left(\left(P_{IN\,k}\alpha_0, P_{OUT\,k} y_o \right), \left(P_{IN\,k}\alpha_1, P_{OUT\,k} y_1 \right) .. \left(P_{IN\,k}\alpha_n, P_{OUT\,k} y_n \right) \right)$ where:

$$P_{INk}(\alpha_i) = \begin{cases} \varepsilon & if\ \ \alpha_i \notin \sum_k \\ \alpha_i & if\ \ \alpha_i \in \sum_k \end{cases} \text{ and } P_{OUTk}\left(y_i = \begin{bmatrix} y_i(1) \\ \vdots \\ y_i(q) \end{bmatrix} \right) = \begin{bmatrix} y_i'(1) \\ \vdots \\ y_i'(q) \end{bmatrix}$$

where $y_i'(s) = 0$ if the measurable place does not belong to μ_k

where $y_i'(s) = y_i(s)$ otherwise.

Example 4. Consider the *IPN* system model depicted in the figure 4. We partition the producer-consumer model to obtain a distributed model. The figure 8 depictes a distribution DN_i. For the sake of simplicity, we use in the example the same names for duplicated nodes (places or transitions) belonging to different modules. The distribution has three modules, i.e., $|DN_i| = 3$; $I_k^C(O_k^C)$ is represented by the dashed arcs. The module μ_1 has the transitions $T_1 = \{t_1,t_2,t_3,t_9\}$ and the place set $P_1 = \{p_1,p_2,p_3,p_{11}\} \cup \{p_{10}\}$. We are preserving the property of event detectability of common transitions duplicating measurable places, and using these copies in different modules.

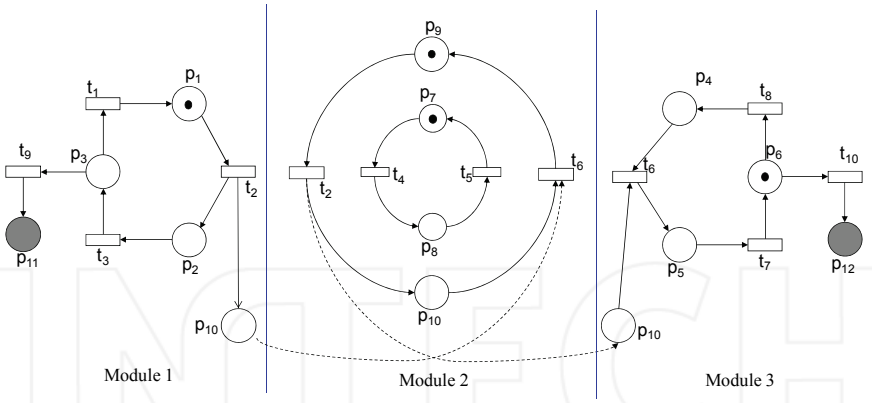

Fig. 8. Distributed Interpreted Petri Net Models

5.2 Distributed input-output diagnosability

The results of centralized diagnosability are applied to the modules.

A module is locally diagnosable if, for every local fault we can detect it only using local information, else it is conditionally diagnosable.

Definition 8. (Local Diagnosability) A module $\mu_n \in DN_i$ is said to be locally input-output diagnosable in $k < \infty$ steps if any faulty marking $M_{fn} \in R(\mu_n, M_{0n})$ is distinguishable from any other $M_{kn} \in R(\mu_n, M_{0n})$ using words ω_n, where $P\Lambda_{\mu n}(\omega) = \omega_n$.

Definition 9. (Conditional Diagnosability) A module $\mu_n \in DN_i$ is said to be conditional input-output diagnosable in $k < \infty$ steps if any faulty marking $M_{fn} \in R(\mu_n, M_{0n})$ is distinguishable from any other $M_{kn} \in R(\mu_n, M_{0n})$ using words ω_n and ω_m, where $P\Lambda_{\mu n}(\omega) = \omega_n$ and $P\Lambda_{\mu m}(\omega) = \omega_m$, ω_m denotes the set of all input-output sequences that lead to a marking a duplicate place in μ_m, where $\mu_n \neq \mu_m$.

5.3 Communication channels

Definition 10. The border transitions between two modules μ_k and μ_z is: $T^{kz}_{border} = \{t_i \mid t_i \in \{T_k \cap T_z\}\}$. This concept can be extended to several modules.

The communication channels between two modules are represented by $I_k^C(O_k^C)$. We assume that every module can communicate with every other module. The firing of a transition $t_j \in T_k$ may be local to the module μ_k and cause only a local marking change, or it may involve communication with another module if $t_j \in T^{kz}_{border}$. Every time a transition $t_j \in T^{kz}_{border}$ is fired in μ_k then a message is sent to every μ_z containing the same $t_j \in T^{kz}_{border}$ to put a token in some place $p_x \in P_z^C$ such that $p_x \in {}^\bullet t_j$. In (Lampson, 1993) it is proposed different ways to create protocols for implementing reliable messages.

5.4 Redundant diagnoser

Since distributed diagnosers leads to the use of several computers (*CPU*), then, redundancy can be introduced in the diagnosers. For instance, the Triple Modular Redundancy (*TMR*) can be used in this case.

Assume that a distribution $DN_i=\{\mu_1, \mu_2,\ldots,\mu_m\}$ was obtained and that it was distributed over *m* computers (see figure 9).

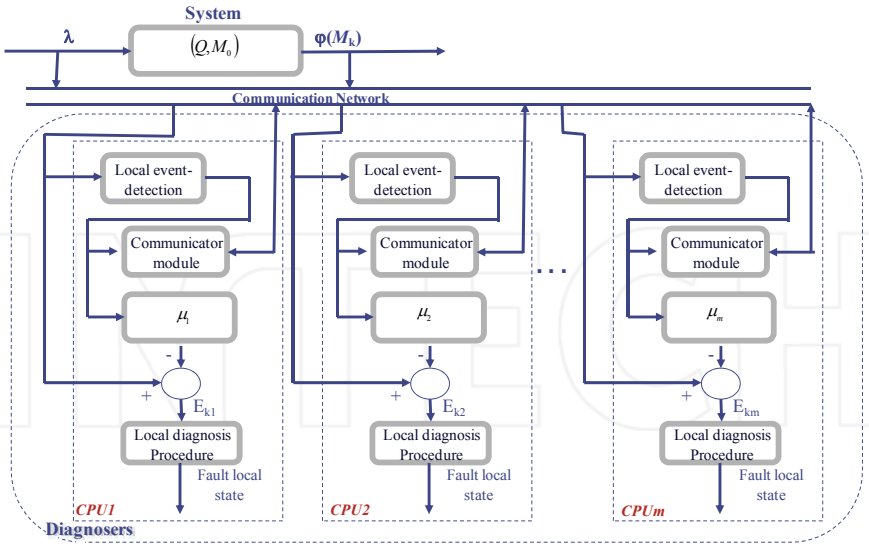

Fig. 9. On-Line Distributed Diagnoser

Then, a *TRM* scheme can be also applied to figure 9 (see figure 10), increasing the diagnoser reliability.

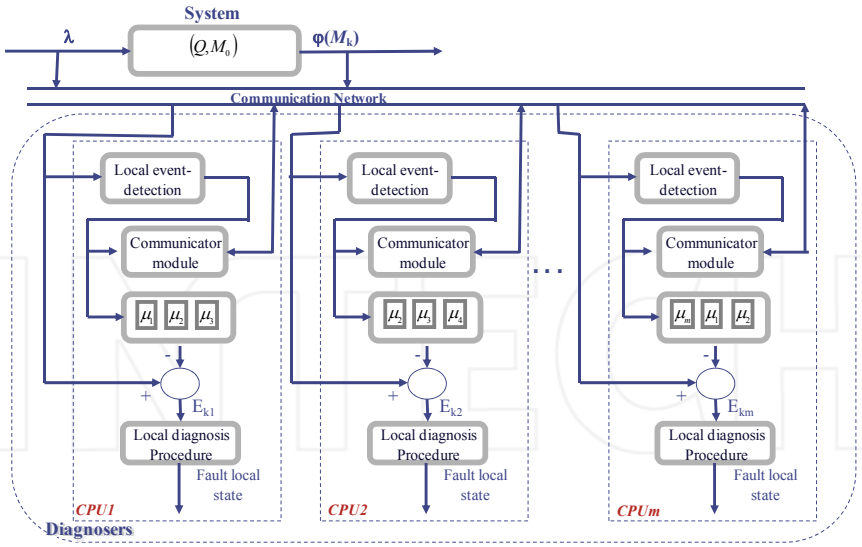

Fig. 10. On-Line Redundant Distributed Diagnoser

6. Conclusions

This chapter introduced the diagnosability property in *DES* modeled using *IPN*. It presented a structural characterization of this property using the T-semiflows of the *IPN*.

The approach herein presented exploits the *IPN* structure to determine when it is diagnosable, this approach leads to polynomial characterization of diagnosability.

Based on the *DES* model, three different types of diagnosers were presented. The first one was a centralized version, allowing to detect and locate faults. Sometimes, however, the system could very large; leading to large diagnoser models thus the other two diagnosers are designed to tackle this problem.

The second diagnoser is a reduced scheme. It uses one place; however the number of tokens could be large. The third diagnoser is a distributed one, were the diagnoser model is distributed over different computers. Adopting this approach the problems that appear in centralized versions are eliminated. Moreover, in this last case, a redundant diagnoser can be used for increasing the reliability of the distributed diagnosers.

7. References

Aguirre-Salas L., O. Begovich, A. Ramírez-Treviño (2002). "Observability in Interpreted Petri Nets using Sequence Invariants". Proc. of the IEEE Control and Decision Conference, pp. 3602-3607, Las Vegas, USA. December 2002.

Alcaraz-Mejía M., E. López-Mellado, Antonio Ramírez-Treviño, I. Rivera-Rangel (2003). "Petri Net Based Fault Diagnosis of Discrete Event Systems", Proc. of the IEEE International Conference on Systems, Man and Cybernetics. pp. 4730-4735, October 2003.

Arámburo-Lizárraga J., E. López-Mellado and A. Ramírez-Treviño (2005). "Distributed Fault Diagnosis using Petri Net Reduced Models". *Proc. of the IEEE Int. Conf. on Systems, Man and Cybernetics*. pp. 702-707, October 2005.

Arámburo-Lizárraga J., E. López-Mellado, and A. Ramírez-Treviño (2007). "Design of Low Interaction Distributed Diagnosers for Discrete Event Systems". Proc. of the 4th International Conference on Informatics in Control, Automation and Robotics. pp. 189-194. Angers, France. May 2007.

Arámburo-Lizárraga J., E. López-Mellado, A. Ramírez-Treviño and E. Ruiz-Betlrán (2007). "Reliable Distributed Fault Diagnosis using Redundant Diagnosers". Proc. of 1st IFAC Workshop on Dependable Control of Discrete Systems (DCDS'07). Cachan, France. June, 2007.

Benveniste A., S. Haar, E. Fabre and C. Jara (2003). "Distributed and Asynchronous Discrete Event Systems Diagnosis". *42nd IEEE Conference on Decision and Control*. 2003.

Contant O, S. Lafortune and D. Teneketzis (2004). "Diagnosis of modular discrete event systems". *7th Int. Workshop on Discrete Event Systems* Reims, France. September, 2004.

Debouk R, S. Lafortune and D. Teneketzis (2000). "Coordinated Decentralized Protocols for Failure Diagnosis of Discrete Event Systems", Kluwer Academic Publishers, Discrete Event Systems: Theory and Applications, vol. 10, pp. 33-79, 2000.

Desel J. and J. Esparza(1995). "Free Choice Petri Nets". Cambridge University Press, 1995.

Genc S. and S. Lafortune (2003). "Distributed Diagnosis of Discrete-Event Systems Using Petri Nets" *Proc. of the 24th. ATPN* pp. 316 - 336, June, 2003.

Genc S. and S. Lafortune (2005). "A Distributed Algorithm for On-Line Diagnosis of Place-Bordered Nets". 16th IFAC World Congress, Praha, Czech Republic, July 04-08, 2005.

Giua A. and F. DiCesare. "Petri Net Structural Analysis for Supervisory Control". IEEE Transactions on Robotics and Automation, Vol.10, No.2, pp. 185-195, April 1994.

Jalote P. (1994). "Fault Tolerance in distributed systems". Prentice Hall. 1994

Jiroveanu G. and R. K. Boel (2003). "A Distributed Approach for Fault Detection and Diagnosis based on Time Petri Nets". *Proc. of CESA*. Lille, France, July 2003.

Koh J, F. DiCesare(1991). "Transformation Methods for Generalized Petri Nets and Their Applications to Flexible Manufacturing Systems". IEEE Transactions on SMC. vol. 21, no. 6, pp. 1512-1522. 1991

Lampson B.W. Reliable Messages (1993). In S. Mullender, (ed), "Distributed Systems: Architecture and Implementation", chapter 10. Addison-Wesley, 1993.

Lefebvre D. and C. Delherm (2007). "Diagnosis of DES with Petri Net models". IEEE Transactions on Automation Science and Engineering, vol. 4, no. 1, pp. 114-118, 2007.

Pencolé Y. (2004) "Diagnosability analysis of distributed discrete event systems". *Proc. of the 15th International Workshop on Principles of Diagnosis*. Carcassonne, France. June 2004.

Qiu W. and R. Kumar (2005). "Decentralized Diagnosis of Event-Driven Systems for Safely Reacting to Failures". IFAC World Congress, Prague, July 2005.

Ramírez-Treviño A., I. Rivera-Rangel and E. López-Mellado (2003). "Observability of Discrete Event Systems Modeled by Interpreted Petri Nets". *IEEE Transactions on Robotics and Automation*, vol. 19, no. 4, pp. 557-565. August 2003.

Ramírez-Treviño A., E. Ruiz Beltrán, I. Rivera-Rangel, and E. López-Mellado (2004). A. Ramírez-Treviño, E. Ruiz Beltrán, I. Rivera-Rangel, E. López-Mellado. "Diagnosability of Discrete Event Systems. A Petri Net Based Approach". Proc. of the IEEE International Conference on Robotic and Automation. pp. 541-546, April 2004.

Ramírez-Treviño A., E. Ruiz Beltrán, I. Rivera-Rangel, E. López-Mellado (2007). "On-line Fault Diagnosis of Discrete Event Systems. A Petri Net Based Approach". *IEEE Transactions on Automation Science and Engineering*. Vo1. 4-1, pp. 31-39. January 2007.

Sampath M., R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis (1995). "Diagnosability of discrete event systems", IEEE Transactions on Automatic Control, vol 40, no. 9, pp. 1555-1575, 1995.

Sampath M., R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis (1996). "Failure Diagnosis Usign Discrete-Event Models", IEEE Transactions on Control System Technology, vol. 4, no.2, pp. 105-124. 1996.

Advances in Robotics, Automation and Control

Edited by Jesus Aramburo and Antonio Ramirez Trevino

ISBN 978-953-7619-16-9
Hard cover, 472 pages
Publisher InTech
Published online 01, October, 2008
Published in print edition October, 2008

The book presents an excellent overview of the recent developments in the different areas of Robotics, Automation and Control. Through its 24 chapters, this book presents topics related to control and robot design; it also introduces new mathematical tools and techniques devoted to improve the system modeling and control. An important point is the use of rational agents and heuristic techniques to cope with the computational complexity required for controlling complex systems. Through this book, we also find navigation and vision algorithms, automatic handwritten comprehension and speech recognition systems that will be included in the next generation of productive systems developed by man.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Jesús Arámburo-Lizárraga, Antonio Ramírez-Treviño, Ernesto López-Mellado and Elvia Ruiz-Beltrán (2008). Fault Diagnosis in Discrete Event Systems Using Interpreted Petri Nets, Advances in Robotics, Automation and Control, Jesus Aramburo and Antonio Ramirez Trevino (Ed.), ISBN: 978-953-7619-16-9, InTech, Available from:
http://www.intechopen.com/books/advances_in_robotics_automation_and_control/fault_diagnosis_in_discrete_event_systems_using_interpreted_petri_nets

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821