44

TIMED-EVENT-STATE-BASED DIAGNOSER FOR MANUFACTURING SYSTEMS

M. Sayed-Mouchaweh, A. Philippot, V. Carré-Ménétrier, B. Riera
Université de Reims, CReSTIC - LAM
Moulin de la Housse B.P. 1039
51687 REIMS Cedex 2, FRANCE
Moamar.sayed-mouchaweh@univ-reims.fr

This paper proposes an adapted diagnoser for manufacturing systems. This diagnoser combines event and state based models to infer the fault's occurrence using event sequences and state conditions characterized by sensor's readings and commands issued by the controller. Furthermore, this diagnoser uses expectation functions to capture the inherent temporal dynamics of the system represented by time delays between correlated events.

1. INTRODUCTION

Fault diagnosis is defined as the operation of detecting and isolating faults. Manufacturing systems are an example of Discrete Event Systems (DES). Their behavior is generally based on two characteristics: all information regarding their operation is determined from the event sequencing and timing of events and their initial state may not be known.

The majority of DES fault diagnosis methods are based on a finite-state automaton. Some examples of the use of automata can be found in (Philippot, 2005), (Sampath, 1994), (Tripakis, 2001) and the references therein. This model accounts for the normal and failed behaviors of the process.

Since not every process is diagnosable, a notion of diagnosability must be defined (Lin, 1994), (Sampath, 1994), (Zad, 2003), (Mouchaweh, 2005) to determine if a process is diagnosable according to a certain set of observable events and pre-defined partitions of faults. It defines a diagnoser that must be able to infer the fault's occurrence after both the occurrence of the fault and the initialisation of the diagnoser and within a finite delay. This notion is formalized differently according to whether the fault model is event-based or state-based ones.

In (Sampath, 1994) a diagnosability notion using an event-based model is defined. All information relevant to the diagnosis is captured in the event set of the model. A process model is diagnosable if and only if any pair of faulty/non-faulty behaviors can be distinguished by the diagnoser using their projections to observable behaviors. The diagnoser, defined by this notion, can handle the actuator and sensor

faults. However, the diagnoser and the process model must be initiated at the same time to allow the process model and diagnoser to response simultaneously to events. This initialization is hard to obtain in manufacturing systems since their initial state may not be known. To enhance the diagnosability, the above framework is extended to dense-time automata (Tripakis, 2001). This extension is useful since it allows diagnosers to base their decisions not only on the sequences of observed events, but also on the time occurrence of these events.

To find a remedy to the initialization problem, a diagnosability notion using a state-based model is proposed in (Lin, 1994). In this notion, since the process states describe the conditions of its components, diagnosing a fault can be seen as the identification in which state or set of states the process belongs to. In (Zad, 2003), a diagnoser using a state-based model is proposed. However, the diagnosis is limited to the case of actuator faults. While manufacturing systems use many sensors entailing the necessity of diagnosing their faults.

In (Pandalai, 2000) Pandalai *et al.* use template models for monitoring DES, specifically manufacturing systems. Template models are based on the notion of expected event sequencing and timing relationships. However, these models do not allow the analysis of diagnosability properties based on a diagnosability notion, as the case for event or state based models.

This paper presents a diagnosability notion that defines a diagnoser combining event and state based models. Furthermore this diagnoser uses expectation functions in order to take into account the inherent temporal dynamics of processes. Consequently this diagnoser is adapted to diagnose manufacturing systems.

The paper is structured as following: firstly, the proposed diagnosability notion is introduced. Secondly, the construction of the diagnoser, defined by this notion, is explained using an illustration example of manufacturing systems. Finally, a conclusion and the perspectives of our future work end this paper.

2. TIMED-EVENT-STATE-BASED DIAGNOSABILITY NOTION

In this paper, we consider the problem of diagnosing DES, specifically manufacturing systems with discrete sensors and actuators.

2.1 Notations and definitions

2.1.1 Process components model

Let G and its corresponding prefixed closed language, $L = L(G)$, be the process model. This model accounts for normal and failed behaviors of the process. $G = (\Sigma, Q, Y, \delta, h, q_0)$ is a Moore automaton. Σ is the set of finite events, Q is the set of states, Y is the output space, $\delta : \Sigma \times Q \rightarrow Q$ is the state transition function. $\delta(\sigma, q)$ gives the set of possible next states if σ occurs at q. $h: Q \rightarrow Y$ is the output function. $h(q)$ is the observed output at q. q_0 is the initial state.

Balemi *et al.* (Balemi, 1993) define controllable events $\Sigma_c \subseteq \Sigma$ as the controller's outputs sent to actuators, and uncontrollable events $\Sigma_u \subseteq \Sigma$ as the controller's inputs

coming from sensors. ($\Sigma_o = \Sigma_c \cup \Sigma_u$) $\subset \Sigma$ is the set of observable events.

We use the Boolean DES (BDES) model, introduced in (Wang, 2000), to construct *G*. This modelling was initially used for the supervisory control of DES. We develop it to realize the diagnosis of DES. Each state of *G* is represented by an output vector, h_j, considered as a Boolean vector whose components are Boolean variables characterizing the state variables. Let *n* denote the number of state variables of *G*, the output vector, h_j, of each state, q_j, can be defined as :

$$\forall q_j \in Q,\ h(q_j) = h_j = (h_{j1},...,h_{jp},...,h_{jn}),\ h_{jp} \in \{0,1\}, 1 \le j \le 2^n,\ h_j \in Y \subseteq \mathrm{B}^n$$

A transition from one state to another is defined as a change of the variable state value from 0 to 1, or from 1 to 0. Thus each transition produces events characterized by either rising, $\uparrow\alpha$, or falling edges, $\downarrow\alpha$.

To describe the effect of the occurrence of an event, $\alpha \in \Sigma_o$, a displacement vector, E_α, is used. It is defined as a Boolean vector, $E_\alpha = (e_{\alpha 1},..., e_{\alpha p},..., e_{\alpha n})$, in B^n. If $e_{\alpha p} = 1$, then the value of p^{th} state variable, h_{jp}, will be set or reset when α occurs. While if $e_{\alpha p} = 0$, the value of p^{th} state variable, h_{jp}, will remain unchanged when α occurs. Consequently we can write the transition function as:

$$\forall q_i, q_j \in Q : q_j = \delta(\alpha, q_i) \Rightarrow h_j = h_i \oplus E_\alpha \qquad (1)$$

Similarly, we can define the displacement vectors for the other events. The set of all the displacement vectors of all the events provides the displacement matrix *E*.

For each event, $\alpha \in \Sigma_o$, an enable condition, $en_\alpha(q_i) \in \{0,1\}$, is defined in order to indicate if this event can occur at the state q_i, $en_\alpha(q_i) = 1$, or not, $en_\alpha(q_i) = 0$. Consequently, (1) can be re-written as :

$$\forall q_i, q_j \in Q : q_j = \delta(\alpha, q_i) \Rightarrow h_j = h_i \oplus (E_\alpha . en_\alpha(q_i)) \qquad (2)$$

2.1.2 Constrained-process model

Let $S = (\Sigma, Q_S, Y, \delta_S, h, q_0)$ denote the constrained-process model, characterized as a Moore automaton. It defines the desired behavior of the process which is represented by the prefixed closed specification language, $K = L(S) \subseteq L(G)$. *S* and its language, *K*, are constructed by experts. The set of states, Q_S , is included in *Q*.

The user provides the automaton *S* representing the desired behavior that the process should follow. We represent *S* as a BDES model. To obtain the transition function, δ_S, the enablement conditions for all the process events, $\forall\alpha \in \Sigma_o$, must satisfy all the specifications, *K*, representing the desired behaviour:

$$\forall \alpha \in \Sigma_0, \forall q_i, q_j \in Q_S : q_j = \delta_S(\alpha, q_i) \Rightarrow en_\alpha(q_i) = 1, h_j = h_i \oplus (E_\alpha . en_\alpha(q_i)) .$$

2.1.3 Definitions and conditions of Diagnosability

Let $F = \{F_1, F_2,..., F_r\}$ define the set of fault modes to be diagnosed. Each fault mode corresponds to some kind of faults in a component (sensor, actuator) or a set of such faults. We will take the case of simple fault for which a fault mode corresponds to a simple fault, $F_i = \{f_i\}$. Additionally, we assume at most one of the fault modes may occur at a time. Let Ψ_{f_i} define the set of all the event sequences ending by the fault f_i. Thus $\Psi_f = \bigcup_{i=1}^{r} (\Psi_{f_i})$ denotes the set of all the event

sequences ending by a fault from F. Consequently $\Psi_f \subseteq (L - K)$, i.e., all the faulty sequences ending by a fault of F are considered as violation of the specification language K. An observation mask is defined as : P: $\Sigma^* \cup \{\varepsilon\} \rightarrow \Sigma_o^*$, where Σ^* is the set of all event sequences of the language $L(G)$ and Σ_o^* is the set of all observable sequences. The inverse mask is defined as: $P_L^{-1}(u) = \{s \in L : P(s) = u\}$. The set of faulty states is defined as $S_f : \bigcup_{i=1}^{r} (S_{f_i})$ where S_{f_i} is the set of states reached by the occurrence of the fault f_i. Let H_{f_i} denote the set of all state output vectors of the faulty state partition S_{f_i}, then the output partition H_{f_i} can be defined as: $\forall q' \in S_{f_i}, h' = h(q') \Rightarrow h' \in H_{f_i}$. Since the state output vectors are used to diagnose fault's occurrence in state-based model, then two states belonging to different faulty state partitions must have different output vectors.

We define the set of fault labels $\Lambda_f = \{f_1,...,f_r\}$ in order to indicate the occurrence of a fault belonging to one of fault partitions $F = \{\{f_1\}, \{f_2\},...,\{f_r\}\}$. In adding the normal label N, which indicate the non presence of any fault, we can obtain the set of all the labels used by the diagnoser : $\Lambda = \{N\} \cup \Lambda_f$.

2.1.4 Events timing delays modelling

The majority of sensors and actuators in manufacturing systems produces constrained events since state changes are usually effected by a predictable flow of materials (Pandalai, 2000). Therefore, a temporal model centered on the notion of expected event sequencing and timing relationships can be used. This expectation function is constructed for observable events and it describes the next events that should occur and the relative time periods in which they are expected.

In this paper, we define an expectation function for each controllable event, $\beta \in \Sigma_c$, in order to predict uncontrollable but observable consequent events within a pre-defined time periods. These pre-defined time periods are determined by experts according to the process dynamic and to the desired behaviour S. If $u = \beta u_1 \alpha$ is an observable sequence starting by a controllable event β, and ending by the consequent event α and the observable events $u_1\alpha$ occur at the states q_i, …,q_j, then each expectation function $EF(u)$ is created when the event β occurs. This function has the following form: $EF(u) = (\{\{q_i\},..., \{q_j\}\}, \alpha, [t1, t2], \{\{f_i\},..., \{f_j\}\})$, $\forall f_i \in \Lambda_f$. This expectation means that when β has occurred, the event α should occur at any instant between t_1 and t_2. If it is the case then the expectation function is satisfied and it will provide the value 0. If the event α has occurred before t_1 or after t_2 then the expectation function is not satisfied and it will provide the value 1. If this non satisfaction occurs at the state q_i then this expectation function provides the fault label, $l = \{f_i\}$, to indicate that the cause of this non satisfaction is the occurrence of the fault f_i. Similarly the non satisfaction of the expected function at the state q_j indicates the occurrence of the fault f_j and it provides the fault label $l = \{f_j\}$. The expectation function is deleted when it is satisfied.

2.2 Timed-Event-state-based diagnosability notion formulation

The fault diagnosis problem is to diagnose unambiguously the occurrence of the fault f_i entailing a faulty behavior belonging to $(L - K) \cap \Psi_{f_i}$, by the diagnoser within a bounded delay. Consequently the timed-event-state-based diagnosability notion is defined as follows: a process model G with its language L, and a specification language K is diagnosable according to the observation mask P and the fault partitions F iff:

$$\left.\begin{array}{l}\exists k \in \mathrm{N}, \forall i \in \{1,2,...,r\}, \forall st \in (L-K) \cap \Psi_{f_i}, \\ |t| \geq k, \forall u \in P^{-1}P(st) \cap (L-K)\end{array}\right] \Rightarrow u \in (L-K) \cap \Psi_{f_i}$$

$$\forall q \in Q, q' = \delta(st, q), q'' = \delta(u, q), h' = h(q'), h'' = h(q'') \Rightarrow h', h'' \in H_{f_i} \quad (3)$$

$$\exists z \in \{1,2,...,m\} \Rightarrow EF_z(P(st)) = 1 \wedge l = \{f_i\}$$

The satisfaction of (3) means that the occurrence of the fault f_i is diagnosable by one or more of the following three ways :

- If the faulty event sequence, s, ending by the fault f_i is distinguishable by the diagnoser after the execution of $k = |t|$ transitions where t is a continuation of s. Then, any other event sequence, u, belonging to $(L - K)$ and producing the same observable event sequence as st, $P(u) = P(st)$, should contain in it the fault f_i.
- If the state q' is reached by an event sequence containing the fault f_i and possessing an output vector $h' = h(q')$ belonging to the output partition H_{f_i}, then any other state reached by any other event sequence containing the same fault, f_i, should possess an output vector $h'' = h(q'')$ belonging to the same output partition H_{f_i}.
- There is at least one expectation function, defining a temporal constraint between the occurrence of the observable events $P(st)$ not satisfied due to the occurrence of the fault f_i. This expectation function should provide the fault label $l = \{f_i\}$ as the cause of this non satisfaction.

2.3 Timed-Event-state-based diagnoser definition

The diagnoser D defined by (3) is considered as a Moore automaton : $D = (\Sigma_o \cup \{e_{EF1},..., e_{EFm}\}, Q_D, Y, \delta_D, h, q_{0D})$. The set of events, $(e_{EF1},..., e_{EFm})$, corresponds to the non satisfaction of expectation functions indicating a fault. Each diagnoser state, $q_{iD} \in Q_D$, is of the form: $q_{iD} = (h_i, L_i)$, where h_i is the output vector of the diagnoser states q_{iD}, characterized by sensor's readings and commands issued by the controller. L_i is a subset of the set, Λ, of faults labels with the normal label. We can obtain $(2^{|\Lambda|}-1)$ possible subsets of labels for L_i. If $L_i = \{N\}$ or $\{f_j\}$ then the diagnoser, when it reaches the state q_{iD}, can decide with certainty the non presence of fault or the occurrence of the fault f_j. If L_i contains the label N and any other fault label then the diagnoser, at the state q_{iD}, cannot decide whether a fault has occurred or the system is in normal function, i.e., ambiguity case. The transition function δ_D is based on (1) and is defined as, $\delta_D : \Sigma_o \cup \{e_{EF1},..., e_{EFm}\} x Q_D \rightarrow Q_D$. (1) calculates the output vectors without verifying the enablement conditions. Thus all the output vectors of

faulty states reached by a fault, characterized by the occurrence of an event not authorized at this state, can be calculated.

To construct the diagnoser, each diagnoser state must be defined. This state must be reached by an observable event belonging to $\Sigma_o \cup \{e_{EF1},..., e_{EFm}\}$. The construction of this diagnoser will be explained in detail using the illustration example of the next subsection.

3. ILLUSTRATION EXAMPLE

To explain the construction of the timed-event-state-based diagnoser, let take the example of a wagon moving from an initial position, state *A*, measured by a sensor *a*, towards a terminal position, state *B*, measured by another sensor *b*, passing by the state *A-B* indicating that the wagon is located somewhere between *A* and *B*. This movement of two directions (right and left) is realized by two commands: left, *L*, and right, *R* as it is depicted in Figure 1.

The following hypotheses are verified for this example :

- There is one part (i.e. one wagon),
- Accepted response time is defined for the actuator by the designer,
- The wagon inertia is null,
- The actuator does not fail during operation, i.e., if it does fail, the fault occurs at the start of operation,
- The operating conditions of the process initial state are normal,
- The occurrence of any fault can be expected only after the activation of a command by the controller.

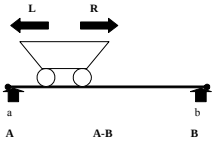
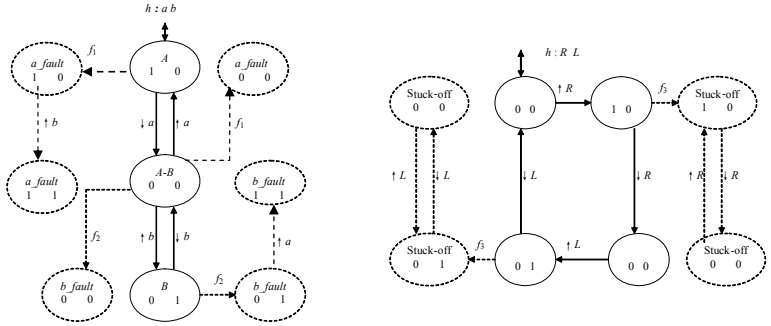


Figure 1 – Example of a wagon with two directions of movement

3.1 Component Boolean DES models

The wagon example consists of two components: the change of the wagon location measured by the sensors *a* and *b* and the wagon motor behavior. Since each BDES model must account for the normal and failed behaviors, the fault partitions to be diagnosed must be defined. Let $F = \{\{f_1\}, \{f_2\}, \{f_3\}\}$ be the set of faults to be diagnosed. f_1, f_2, and f_3 indicate, respectively, fault in sensor *a*, fault in sensor *b* and the wagon motor is stuck-off in one of the two senses right or left. The Figure 2 shows the BDES models for the change of process location and the behaviour of the wagon motor due to the commands *R* and *L*. Four Boolean state variables *a*, *b*, *R* and *L* are used to describe the overall wagon behaviour. State variables *a* and *b* are true when the wagon is located, respectively, in the position *A* or *B*. The value of *R* or *L* is 1 when they are enabled and 0 when they are disabled. If the fault f_1 occurs at the state *A*, then the model will transit to the state *a_fault* with the state variables (*a b*) = (1 0). When the wagon arrives at the position *B*, the state *a_fault* will be characterized by the state variables (*a b*) = (1 1). Similarly if *R* is enabled then the

BDES motor behaviour model will transit to the state characterized by (R L) = (1 0). If the motor is stuck-off at this instant, then the model will transit to the state *stuck_off* with the same output vector (R L) = (1 0). The same reasoning can be applied for the occurrence of the other faults. The sets of observable and controllable events are: $\Sigma_o = \{\uparrow R, \downarrow R, \uparrow L, \downarrow L, \uparrow a, \downarrow a, \uparrow b, \downarrow b\}$ and $\Sigma_c = \{\uparrow R, \downarrow R, \uparrow L, \downarrow L\}$.



a) b)

Figure 2 – Boolean DES models for the change of wagon location, a), and the wagon motor behavior, b)

3.2 Constrained-process model construction

The constrained-process model, S, for the wagon example is depicted in the Figure 3. It corresponds to the desired behavior provided by an expert. In BDES modeling, this desired behavior can be described using two tables; the first explains the enablement conditions for the occurrence of each event, Table 1, and the second is the displacement matrix E, Table 2. We can notice that the only event allowed to take place in the initial state, characterized by the output vector $h_1 = (abRL)$=(1000), is the enablement of the controllable event $\uparrow R$ since its enablement condition, $en_{\uparrow R}$ (q_1) = 1, is satisfied at this state and the enablement conditions for all the other events are false. The output vector of the next state can be calculated using (2): $h_2 = h_1 \oplus (E_{\uparrow R}.en_{\uparrow R}(q_1)) = (1000) \oplus ((0010).1) = (1010)$. Similarly we can calculate output vectors for next states after the occurrence of each authorized event.

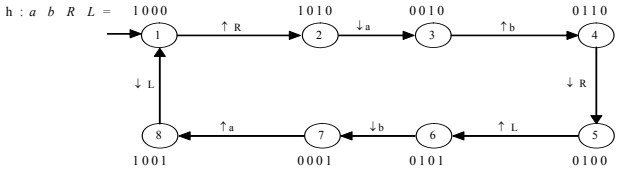


Figure 3 – Constrained-process model, S, for the wagon example

Table 1 – Enablement conditions for the events of the wagon example

Event σ	Enable condition en_σ	Event σ	Enable condition en_σ	Event σ	Enable condition en_σ	Event σ	Enable condition en_σ
$\uparrow a$	$\bar{a}.\bar{b}.\bar{R}.L$	$\uparrow b$	$\bar{a}.\bar{b}.R.\bar{L}$	$\uparrow R$	$a.\bar{b}.\bar{R}.\bar{L}$	$\uparrow L$	$\bar{a}.b.\bar{R}.\bar{L}$
$\downarrow a$	$a.\bar{b}.R.\bar{L}$	$\downarrow b$	$\bar{a}.b.\bar{R}.L$	$\downarrow R$	$\bar{a}.b.R.\bar{L}$	$\downarrow L$	$a.\bar{b}.\bar{R}.L$

Table 2 – Displacement matrix E of the wagon example

State variable	$\uparrow a$	$\downarrow a$	$\uparrow b$	$\downarrow b$	$\uparrow R$	$\downarrow R$	$\uparrow L$	$\downarrow L$
a	1	1	0	0	0	0	0	0
b	0	0	1	1	0	0	0	0
R	0	0	0	0	1	1	0	0
L	0	0	0	0	0	0	1	1

3.3 Expectation functions construction

The faults entailing the actuator to be stuck-off can be diagnosed using the principal of watchdogs to watch the actuator response times. We use expectation functions to model the actuator response times which can be obtained by learning, and/or by technical documentation. For the wagon example, we define 2 expectation functions, one for each command enablement: $EF(\uparrow R \downarrow a \uparrow b)$, $EF(\uparrow L \downarrow b \uparrow a)$.

To construct these expectation functions, we use the desired behavior, S, provided by the user, see Figure 3. The enablement of R, entails the event $\downarrow a$ at the state q_2 and the event $\uparrow b$ at the state q_3. We expect $\uparrow b$ to occur within the time period [3, 5] according to the process dynamic. If this event does not occur at q_2 then the wagon motor has not responded since $\downarrow a$ did not occur. Thus the non satisfaction of the expectation function at this state indicates the occurrence of the fault f_3, i.e., $l = \{f_3\}$. If $\downarrow a$ has occurred, then S will transit to the state q_3. If $\uparrow b$ has not occurred then the non satisfaction of the expectation function provides the label $l = \{f_2\}$ to indicate that the sensor b is faulty since the wagon has responded, because the event $\downarrow a$ has occurred. While if $\uparrow b$ has occurred too early, then the model will transit to the state q_4 and the expectation function will not be satisfied at this state and will provide the fault label $l = \{f_2\}$. Consequently the expectation function can be written as following :

$EF(\uparrow R \downarrow a \uparrow b) = (\{\{q_2\},\{q_3\},\{q_4\}\}, \uparrow b, [3, 5], \{\{f_3\},\{f_2\},\{f_2\}\})$. Similarly the expectation function for the enablement of the command L can be written as following : $EF(\uparrow L \downarrow b \uparrow a) = (\{\{q_6\},\{q_7\},\{q_8\}\}, \uparrow a, [3, 5], \{\{f_3\},\{f_1\},\{f_1\}\})$.

3.4 Timed-Event-state-based diagnoser construction

The diagnoser D for the wagon example is depicted in Figure 4. It contains, besides the states of the desired behavior model S, all the faulty states reached by the occurrence of a fault belonging to F. Each one of these faulty states is reached due to the non satisfaction of the enablement condition of an event or of an expectation function. This diagnoser is constructed using the following steps :

1. We start from the first diagnoser state, q_{1D}, corresponding to the process initial state, which is characterized by the state vector $h_1 = 1000$. At this state we suppose that the diagnoser is in normal state $L_1 = \{N\}$.
2. All the enablement conditions for all the wagon controllable events will be tested in order to find the authorized event to occur at this state. We can find that the activation of R is the only event authorized to take place since its enablement condition, $en_{\uparrow R}(q_{1D}) = a.\overline{b}.\overline{R}.\overline{L}$, is equal to 1, (see Table 1).
3. The occurrence of $\uparrow R$ transits the diagnoser to the second diagnoser state,

q_{2D}. The output vector for this state can be calculated using (1): $h_2 = (h_1 = 1000) \oplus (E_{\uparrow R} = 0010) = (1010)$. Since $en_{\uparrow R}(q_{1D}) = 1$, then this state corresponds to a state of the desired behaviour *S*.

4. After each command enablement, all the fault labels and the label *N* will be included in the label of this state : $L_2 = \{f_1, f_2, f_3, N\}$ since after each command enablement, any fault belonging to *F* can occur. Then, all the enablement conditions of the possible observable events will be tested. Here there are three possibilities: either the event $\downarrow a$, or $\uparrow b$ or the event $e_{EF(\uparrow R \downarrow a \uparrow b)}$ corresponding to the non satisfaction of the expectation function $EF(\uparrow R \downarrow a \uparrow b)$. The event $\downarrow a$ is authorized, $en_{\downarrow a}(q_{2D}) = a.\bar{b}.R.\bar{L} = 1$ and it transits the diagnoser to the state q_{3D}. The output vector for this state can be calculated using (1) :
$h_3 = (h_2 = 1010) \oplus (E_{\downarrow a} = 1000) = (0010)$. The occurrence of $\uparrow b$ conducts *D* to the state q_{9D}. This event is not authorized at q_{2D} , $en_{\uparrow b}(q_{2D}) = \bar{a}.\bar{b}.R.\bar{L} = \bar{1}.\bar{0}.1.\bar{0} = 0$ and the only reason of this non enablement is the variable state of the sensor *a*. Thus the label of this state contains only the label f_1, i.e., $L_9 = \{f_1\}$. The output vector for this state is: $h_9 = (h_2 = 1010) \oplus (E_{\uparrow b} = 0100) = (1110)$. If $EF(\uparrow R \downarrow a \uparrow b)$ is not satisfied at q_{2D}, then the event $e_{EF(\uparrow R \downarrow a \uparrow b)}$ will occur to indicate the occurrence of the fault f_3. Thus this event will transit the diagnoser to the state, q_{11D}, with $h_{11} = h_2$ and $L_{11} = \{f_3\}$. Consequently, the state corresponding to the one with the authorized event, q_{3D}, will possess the label $L_3 = L_2 - L_{11} - L_9 = \{N, f_2\}$.
5. Similarly, the other diagnoser states can be calculated to obtain, at the end, the diagnoser of the Figure 4.

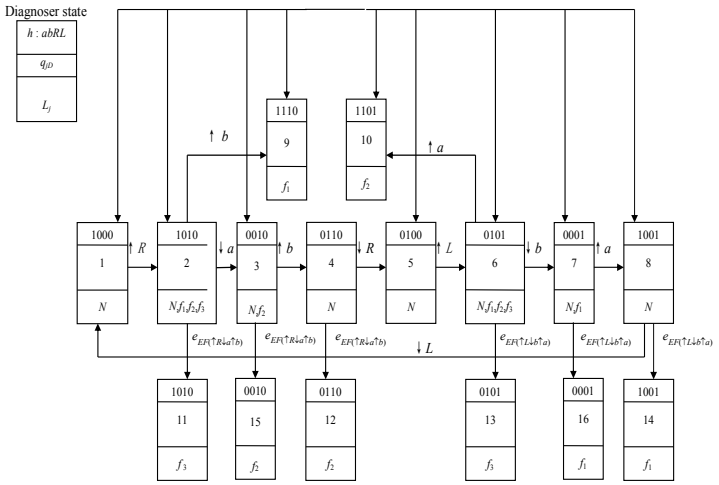


Figure 4 – Timed-event-state-based diagnoser for the wagon example

The diagnoser can be initiated at any state distinguished by its output. While if

the diagnoser was initiated at any state distinguished by an event, the diagnoser cannot diagnose a past occurrence of a fault. As an example, if the fault f_3 has occurred before the diagnoser initialisation, then the diagnoser cannot distinguish between q_{2D} and q_{11D} since they have the same output vectors : $h_2 = h_{11}$. The state 2 has the labels $L_2 = \{N, f_1, f_2, f_3\}$ which means an ambiguity and thus the diagnoser cannot decide if a fault has occurred or not.

4. CONCLUSION

In this paper, a timed-event-state-based diagnoser is proposed for manufacturing systems. The goal of this combination is to find a remedy to the problem of synchronisation between the process and its diagnoser. This problem can be often seen in manufacturing systems. To enhance the diagnosability and to capture the inherent temporal dynamics of the process, expectation functions are used. They model the temporal information about actuator's minimal and maximal response times.

Since manufacturing systems are modular, we are developing a distributed diagnosis module to perform the diagnosis. This module uses the timed-event-state-based diagnoser, proposed in this paper, as a local diagnoser in a distributed structure. In addition, we are developing the diagnosis module to relax the hypothesis of unique source or part in order to be adapted to manufacturing system.

5. REFERENCES

1. Balemi S, Hoffmann GJ, Gyugyi P, Wong-Toi H, Franklin GF. Supervisory control of a rapid thermal multiprocessor, IEEE Transactions on Automatic Control, vol. 38, n°7, pp. 1040-105, 1993.
2. Lin F. Diagnosability of Discrete Event Systems and its Applications, In *Discrete Event Dynamic Systems4*, Kluwer Academic Publishers, Boston, USA. 1994.
3. Pandalai D, Holloway LEN. Template Languages for Fault Monitoring of Timed Discrete Event Processes, In *IEEE Transactions On Automatic Control 45(5),* 2000.
4. Philippot A, Sayed Mouchaweh M, Carré-Ménétrier V. Multi-models approach for the diagnosis of Discrete Events Systems, In *IMACS'05*, *International conference on Modelling, Analyse and Control of Dynamic Systems*, Paris-France, 2005.
5. Sampath M, Sungupta R, Lafortune S, Sinnamohideen K, Teneketzis D. Diagnosability of discrete event systems, In *11th Int. Conf. Analysis Optimization of Systems: Discrete Event Systems*, Sophia-Antipolis, France, 1994.
6. Sayed Mouchaweh M, Philippot A, Carré-Ménétrier V. Detectability and Diagnosability of Discrete Event Systems: Application on manufacturing systems, 2nd ICINCO/IFAC International Conference Informatics in Control, Automation and Robotic, 14-17 September 2005, Barcelona, Spain.
7. Tripakis S. Fault Diagnosis for Timed Automata, VERIMAG (www-verimag.imag.fr). 2001.
8. Wang Y. *Supervisory Control of Boolean Discrete-Event Systems*, Thesis of Master of Applied Sciences, University of Toronto, Canada, 2000.
9. Zad SH, Kwong RH, Wonham WM. Fault Diagnosis in Discrete Event Systems: Framework and model reduction, *IEEE Transactions On Automatic Control 48(7).* 2003.