ALGORITHME D'INTERPRETATION D'UNE BASE DE SIGNATURES TEMPORELLES CAUSALES POUR LE DIAGNOSTIC EN LIGNE DES SYSTEMES A EVENEMENTS DISCRETS

Ramla Saddem, Armand Toguyéni, Tagina Moncef

9e Conférence Internationale de Modélisation, Optimisation et SIMulation - MOSIM'12
06 au 08 Juin 2012 - Bordeaux - France
« Performance, interopérabilité et sécurité pour le développement durable »

ALGORITHME D'INTERPRETATION D'UNE BASE DE SIGNATURES TEMPORELLES CAUSALES POUR LE DIAGNOSTIC EN LIGNE DES SYSTEMES A EVENEMENTS DISCRETS

R. SADDEM, A.K.A. TOGUYENI

LAGIS / Ecole Centrale de Lille
BP 48, Cité Scientifique
59651 Villeneuve d'Ascq - France
ramla.saddem@ec-lille.fr, armand.toguyeni@ec-lille.fr

M. TAGINA

LI3 / ENSI
Campus Universitaire de la Manouba
2010 Manouba - Tunisie
moncef.tagina@ensi.rnu.tn

RESUME : Le diagnostic est devenu une fonction essentielle de la commande sure des systèmes notamment dans le cadre de la surveillance en ligne. Dans cette étude, nous nous intéressons au diagnostic des systèmes à événements discrets (SED). Nous proposons une méthode appelée Signatures Temporelles Causales (STC) qui est basée sur l'ordonnancement partiel des événements caractéristiques du fonctionnement du système. Nous montrons que cette méthode est proche de formalismes comme celui des chroniques mais en diffère notamment au niveau de l'algorithme d'interprétation. Le codage des STC est généralement basé sur des connaissances expertes, pouvant conduire à un surdiagnostic. Dans cette étude nous proposons une approche pour la vérification de la cohérence d'une base de STC et nous proposons un algorithme correct d'interprétation des événements en entrée.

MOTS-CLES : *diagnostic, Signatures Temporelles Causales, surdiagnostic, cohérence, événements, chroniques.*

1 INTRODUCTION

Les systèmes industriels automatisés nécessitent d'être surveillés en ligne afin d'éviter des situations critiques. Notre étude concerne notamment les systèmes à événements discrets (SED) et plus particulièrement le diagnostic de leurs défaillances. Nous proposons les Signatures Temporelles Causales ou STC (Toguyeni *et al.*, 1991 ; Toguyéni *et al.*, 1997), un formalisme permettant d'interpréter des événements pour diagnostiquer des SED comme les systèmes manufacturiers.

Les STC sont des règles construites à partir de deux catégories d'événements : des événements caractérisant le contexte courant (ordre de la partie commande ou compte-rendu de capteurs), et des événements appelés symptômes et caractérisant des défaillances (Toguyeni *et al.*, 1991). Chaque STC consiste en un ordonnancement partiel d'un ensemble d'événements avec l'expression de contraintes temporelles spécifiques à chaque défaillance.

Ces dernières années, plusieurs auteurs ont proposé des méthodes basées sur le codage hors ligne de modèles d'événements partiellement ordonné. Ainsi, un concept assez proche a été proposé par le Laboratoire d'Analyse et d'Architecture des Systèmes (LAAS), au début des années 90, les chroniques. Comme les STC, une chronique peut être définie comme un ensemble d'événements partiellement ordonnés et contraints par des relations temporelles (Dousson *et al.*, 1993). Au cours des dix dernières années, ce formalisme a été utilisé et/ou étendu par différents auteurs, particulièrement pour le diagnostic des défaillances (Boufaied *et al.*, 2002). Il a également été exploité par France Télécom pour le diagnostic de pannes dans les réseaux de télécommunications (Cordier et Dousson, 2000) (Guerraz et Dousson, 2004) ou pour le diagnostic des défaillances d'un service web (Cordier *et al.*, 2007). Une récente étude propose leur interprétation à l'aide de la technique des automates dupliqués, ou l'écriture des chroniques en réseau de Petri colorés (Bertrand, 2009).

Dans (Holloway et Pandalai, 2000), les auteurs proposent un formalisme similaire, appelé « template language ». Cependant, ce langage de « templates » est utilisé pour indiquer le bon fonctionnement d'un système.

Le problème commun des approches orientées diagnostic est que la connaissance codée est généralement une connaissance experte. Dans le meilleur des cas, elle peut être dérivée à partir de connaissances comme les arbres de défaillance (Guerraz et Dousson, 2004). Toutefois, le problème est que chaque STC ou chronique est construite séparément sans tenir compte des autres règles. Aussi, il demeure nécessaire de vérifier la cohérence d'un ensemble de règles pour assurer complétude et déterminisme dans l'hypothèse de défaillances multiples.

La contribution de cette étude est double. Elle propose d'abord une méthode basée sur la technique du model checking pour vérifier la cohérence d'une base de STC. Ensuite, elle propose un algorithme d'interprétation basé sur le concept de monde qui garantit la correction du diagnostic.

L'article est structuré comme suit. Dans le paragraphe 2, nous faisons une rapide présentation de l'écriture des STC. Dans le paragraphe 3, nous présentons la problé-

matique de la non cohérence d'une base de STC et les risques de surdiagnostic qui en découle lorsqu'on utilise une technique d'interprétation comme celle des automates dupliqués. Dans le paragraphe 4, nous proposons une méthode de vérification de la cohérence d'un ensemble de STC basée sur la technique du modèle checking. Cela nécessitera de s'appuyer sur un formalisme adapté comme celui des automates temporisés. Une des difficultés de la méthode proposée est l'explosion combinatoire des modèles. Nous, montrons dans le 5ème paragraphe comment éviter cet écueil, en l'appliquant à un exemple de système manufacturier. Enfin, dans le 6ème paragraphe, nous proposons un nouvel algorithme d'interprétation basé sur le concept de monde.

2 LES SIGNATURES TEMPORELLES CAUSALES

2.1 Définition

Une Signature Temporelle Causale (STC) est un sous-ensemble d'événements observables partiellement ordonnés qui caractérise un comportement défaillant d'un système. Ces événements sont contraints par un ensemble de contraintes temporelles portant sur leurs occurrences. En pratique, une STC est la description d'un motif temporel définissant un ordre partiel, sur des événements décrits par leur type et leur date d'occurrence. Les relations entre les événements peuvent être logiques (conjonction) ou temporelles (séquence, absence, …).

Ainsi chaque situation anormale peut être décrite par une ou plusieurs STC, les événements pouvant être normaux (ordre de commande, compte-rendu d'un capteur) ou pouvant correspondre à des symptômes (Toguyeni *et al.*, 1991). Les contraintes portent sur leur date d'occurrence. Le diagnostic à l'aide des STC consiste donc à interpréter en ligne l'occurrence d'événements afin d'instancier les motifs qui seront reconnus. Une STC est reconnue lorsque tous ses événements se produisent en respectant leurs contraintes. Ceci détermine si le système est normal ou défaillant.

2.2 Syntaxe générale

Afin de faciliter l'usage des STC, nous avons proposé une syntaxe permettant une écriture simplifiée. Mais comme nous le montrerons dans la section 4, nous pouvons utiliser des formalismes comme les automates temporisés voire les réseaux de Petri T-temporels (Saddem et *al.*, 2010).

Une STC est une règle composée d'une partie « conditions » et d'une partie « conséquences ». La partie « conditions » est composée de un ou plusieurs triplets. Soit (X, Y, ct) un triplet. « X » et « Y » sont deux événements avec « X » l'événement de référence et « Y » l'événement contraint, attendu par rapport à « X ». «ct» est une contrainte temporelle qui permet de modéliser une date, un délai ou une durée. S'il n'y a pas de contrainte temporelle entre « X » et « Y », « ct » devient « nct », ce qui signifie aucune contrainte de temps. Soit la STC décrite par l'équation (1). Cette STC se compose de six triplets.

(In, A, nct) * (In, C, nct) * (A, B, ct1) * (A, D, ct2) * (C, D, ct3) * !(D, E, ct4) => G (1)

« A », « B », « C », « D », « E » sont des événements ; « In » est l'événement toujours occurrent, que nous introduisons de manière théorique pour définir la référence temporelle des événements qui ne sont pas contraints ; « G » est une défaillance ou un événement inobservable qui est déduit ; « cti » est la i^{éme} contrainte temporelle ; « nct » signifie pas de contrainte temporelle ; (In, A, nct) signifie que « A » est un événement sans contrainte. Le « ! » est un opérateur de négation. !(D, E, ct4) signifie que « E » ne doit pas survenir après « D » tant que la contrainte temporelle « ct4 » est vraie. « * » est un opérateur qui sépare chaque triplet et qui signifie « et à un autre moment ». Ainsi l'opérateur « * » doit être utilisé comme un « et » qui lie temporellement la reconnaissance de chaque triplet.

Dans cette étude, nous supposons que chaque événement est pris en compte qu'une seule fois dans une STC. Cela implique que dans l'équation (1), l'événement « D » dans le triplet !(D, E, ct4) est le même événement que dans les triplets (A, D, ct2) et (C, D, ct3). Dans le cas général, pour prendre en compte des occurrences distinctes du même événement et rester dans le cadre de cette hypothèse, il suffit de nommer différemment les différents événements liés à une même source (par exemple des événements liés à un même capteur). Par exemple, si nous voulons modéliser deux occurrences distinctes de « D », il suffirait de les nommer « D1 » et « D2 » dans la STC.

Remarque : Les contraintes temporelles correspondent à un temps relatif entre l'événement de référence et l'événement attendu.

Le sens de l'équation (1) est ainsi le suivant. Si on a l'occurrence de l'événement « A » (respectivement, l'occurrence de l'événement « C »), ensuite si on a les occurrences de l'événement « B » satisfaisant la contrainte « ct1 » par rapport à « A » et l'occurrence de l'événement « D » satisfaisant la contrainte « ct2 » par rapport à « A » et la contrainte « ct3 » par rapport à « C », et si l'on n'a pas l'événement « E » après l'occurrence de « D » tant que la contrainte « ct4 » est vraie, on peut en déduire l'évènement « G ».

2.3 Expression des contraintes temporelles

Pour exprimer les contraintes temporelles, on distingue deux types de structures : les structures de points et les structures d'intervalles. Nous avons retenu les structures d'intervalles car elles permettent de modéliser des notions comme la durée et elles sont conformes à la modé-

lisation temporelle du comportement du procédé pour la détection (Toguyèni *et al.*, 1997).

Une structure d'intervalle nous permet de définir trois types de contraintes temporelles : la date, la période et la durée. La date Δt (Figure 1a) permet de modéliser de manière certaine le temps séparant deux événements. La période P (Figure 1b) permet de modéliser avec un degré d'incertitude le temps séparant l'occurrence de deux événements. La durée (Figure 1c) sert à modéliser qu'une information est vraie à partir d'une date Δt et reste vraie sur tout l'intervalle de temps associé. Dans ce dernier cas on ne peut plus vraiment parler d'événement. On caractérise soit l'état d'un signal logique sur une période donnée soit la non occurrence d'un événement sur une période.

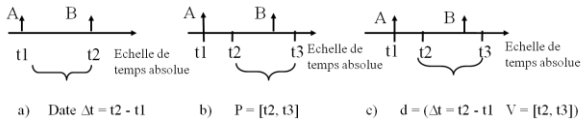

Figure 1 : Entité temporelles utilisées dans les contraintes

L'avantage de l'approche est la relative simplicité d'écriture des STC à partir par exemple d'une Analyse des Modes de Défaillances, de leurs Effets et Criticité (AMDEC) (Guerraz et Dousson, 2004). La limitation majeure dans ce cadre, est d'abord de garantir de la complétude de la connaissance anormale ainsi encodée. Le deuxième niveau de difficulté est la cohérence des contraintes temporelles relative à une STC. Comme les contraintes sont définies entre deux événements, l'ensemble des contraintes peut être incohérent. Par conséquent, nous avons emprunté au concept de chronique, l'utilisation d'un graphe temporel pour vérifier la cohérence en établissant pour chaque STC son graphe temporel minimal complet (Dousson *et al.*, 1993) (Toguyeni *et al.*, 1997). La Figure 2 présente le graphe temporel de la STC donnée par l'équation (1). C'est le graphe temporel obtenu en modélisant les contraintes de l'utilisateur. Dans un graphe temporel, chaque événement est représenté par un état et chaque contrainte temporelle par un arc orienté entre deux états. Dans la Figure 2 les contraintes sont exprimées par des intervalles de temps.

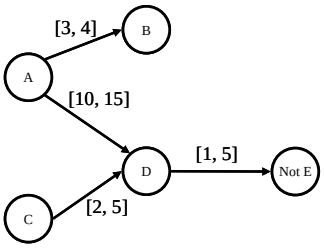

Figure 2 : Graphe temporel d'une STC d'après les contraintes de l'utilisateur

Un troisième niveau de difficulté réside dans l'interprétation des événements en entrée du système de diagnostic dans le cadre de l'hypothèse de défaillances multiples. Le problème posé dans ce cadre, est la possibilité qu'une séquence d'événements produits par plusieurs défaillances entraine un surdiagnostic, c'est à dire l'isolation d'une défaillance supplémentaire. Nous reviendrons sur cette problématique dans la partie 3.

3 PROBLEMATIQUE : INCOHERENCE RESULTANT D'UNE INTERPRETATION SELON LA TECHNIQUE DE L'AUTOMATE DUPLIQUE

Pour comprendre comment s'effectue la reconnaissance des STC, introduisons deux concepts : le motif d'une STC et l'instance d'une STC. Un motif de STC est une règle écrite et vérifiée hors ligne de manière à caractériser un fonctionnement d'un système à l'aide d'événements et d'un ensemble de contraintes cohérentes entre ces événements. Une instance de STC et la reconnaissance en ligne d'un motif de STC.

L'algorithme proposé ici est similaire à la technique de l'automate dupliqué (Bertrand, 2009). Il est basé sur les règles suivantes :

Règle 1 : Si un événement est le premier événement d'un motif de STC, créer une instance de cette STC et affecter l'événement à cette instance.

Règle 2 : Si un événement est attendu dans une instance de STC, dupliquer cette instance et l'affecter à l'instance initiale.

La règle 2 permet de tenir compte du fait que l'affectation de l'événement courant peut être une hypothèse erronée qui ne soit pas validée dans le futur. Donc chaque fois que survient un événement on fait deux hypothèses, l'affecter à l'instance et attendre une nouvelle occurrence de cet événement.

Règle 3 : Si un événement est affecté à une instance de STC, mettre à jour les fenêtres du graphe de contraintes temporelles pour déterminer le prochain événement attendu et la deadline de la STC.

La deadline de la STC est le délai maximal d'attente d'un événement.

Règle 4 : Si un événement est attendu dans une instance de STC et ne survient pas avant la deadline alors supprimer cette instance.

La règle 4 permet de supprimer les instances de STC pour lesquelles on a fait des hypothèses d'affectation incohérentes.

Règle 5 : Si tous les événements d'une STC ont été reconnus, alors intégrer les conséquences de cette STC dans l'ensemble des causes reconnues.

Afin d'illustrer cet algorithme, considérons l'ensemble de STC donné par l'équation (2).

STC1 : (In, S1, [0,+∞[)*(S1, S2, [1,3])*(S2, S3,[1,2]) => F1
STC2 : (In, S2, [0,+∞[)*(S2, S1, [1,3]) => F2 (2)
STC3 : (In, S3, [0,+∞[)*(S3, S1, [1,2]) => F3

Supposons que les symptômes détectés au fil du temps soient la séquence temporelle (S1, 0) * (S2, 2) * (S2, 3) * (S3, 4) * (S1, 5). Notons que dans cette séquence d'entrée, dans le doublet (Si, Ti), « Si » modélise un événement et « Ti » une date absolue. Afin de simplifier l'illustration de notre algorithme de reconnaissance, nous prenons comme référence du temps absolu la date de détection du premier événement (instant initial t0 = 0).

La Figure 3 illustre l'application de cet algorithme de reconnaissance à l'exemple donné par l'équation (2). Dans cette figure, nous utilisons la notation suivante « STCi-j (E, [Sc], TOi) ». STCi-j est la j^{éme} instance de STCi. E est l'ensemble ordonné des événements produits dans leur ordre d'occurrence (soient Sa le l^{er} événement, après Sb, E = Sa, Sb). [Sc] est le prochain événement attendu, et TOi est la deadline de STCi-j.

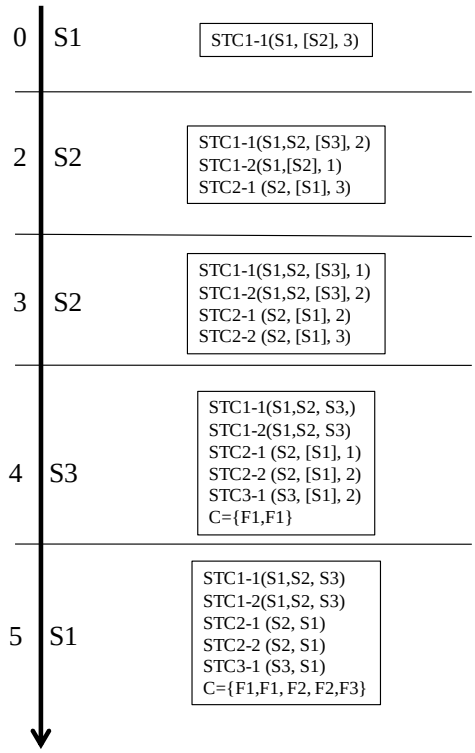

Figure 3 : Illustration de l'algorithme basé sur la technique de l'automate dupliqué

A t0 = 0, S1 se produit, Il ne peut être affecté qu'à la première STC : STC1. D'où dans la figure 3, STC1-1(S1, [S2], 3) qui signifie, S2 est attendu au plus tard 3 ut (pour valider le triplet (S1, S2, [1,3]) dans STC1).

A t1 = 2 ut, S2 se produit. S2 peut être affecté soit à STC1 (comme on attend un S2), et après, on attend S3 dans un délai de 2 ut (pour valider le triplet (S2, S3, [1,2]) dans STC1), soit ce S2 produit à t = 2 ut est affecté à STC2, et dans ce cas on attend S1 dans un délai de 3 ut (pour valider le triplet (S2, S1, [1,3]) dans STC2) et pour STC1-2, on attend encore 1 ut l'occurrence d'un S2.

De la même manière, et pour chaque événement produit, on applique les règles de l'algorithme.

L'algorithme semble avoir bien fonctionné dans la mesure où tous les symptômes ont été interprétés. Mais dans la réalité les symptômes en entrée du diagnostic ont été produits par deux causes : la cause F1 a produit les symptômes (S1, 0)*(S2, 2)*(S3, 3) et la cause F2 a produit les symptômes (S2, 3)*(S1, 5). Le diagnostic de F3 correspond donc à une situation de surdiagnostic. Ce problème est dû à un problème de cohérence de la base ou à un problème de correction de l'algorithme d'interprétation. Dans les sections suivantes, nous proposons des solutions à ces problèmes.

4 CONTROLE DE COHERENCE BASE SUR LE MODEL CHECKING

Les chroniques ou les STC sont généralement considérées comme des modèles de connaissances de surface par opposition aux modèles fondés sur les propriétés physiques d'un système. Par conséquences, certaines règles partageant les mêmes événements, dans le contexte de défaillances multiples, certaines séquences d'événements (séquences temporisées) en entrée de la tache de diagnostic peuvent conduire à un surdiagnostic. Cela est dû à l'entrelacement des événements produits par les différentes défaillances. Cet entrelacement peut caractériser une autre défaillance que celles ayant causées ces événements. C'est également un problème de cohérence.

Dans cette section, nous voulons proposer une méthode pour vérifier la cohérence d'un ensemble de STC. La présentation des STC faite dans les sections précédentes montre qu'elles correspondent à un système de transitions temporisé. Aussi dans cette étude, nous avons choisi d'utiliser le formalisme des automates temporisés pour formaliser la modélisation des STC.

4.1 Automates temporisés (AT)

Ils ont été initialement proposés par (Alur et Dill, 1994). Un automate temporisé A est un tuple (Q, q0, X, Σ, E, Inv) (Tripakis, 2002) avec : Q est un ensemble fini d'états ; q0 est l'état initial; X est un ensemble fini d'horloges avec xi ∈ ℜ+ ; Σ est un ensemble fini d'actions ; E ⊆ Q⊗C(X) ⊗Σ⊗2^X⊗Q est un ensemble fini

de transitions ; Inv $\in C(X)^Q$ associe une contrainte à chaque état.

4.2 Utilisation de l'outil KRONOS pour vérifier la cohérence d'une base de STC

KRONOS est un outil informatique développé en vue de permettre la vérification des spécifications des systèmes temps-réel modélisés par des automates temporisés (Bozga *et al.*, 1998). Nous avons choisi KRONOS pour son aptitude à extraire des séquences satisfaisant une propriété. En effet, pour vérifier la cohérence d'un ensemble de STC, l'idée consiste à vérifier s'il existe des séquences temporisées en entrée du diagnostic conduisent à des situations de surdiagnostic. Le principe de la reconnaissance de ces situations consiste à reconnaitre si certains symptômes en entrée participent à l'interprétation de plusieurs STC. Il suffit donc qu'il y ait moins de symptômes dans la séquence d'entrée que la somme des nombres de symptômes caractérisant chaque STC considérée.

La méthode de vérification comprend 4 étapes.

Etape 1 : Elle consiste à coder chaque STC dans le formalisme des automates temporisés.

Etape 2 : Elle consiste à construire un automate temporisé synthèse de tous les automates qui formalisent les différentes STC. Cet automate est construit par le produit d'automates en synchronisant tous les événements communs (produit synchrone). En effet, dans la section précédente, nous avons vu qu'une des causes de surdiagnostic est que le même événement est affecté à plusieurs STC. Pour construire l'automate de synthèse, on peut utiliser sous KRONOS la commande suivante (3) :

« kronos -out synthese.tg stci.tg stcj.tg » (3)

avec « synthese.tg » l'automate résultant, et « stci.tg » et « stcj.tg » correspondant respectivement aux automates de STCi et STCj. Pour obtenir le produit synchrone par rapport aux événements communs, on énumère dans chaque fichier de STC les événements qui sont synchronisés (Utiliser le champ « sync # » dans les fichiers « stci.tg »).

Etape 3 : Elle consiste à extraire toutes les séquences temporisées qui permettent d'atteindre les états qui correspondent à plusieurs défaillances. La commande suivante de KRONOS donne ces séquences :

« kronos –allpaths –FULLDFS –forw synthese.tg defaillances.tctl » (4)

avec « defaillances.tctl » définissant un état atteint correspondant à des défaillances multiples. En fait, ici on utilise les capacités d'analyse d'accessibilité d'états de KRONOS.

Etape 4 : Elle consiste à analyser les séquences obtenues pour voir si elles correspondent ou pas à un cas de surdiagnostic. Il y a surdiagnostic quand un symptôme est affecté à plusieurs défaillances. L'application de cette analyse à l'exemple des STC proposées par l'équation (2) nous permet de constater qu'une séquence temporisée telle que (S1 ; 0) * (S3 ; 2) * (S2 ; 3) * (S1 ; 4) * (S3 ; 5) permet de diagnostiquer les 3 défaillances sans aucune violation de contraintes. Ce résultat n'est pas correct, car il est nécessaire de détecter à des dates différentes 3 symptômes « S1 », 2 symptômes « S2 » et 2 symptômes « S3 » (donc 7 symptômes). La séquence temporisée considérée ici a seulement 5 symptômes. Cela signifie par exemple que le dernier symptôme « S1 » est utilisé pour caractériser à la fois la défaillance F2 et la défaillance F3.

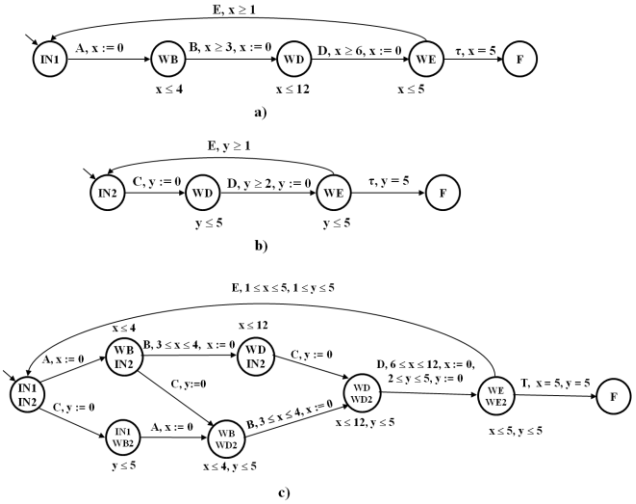

Figure 4 : Codage de STC en automates temporisés

5 APPLICATION AU DIAGNOSTIC DES DEFAILLANCES D'UN CONVOYEUR A BANDE

Nous considérons pour cette étude de cas, l'exemple donné par une partie d'un convoyeur à bande continue illustré par la Figure 5. Ce type de convoyeur comprend une bande qui est actionnée en continu par un moteur à courant continu. Dans le cadre de notre étude, le convoyeur comprend quatre postes de chargement/déchargement de produits notés « P_i » avec $i \in [1..4]$. Chaque poste est structuré de manière identique. Il comprend un capteur d'entrée noté « p_ie », un capteur de sortie noté « p_is » et une butée notée « B_i », avec i le numéro du poste. Par souci de simplicité, les postes sont tous supposés de capacité unitaire.

Dans cette étude, par souci de simplicité, nous nous limitons au diagnostic des défaillances de trois composants : le capteur « p_{1s} », l'aiguillage « A_1 », et la butée « B_1 ». La Figure 6 donne l'abstraction du comportement temporel du système pour le transfert d'un produit de P_1 vers P_2 (correspond à l'ordre « A1B2 ») ou de P_1 vers P_4 (correspond à l'ordre « A1B4 »). Par exemple le premier axe temporel indique que si la partie

commande émet l'ordre « A1B2 » ou « A1B4 », le capteur « p_{1s} » doit se déclencher entre 1 et 3 ut en raison de la détection de passage de la pièce transférée.

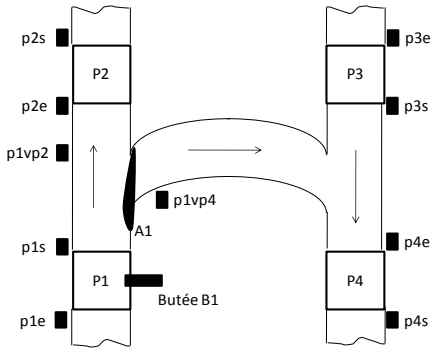

Figure 5 . Convoyeur à bande

En cas de comportement anormal d'un capteur « cp_i », il peut détecter soit un symptôme de type I (S^1_{cpi}) qui signifie que le capteur ne s'est pas déclenché dans le délai prévu ou un symptôme de type II (S^2_{cpi}) qui, au contraire, signifier un déclenchement intempestif (Toguyeni *et al.*, 1991).

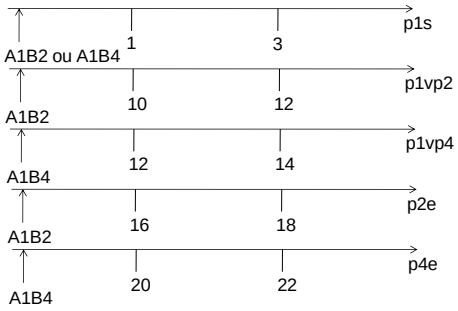

Figure 6 . Abstraction du comportement normal d'une partie du convoyeur

Nous supposons qu'un module de détection est placé en filtre entre la partie opérative et la partie commande (Figure 7).

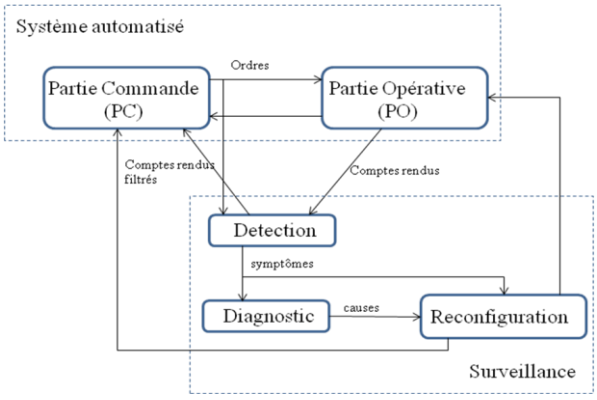

Figure 7 : Structure du système de surveillance pour le diagnostic des défaillances

Nous donnons à présent quelques STC relatives à ces défaillances :

STC1: (In, A1B2,[0,+∞[)* (A1B2, S^1_{p1s}, [3,3]) * (A1B2, p1vp2, [10,12]) * (A1B2, p2e, [16,18]) => (collage, p1s, 0)

Le sens de STC1 est le suivant. Si après la commande du transfert d'un produit de « P1 » vers « P2 », le capteur de sortie « p1s » ne se déclenche pas au bout d'un délai maximum de 3 ut, cela conduit à la détection d'un symptôme de type I (S1p1s).

En cas de déclenchement des deux autres capteurs situés sur le parcours (respectivement p1vp2 et p2e) dans les délais correspondant au comportement normal (Figure 6), on conclue que la cause de ce symptôme c'est le collage à zéro du capteur « p1s ». De la même manière, nous écrivons 5 STC supplémentaires :

STC2: (In, A1B4,[0,+∞[)*(A1B4, S1p1s, [3,3]) * (A1B4, p1vp4, [12,14]) * (AB14, p4e, [20,22]) => (collage, p1s, 0)

La STC2 est duale de la STC1. Elle sert dans le contexte d'un transfert de « P1 » vers « P4 ».

STC3: (In, A1B2,[0,+∞[)* (A1B2, p1s, [1,3]) * (A1B2, S1p1vp2, [12,12])* (A1B2, S1p2e, [18,18]) * => (blocage, A1, P1vP4)

La STC3 sert à localiser une défaillance de l'aiguillage « A1 » bloqué dans le sens des transferts de « P1 » vers « P4 ». Cette situation peut être détectée lors d'un transfert de « P1 » vers « P2 » quand les capteurs placés le long de la trajectoire ne détecte rien, donnant des symptômes de type I.

STC4: (In, A1B4,[0,+∞[)* (A1B4, p1s, [1,3]) * (A1B4, S1p1vp4, [14,14]) * (A1B4, S1p4e, [22,22]) => (blocage, A1, P1vP2)

La STC4 est duale de la STC3.

STC5: (In, A1B2,[0,+∞[)* (A1B2, S1p1s, [3,3]) * (A1B2, S1p1vp2, [12,12])) * (A1B2, S1p2e, [18,18]) => (blocage, B1, sortie)

La STC5 sert à diagnostiquer la défaillance de la butée « B1», bloquée en position sortie, en cas de demande de transfert de « P1 » vers « P2 ». Dans ce cas, aucun des capteurs le long de trajectoire ne détecte de pièce ce qui conduit à la génération en séquence de 3 symptôme de type I.

STC6: (In, A1B4,[0,+∞[)* (A1B4, S1p1s, [3,3]) * (A1B4, S1p1vp4, [14,14])) * (A1B4, S1p4e, [22,22]) => (blocage, B1, exit)

STC6 est duale de STC5 pour le contexte d'un transfert de « P1 » vers « P4 ».

En fait, les 6 règles peuvent être groupées en deux groupes de règles : le groupe des règles commençant par l'événement « A1B2 » (STC1, STC3, STC5) et le groupe des règles commençant par l'événement « A1B4 ». Puisque « A1B2 » et « A1B4 » sont deux ordres opposés, les deux classes de règles ne peuvent pas être interprétées simultanément. Par conséquent, la vérification de la cohérence peut être effectuée classe par classe. Ici, nous illustrons notre méthode avec la classe initiée par « A1B2 ».

La figure 8 illustre la réécriture de l'automate temporisée correspondant. Nous effectuons d'abord, la composition parallèle de STC3 et STC5 en synchronisant les événements « A1B2 », « S1p1vp2 » et « S1p2e ». Nous effectuons ensuite la composition parallèle de l'automate obtenu avec STC1 en synchronisant les événements « A1B2 » et « S1p1s ». L'automate final est représenté par la Figure 9.

Comme développé dans la partie 3, on peut repérer les séquences temporisées qui pourraient conduire un surdiagnostic avec un algorithme incorrect comme celui basé sur la duplication d'automate.

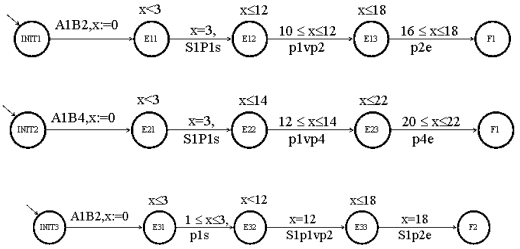

Figure 8 : Ré-écriture des STC en AT

6 ALGORITHME D'INTERPRETATION BASE SUR LA NOTION DE MONDE

Les algorithmes de reconnaissance des chroniques sont basés sur la technique de duplication d'automates (Bertrand, 2009). L'algorithme de reconnaissance de STC proposé ici est basé sur le concept de monde. Dans notre approche, un monde représente un ensemble d'hypothèses cohérentes d'affectations d'événements reçus par la tâche de diagnostic à une et une seule instance de STC.

En pratique, un monde « W_i » (avec i l'indice de ce monde) est caractérisé par : « Evt_i » (l'ensemble des événements reçus et affectés), « L_i » (l'ensemble des STC instanciées), « $Causes_i$ » (l'ensemble des causes inférées), « NE_i » (le prochain événement attendu), « MAX_TO_i » (le time out donnée par « NE_i » calculé en prenant le minimum des fenêtres temporelles obtenues en propageant la date d'occurrence de chaque événement le précédent compte tenu des contraintes temporelles les reliant à « NE_i »), « MIN_TO_i » (le maximum des minimums des contraintes temporelles propagées à partir des événements précédents).

Notre algorithme est basé sur les règles suivantes :

Règle 1 : Hypothèse du monde Fermé
Cette règle signifie que si un événement n'est pas explicitement reçu par la tâche de diagnostic, il est supposé être inexistant (donc la négation est vraie).

Un événement reçu en entrée est supposé vrai jusqu'à ce qu'il ait contribué à la reconnaissance complète d'une STC.

Règle 2 : Une seule affectation par symptôme
Si un symptôme survient, il est affecté à une et une seule STC dans un monde.

Cette règle est cohérente avec l'hypothèse classique des systèmes asynchrones qui considère que deux événements externes qui sont totalement indépendants ne peuvent pas se produire en même temps. En d'autres termes, pour nous, deux défaillances indépendantes ne produisent pas simultanément les mêmes symptômes. Dans l'écriture de nos STC, nous supposons les défaillances indépendantes.

Règle 3 : Duplication des mondes
Si plusieurs STC sont candidates pour l'affectation d'un symptôme, dupliquer le monde en cours en autant de mondes que d'hypothèses d'affectation cohérente. En fait, chaque possibilité correspond à une hypothèse d'affectation qui est cohérente avec les hypothèses précédentes qui ont été faites auparavant pour les symptômes reçus qui ont été affectés à une seule STC dans ce monde. Cette règle est une conséquence de la règle 2.

Règle 4: Suppression du monde fondé sur un ensemble d'hypothèses incohérentes
Si un événement n'est affectable à aucune des STC du monde considéré (il n'était pas prévu ou il n'est pas compris entre le MIN_TO_i et le MAX_TO_i), ou si un événement attendu n'est pas survenu à MAX_TO_i, cela signifie que toutes les affectations antérieures sont incohérentes. Alors le monde considéré est supprimé.

Cette règle est fondée sur l'hypothèse que la base des STC est complète.

Règle 5 : Fusion des deux mondes
Soient deux mondes « W_i » et « W_j ». Si tous leurs attributs caractéristiques sont égaux (« $Causes_i$ » = « $Causes_j$ » ; « L_i » = « L_j » ; « NE_i » = « NE_j » ; « MIN_TO_i » = « MIN_TO_j » ; « MAX_TO_i » = « MAX_TO_j »), alors garder « W_i » et supprimer « W_j ».

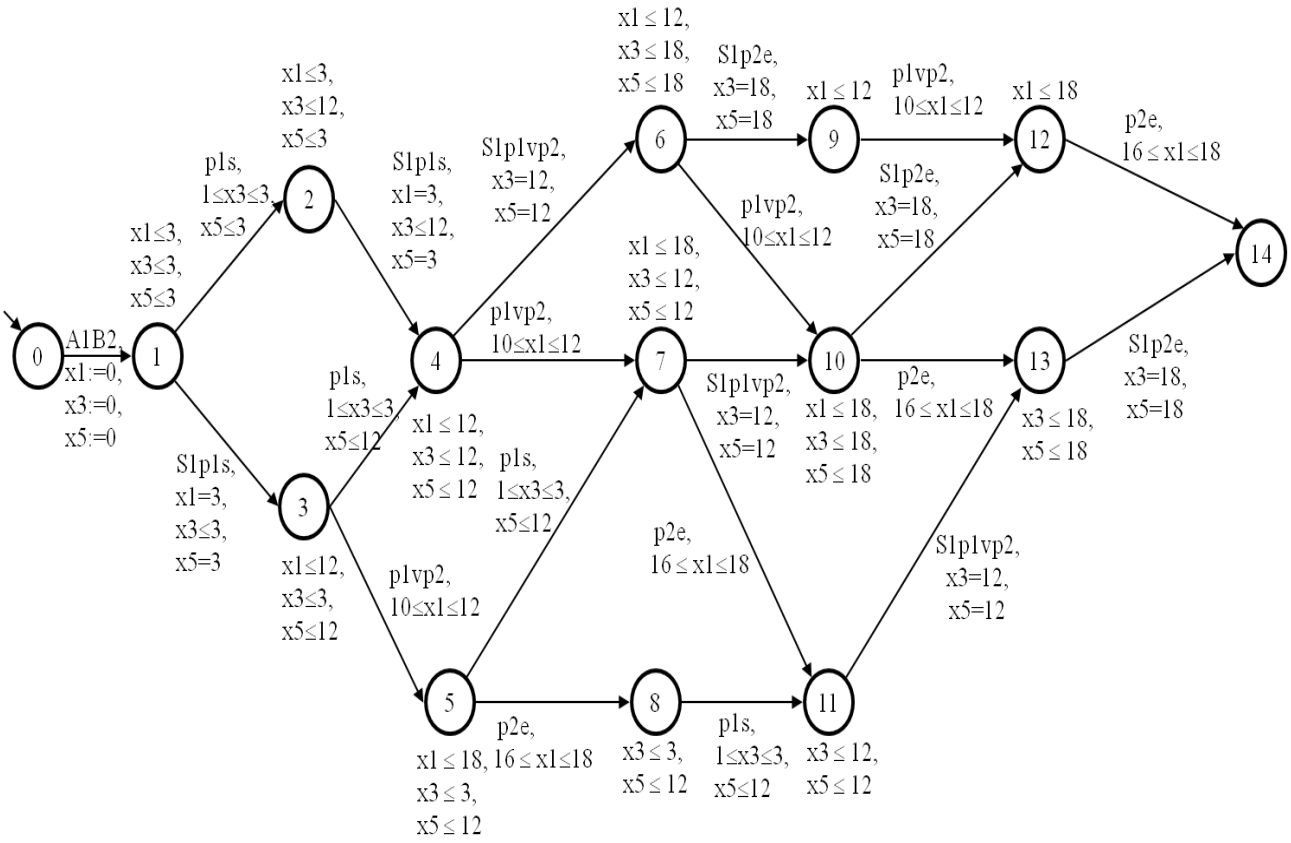

Figure 9 . Composition parallèle des AT de STC1, STC3 et STC5

Règle 6 : Utilisation des causes inférées
Si un seul monde subsiste à une date et l'ensemble de ses causes n'est pas vide, les causes identifiées correspondent au diagnostic. Si « L_i » est vide, alors réinitialiser tous les attributs du monde.

La figure 10 représente un exemple d'arbre de résolution. Cet exemple est basé sur l'ensemble de STC données par l'équation (2).

Lorsque le diagnostic reçoit le premier « S1 », ce symptôme ne peut être affecté qu'à la STC1. Mais quand « S2 » est détecté à l'instant t = 2, d'après la règle 2, STC1 et STC2 sont en concurrence. Par conséquence, le monde précédent « W1 » est dupliqué. On obtient deux mondes « W1 » et « W2 » (duplication de W1 avant instanciation de STC2) qui correspondent à deux hypothèses d'affectation (respectivement H1 et H2). Ainsi dans « W1 », « S2 » est attribué à la STC1. Donc, cette hypothèse sera confirmée si « S3 » se produit dans un délai de 2 u.t. après « S2 », soit avant la date 4 u.t. après cette affectation. L'hypothèse H2 affecte « S2 » à STC2. Par conséquent, dans « W2 », STC1 continue à attendre « S2 » pendant encore un délai de 1 u.t. et, STC2 attend « S1 » dans un délai de 3 u.t. Donc W2.MAX_TO (deadline du monde W2) est égale à 1 u.t., ce qui signifie que l'hypothèse H2 serait fausse si ce time out expire. Dans l'arbre de résolution, les mondes comme « W3 » à la date t = 4, « W2 », « W4 » et « W7 » à t = 5 et « W1 » à t = 6 sont supprimés en raison de l'expiration de leur time out. A t = 5, les deux mondes « W5 » et « W6 » sont équivalents donc ils sont fusionnés d'après la règle 5 (En pratique, l'un d'eux est supprimé pour ne conserver qu'un seul monde).

Ce sont les règles 2, 3 et 4 qui garantissent la correction de cet algorithme. En effet la règle 2 garantit qu'un symptôme n'est affecté qu'à une seule STC dans un monde. En cas de compétition d'affectation la règle 3 garantit que chacun des mondes dupliqués correspond à une hypothèse concurrente d'affectation d'un symptôme. Bien entendu, l'une des hypothèses est fausse. L'hypothèse fausse est révélée par le fait qu'un événement ne peut pas être interprété où que la deadline d'une STC du monde a été atteinte. Dans ce cas les mondes incohérents sont détruits par le règle 4 et ne subsiste que le monde correspondant aux hypothèses cohérentes.

7 CONCLUSION

Dans cette étude, nous avons montré l'intérêt de l'utilisation de méthode telle que les STC pour le diagnostic en ligne des défaillances des SED.

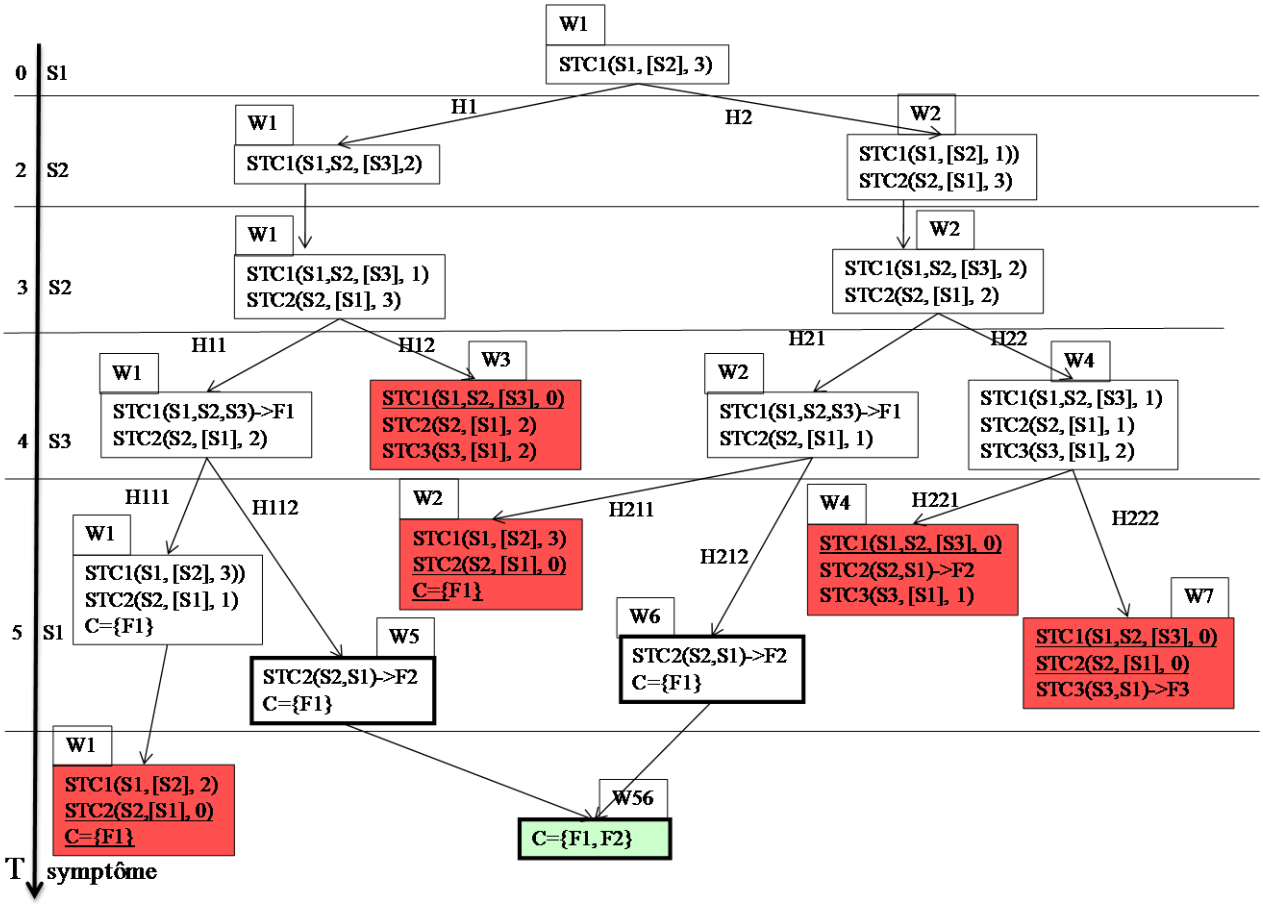

Figure 10 : Arbre de résolution de notre algorithme

Dans le paragraphe 3, nous avons montré que la technique de duplication d'automate pouvait conduire à des situations de surdiagnostic pouvant limiter la disponibilité du système de production par des arrêts non justifiés. Dans la partie 4, nous avons proposé une méthode d'analyse de la cohérence d'une base de STC de manière à identifier les possibilités de surdiagnostic. Cette méthode d'analyse permet a priori de vérifier si l'on peut interpréter une base de STC à l'aide de l'algorithme de duplication qui possède l'avantage d'être simple et peut exigeant en mémoire. Il est donc a priori bien adapté à des contextes de diagnostic de système embarqués. Dans la 6ème partie, nous avons proposé un nouvel algorithme de reconnaissance des STC qui se révèle correct même dans le cas de l'hypothèse de défaillances multiples avec entrelacement de symptômes. En effet les mondes résultants de duplications de contexte peuvent entrainer un besoin important en mémoire. Le 2ème algorithme d'interprétation est donc plus adapté à une utilisation dans le cadre d'une informatique classique, par exemple en exécution sur un PC.

Pour le futur, nous travaux concernent d'une part l'utilisation des STC dans un contexte distribué, et d'autre part rapprocher les approches STC et diagnostiqueur (Tripakis, 2002). En effet, il est clair que l'encodage des STC à partir de connaissances expertes rend impossible la possibilité de garantir la complétude de la base. L'idée est donc de les construire à partir d'une abstraction du comportement souhaité du système, complété des défaillances possibles. Même si dans ce cas, on ne peut toujours pas garantir la complétude pour l'isolation de toutes défaillances, on est au moins sûr de pouvoir toutes les détecter. Une piste dans ce cadre serait de construire des diagnostiqueurs pour surveiller le système et de les implémenter sous forme de STC.

REFERENCES

Alur R. and D. Dill, 1994. A theory of timed automata. *Theoretical Computer Science*, p. 183-225.

Bertrand O., Détection d'activités par un système de reconnaissance de chroniques et application au cas des simulations distribuées HLA, Thèse de docteur de l'université, Paris 13, juin 2009.

Boufaied A., A., Subias and M. Combacau, 2002. Chronicle modeling by Petri nets for distributed detection of process failures. *IEEE International Conference on, Systems, Man and Cybernetics*, Vol. 4, p. 6-9.

Bozga M., C. Daws, O. Maler, A. Olivero, S. Tripakis and S. Yovine, 1998. KRONOS: A Model-Checking Tool for Real-Time Systems. *The 10th International Conference on Computer Aided Verification, CAV'98*, 546-550, Vancouver, BC, Canada.

Cordier M. O. and C. Dousson, 2000. Alarm Driven Monitoring Based on Chronicles. *4th Symposium on Fault Detection, Supervision and Safety for Technical (Safeprocess)*, Budapest, Hongrie.

Cordier M. O., X. Le Guillou, S. Robin, L. Rozé, and T. Vidal, 2007. Distributed Chronicles for On-line Diagnosis of Web Services. *The 18th International Workshop on Principles of Diagnosis (DX'07)*, Nashville, p. 37-44.

Dousson C., P. Gaborit and M. Ghallab, 1993. Situation Recognition : Representation and Algorithms. *International Joint Conference on Artificial Intelligenc*e (*IJCAI'93)*, Chambéry, France, p. 166-172.

Guerraz B. and C. Dousson, 2004. Chronicles construction starting from the fault model of the system to diagnose. *The 15th International Workshop on Principles of Diagnosis (DX'04)*, p. 51–56.

Holloway L. and N. Pandalai, 2000. Template Languages for Fault Monitoring of Timed Discrete Event Processes. *IEEE Transactions on Automatic Control*.

Saddem R., A. K. A. Toguyeni, M. Tagina, 2010. Consistency's checking of chronicles' set Using Time Petri Nets. *The 18th Mediterranean Conference on Control & Automation (MED)*, Marrakech, Moroc, p. 1520–1525.

Toguyeni A. K. A., E. CRaye and J. C. Gentina, 1991. A method of Temporal Analysis to perform on-line Diagnosis. Industrial Electronics Society (IECON'91), *the context of Flexible Manufacturing System*, California, USA.

Toguyeni A. K. A., E. CRaye and J. C. Gentina, 1997. Time and reasoning for on-line Diagnosis of failures in Flexible Manufacturing Systems. *The 15th IMACS world congress on scientific computation, modeling, and applied mathematics*, vol. 6., Berlin, Germany, p. 709-714.

Tripakis S., 2002. Fault Diagnosis for Timed Automata. *The 7th International Symposium on Formal Techniques in Real-Time and Fault Tolerant Systems (FTRTFT'02)*, Springer, p. 205–224.