Identification des états incompatibles lors d'un changement de mode

Gregory FARAUT, Laurent PIÉTRAC, Eric NIEL
Laboratoire Ampère
INSA-Lyon, Bât. St-Exupery, 20 av Albert Einstein, 69621 Villeurbanne, France
Prenom.Nom@insa-lyon.fr

Résumé—**Le changement de mode est un des problèmes que nous pouvons avoir lors de la conception de systèmes à événements discrets (SED). En effet, même basé sur des spécifications triviales, il est difficile de prouver formellement que les modèles de chaque mode et leurs interactions soient sans fautes. Cet article montre que l'utilisation de la théorie de contrôle par supervision est un outil intéressant pour détecter les incompatibilités entre spécifications grâce à une séparation entre le modèle du procédé et le modèle des spécifications. Nous utiliserons un cas simple pour présenter une méthode introduisant de la flexibilité dans les spécifications propres à chaque mode. Cette méthode permet d'ajuster, ou de modifier les incompatibilités entre spécifications afin d'obtenir un changement de mode cohérent.**

Mots-clés—**SED, SCT, Gestion de modes, Automates**

I. INTRODUCTION

Un système comportant plusieurs modes montre potentiellement différents comportements, parfois conflictuels. Ainsi, un comportement indésirable peut typiquement mener vers de graves conséquences comme une défaillance du dit système, voir infliger une blessure physique. La décomposition modale lors de la conception d'un système est une vue régulièrement utilisée dans les entreprises, et de nombreux travaux sur les SED ont tentés d'apporter une aide à la conception d'un système à travers une gestion de mode [1], [2]. Certaines études se concentrent directement sur l'utilisation des automates pour représenter des modes [3], [4], mais très peu prennent en compte les problèmes de commutation entre ceux-ci et de validation qui en découlent. Bien que la conception se concentrait sur le fonctionnement interne de chaque mode grâce à l'indépendance des modèles de mode, le temps de validation, notamment lors de leur interconnexion, est resté et reste encore beaucoup trop grand pour certifier qu'un système possède bien les propriétés – de sécurité notamment – attendues. Des méthodes récentes de validation reposent sur une approche formelle permettant une validation en amont de ces spécifications. Cependant, celles-ci restent confinées à une validation indépendante des modèles et ne prennent que difficilement en compte la reconfiguration du système.

La reconfiguration est due à toute sorte de besoins, qui nécessitent un changement de configuration du système. Suivant la gravité de la panne, la tâche doit être interrompue et redémarrée ensuite, si la mission en cours n'est plus réalisable. Les propriétés telles que l'atteignabilité, l'observabilité ou la contrôlabilité (applicable dans la théorie de contrôle par supervision conventionnelle [5], [6]) peuvent être considérées d'un point de vue qualitatif. L'atteignabilité est peut-être la plus importante des propriétés à valider en relation avec le phénomène de commutation ; elle sous-entend, pour être possible, qu'au moins une trace doive exister et qu'elle puisse être manipulée afin de pouvoir commuter d'un mode à un autre. Dans cet article, nous nous attacherons à détecter la compatibilité entre états des modes (appelée cohérence) et sera déterminée à partir des états de leurs composants communs (utilisés dans différents modes) lors d'une reconfiguration.

Notre approche est basée sur les travaux de [7] et [8], qui permettent d'étudier le comportement interne de chaque mode indépendamment. L'hypothèse principale est basée sur la propriété qu'un seul mode n'est actif à la fois dans notre système. Nous proposons un processus de conception permettant une validation formelle des propriétés, internes et externes, sur les modèles. Les états compatibles sont ceux pour lesquels il existe une cohérence d'états sur les composants communs entre les deux modes liés par la reconfiguration.

Cet article se décompose ainsi : la section 2 introduit les concepts de la théorie de contrôle par supervision (SCT), sur lesquels sont basés nos travaux. La section 3 formalise l'ensemble de la conception intra et intermodale et l'illustre à travers un exemple traité tout du long de cette section. Enfin, nous concluons en indiquant les perspectives de travaux possibles.

II. THÉORIE DE CONTRÔLE PAR SUPERVISION (SCT)

La théorie de Ramadge et Wonham [5] concerne l'étude du contrôle des systèmes à événements discrets. Cette théorie est basée sur la séparation entre le modèle représentant tout ce que le système **peut faire** (le procédé, appelé "G"), le modèle de ce que le système **doit faire, ou pas** (la vivacité et la sécurité du système, représentant les spécifications et appelé "E"), le modèle de ce que le système **fait** (le procédé contrôlé par un superviseur, représenté par "S/G") et le modèle de ce que le système **devrait faire** (le langage désiré, appelé "K").

Le procédé [9] est un générateur d'événements, représenté par un automate G et défini par $G = (Q, \Sigma, \delta, q_0, Q_m)$ où Q est l'ensemble des états finis, Σ est l'ensemble d'événements, $\delta : Q \times \Sigma \rightarrow Q$ est la fonction de transition partielle, q_0 est l'état initial et $Q_m \subseteq Q$ est l'ensemble des états marqués. Les états existent pour une période de temps (durée), jusqu'au moment où un événement intervient instantanément et spontanément. Des exemples d'états, pour une machine, sont "inactif", "en cours de fonctionnement", "en panne", "en cours de réparation". Des exemples d'événements sont "machine commence à travailler", "travail accompli" ou "lancement de la réparation".

Pour un alphabet Σ, nous avons la répartition $\Sigma = \Sigma_c \cup \Sigma_{uc}$ où les sous-ensembles disjoints Σ_c et Σ_{uc} correspondent respectivement aux événements contrôlables et non contrôlables. Les langages associés à G sont le langage $L(G) = \{s \in \Sigma^* : \delta(q_0, s)!\}$ qui représente l'ensemble de toutes les trajectoires possibles, autrement dit l'ensemble des comportements possibles du système, et le langage marqué $L_m(G) = \{s \in \Sigma^* : \delta(q_0, s) \in Q_m\}$ qui représente le sous-ensemble des tra-

jectoires menant à un des états marqués. Ces états permettent de modéliser la fin de tâches, les états à atteindre ou encore les états dans lesquels le système peut s'arrêter.

Pour un procédé G, il y a deux façons possible de construire la loi de commande qui respecte le cahier des charges. Dans la première méthode, nous avons au commencement le modèle du procédé G et le modèle des spécifications E, qui ont évidemment le même alphabet. Le concepteur vérifie l'existence du produit [6] de ces deux automates, résultant en un sous-ensemble du système qui respecte les spécifications. Si le langage généré par $G \times E$ est contrôlable, alors cet automate représente S/G, action du superviseur S sur le procédé G, qui est le modèle de la loi de commande à implémenter. Dans le cas où le langage n'est pas contrôlable, l'utilisation d'une fonction, appelée "suprême contrôlable sous-langage", permet de déterminer le plus grand sous-ensemble de $G \times E$ qui respecte les spécifications et qui soit contrôlable. La seconde méthode consiste, à partir du procédé G, à modéliser la loi de commande que le concepteur désire pour son système (modèle du langage désiré K), et de s'assurer qu'il existe un superviseur S tel que $L(S/G) = K$, autrement dit que le procédé G restreint par ce superviseur permet d'obtenir le modèle K de la loi de commande désirée. Dans ce cas aussi, si le superviseur S n'existe pas, il est possible de déterminer le plus grand sous-ensemble du langage qui respecte les spécifications grâce au suprême contrôlable.

III. Gestion des modes

Dans cette section, nous traitons à titre d'exemple un système, représenté sur la fig.3(a), composé de quatre machines et d'un stock (de capacité égale à 1). Le système doit pouvoir opérer même en cas de panne. C'est la raison de la présence des machines 3 et 4 qui servent à remplacer les machines 1 et 2 en cas de défaillances de celles-ci. Les modèles des machines sont donnés sur la fig.3(b) pour les machines 1 et 2, et sur la fig.3(c) pour les machines 3 et 4. Le système peut fonctionner dans 4 modes particuliers représentant les machines utilisées suivant les pannes intervenues. D'un point de vue d'une gestion de modes, notre contribution propose un processus dans lequel :

- Chaque mode est étudié indépendamment et séparément. Dans la plupart des cas, un mode est caractérisé par les composants représentant son comportant interne et par les composants représentant son comportement commutatif ;
- Les spécifications intramodales devant être satisfaites sont étudiées et appliquées sur le modèle indépendamment des autres spécifications. La SCT est appliquée dans ce cas "normalement" dans chacun des modes (nous nous limitons au cas centralisé).
- Les spécifications intermodales portent sur la gestion de modes et interviennent directement en plus des spécifications intramodales dans le mode concerné. Ceci dépend de la contrôlabilité de l'événement de commutation. Le principal problème est donc de déterminer l'état du modèle quand nous quittons un mode (appelé "mode initial") pour entrer dans un autre mode (appelé "mode final").

Les sections suivantes décrivent successivement le processus intramodale pour chacun des modes donnés dans l'exemple et la conception des procédés intermodaux. Commençons par donner les définitions des composants et modes que nous utilisons.

A. Définitions

Le système est composé de composants (quatre dans le cas de l'exemple). La dynamique de chacun est la même quelque soit le mode. Ces dynamiques comportent d'éventuels événements de faute ou de réparation. De tels événements sont utilisés pour modéliser la commutation entre modes.

Définition 1

$\mathcal{C}$ est l'ensemble des composants. Un composant est modélisé par un automate $G^{\mathcal{C}_i}$ où $G^{\mathcal{C}_i} = (Q^{\mathcal{C}_i}, \Sigma^{\mathcal{C}_i}, \delta^{\mathcal{C}_i}, q_0^{\mathcal{C}_i}, Q_m^{\mathcal{C}_i})$, avec :

- *$Q^{\mathcal{C}_i}$ est l'ensemble d'états du composant i ;*
- *$\Sigma^{\mathcal{C}_i}$ est l'ensemble des événements du composant i, incluant deux sous-ensembles :*
  - *$\Sigma^{\mathcal{C}_i} = \Sigma_c^{\mathcal{C}_i} \cup \Sigma_{uc}^{\mathcal{C}_i}$ avec $\Sigma_c^{\mathcal{C}_i} \cap \Sigma_{uc}^{\mathcal{C}_i} = \phi$;*
  - *$\Sigma^{\mathcal{C}_i} = \Sigma_{\circlearrowright}^{\mathcal{C}_i} \cup \Sigma_{\leftrightarrows}^{\mathcal{C}_i}$ avec $\Sigma_{\circlearrowright}^{\mathcal{C}_i} \cap \Sigma_{\leftrightarrows}^{\mathcal{C}_i} = \phi$;*
    - *$\Sigma_{\circlearrowright}^{\mathcal{C}_i}$ est l'ensemble des événements représentant le comportant interne du composant ;*
    - *$\Sigma_{\leftrightarrows}^{\mathcal{C}_i}$ est l'ensemble des événements de commutation impliquant un changement de mode ;*
- *$\delta^{\mathcal{C}_i}$ est la fonction de transition ;*
- *$q_0^{\mathcal{C}_i}$ est l'état initial du composant i ;*
- *$Q_m^{\mathcal{C}_i}$ est l'ensemble des états marqués du composant i ;* ♦

Définition 2

Les modes d'un système peuvent être définis de plusieurs manières. La plus commune dans l'industrie est la définition de ceux-ci dans le cahier des charges. L'ensemble des modes définis correspond aux différentes configurations que peut avoir notre système pour répondre aux demandes ou besoins particuliers. L'ensemble des modes est noté par $\mathcal{M} = \{M_1, M_2, \ldots, M_n\}$, où $n \in \mathbb{N}$ et $n \geq 1$ (par convention, nous supposons que le mode initial activé est toujours M_1). $\mathcal{C}^{M_j}$ l'ensemble des composants représentant le fonctionnement du système dans le mode M_j, où $\mathcal{C}^{M_j} = \mathcal{C}^{M_j\circlearrowright} \cup \mathcal{C}^{M_j\leftarrow} \cup \mathcal{C}^{M_j\rightarrow}$ tel que :

- *$\mathcal{C}^{M_j\circlearrowright}$ est l'ensemble des composants représentant le comportement interne du mode M_j ;*
- *$\mathcal{C}^{M_j\leftarrow}$ est l'ensemble des composants générant un événement impliquant une commutation entrante dans le mode M_j ;*
- *$\mathcal{C}^{M_j\rightarrow}$ est l'ensemble des composants générant un événement impliquant une commutation sortante du mode M_j ;*
- *$\mathcal{C}^{M_j\leftrightarrows} = \mathcal{C}^{M_j\leftarrow} \cup \mathcal{C}^{M_j\rightarrow}$ est l'ensemble des composants générant un événement de commutation ;*

Il n'existe aucune relation particulière entre $\mathcal{C}^{M_j\circlearrowright}$, $\mathcal{C}^{M_j\leftarrow}$ et $\mathcal{C}^{M_j\rightarrow}$ excepté qu'ils sont tous inclus dans $\mathcal{C}$. Ainsi, un composant peut être inclus dans :

- *$\mathcal{C}^{M_j\circlearrowright}$, ne générer aucun événement de commutation ;*
- *$\mathcal{C}^{M_j\circlearrowright}$ et dans $\mathcal{C}^{M_j\leftarrow}$. Cela signifie que ce composant est utilisé pour représenter le fonctionnement interne du mode et qu'il génère également un événement qui implique une commutation entrante dans ce mode ;*
- *$\mathcal{C}^{M_j\leftarrow}$ ou $\mathcal{C}^{M_j\rightarrow}$. Ce composant est uniquement nécessaire pour représenter le comportement commutatif du mode, et n'est pas utilisé dans le comportement interne de celui-ci.* ♦

Reprenons le système pris en exemple, fig.3(a), et considérons ses quatre modes. Le mode 'nominal' correspond au fonctionnement idéal du système avec seulement les machines 1 et 2 opérationnelles. Les modes dégradés représentent les modes possibles suivants que la machine 1 soit en panne (mode dégradé d_1), la machine 2 seule (mode dégradé d_2), ou les deux en même

temps (mode dégradé d_3). La fig.3(d) montre la décomposition en mode du système, les composants internes à chaque mode (appartenant à l'ensemble $\mathcal{C}^{M_j\circlearrowleft}$) et les événements impliquant une commutation d'un mode à l'autre.

B. Conception intramodale

Pour chaque mode M_j, le procédé $G^{M_j\circlearrowleft}$ résulte en la composition parallèle [6] des automates $G^{\mathcal{C}_i}$ des composants internes au mode et est défini sur $\Sigma^{M_j\circlearrowleft} = \bigcup_{i\in\mathcal{C}^{M_j\circlearrowleft}} \Sigma^{\mathcal{C}_i}$.

Pour chaque mode M_j, la spécification $E_k^{M_j}$ (problème 1) ou le langage désiré K^{M_j} (problème 2) sont définis sur l'alphabet $\Sigma^{M_j\circlearrowleft}$. La spécification générale $E^{M_j\circlearrowleft}$ est le produit des automates de spécifications devant être respectés dans ce mode. Après avoir conçu les n modes nécessaires, le concepteur obtient n procédés non contrôlés $G^{M_j\circlearrowleft}$, n spécifications $E^{M_j\circlearrowleft}$ et n procédés sous contrôle $S^{M_j\circlearrowleft}/G^{M_j\circlearrowleft}$ (appelé par la suite $G_{sup}^{M_j\circlearrowleft}$). Il nous reste alors à considérer la gestion des modes entre eux.

Dans l'exemple, les modèles doivent être contrôlables par rapport aux spécifications définies, telles qu'exprimées par l'automate $E^{n\circlearrowleft}$ représentant la spécification du stock dans le mode nominal. Cette spécification, montrée sur la fig.3(e), implique de construire le suprême contrôlable de L($G_{sup}^{M_j\circlearrowleft}$), car celui-ci n'est pas contrôlable.

C. Conception intermodale

À ce stade, le comportement interne des modes est complètement construit. Dans cette section, nous allons inclure le comportement externe des modes et détecter les trajectoires reliant les modes entre eux, autrement dit, détecter les trajectoires conduisant à un état où un événement de commutation peut être généré. Le processus utilisé est montré sur la fig.1.

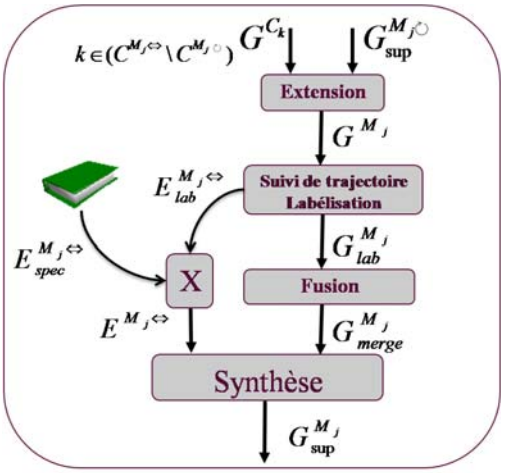

Fig. 1. Processus de la conception intermodale

Ce processus se divise en plusieurs étapes. La première est l'extension des modèles intramodaux. Cette extension sert à rajouter le comportement commutatif (des modes) non précédemment modélisé dans le processus de conception intramodale, car généré par des composants inutiles dans la représentation du comportement interne aux modes. Le suivi de trajectoire suit l'étape de l'extension. Cette étape sert à rechercher les événements de commutation correspondants entre les modes. Afin de pouvoir différencier les couples d'événements équivalents entre modes, chaque couple reçoit un sous-indice. L'ajout de ce sous-indice est ce qui est appelé "labélisation" car cela revient à changer le label des événements de commutation. Le but principal de cette labélisation est d'éviter de rendre les modèles non déterministes à cause de l'étape suivante (étape de fusion III-C.3). En effet, les modèles contiennent à ce stade plus d'informations que nécessaire, nous procédons donc à une fusion des états non significatifs dans chaque mode afin de ne garder que l'information utile. La quatrième et dernière étape est celle de l'application des spécifications intermodales par un contrôle des automates en utilisant la théorie de contrôle par supervision.

C.1 Extension des procédés sous contrôle

Le système étant représenté par différents modes $G_{sup}^{M_j\circlearrowleft}$, il manque une information aux modèles nous permettant de déterminer le comportement commutatif complet entre modes. Pour l'ajouter, nous étendons chaque modèle $G_{sup}^{M_j\circlearrowleft}$ par composition parallèle avec les composants inclus dans $\mathcal{C}^{M_j\leftrightarrows}$ mais seulement ceux n'étant pas déjà inclus dans $\mathcal{C}^{M_j\circlearrowleft}$.

Définition 3
Supposons $G_{sup}^{M_j\circlearrowleft}$ tel que :

$$G_{sup}^{M_j\circlearrowleft} = (Q_{sup}^{M_j\circlearrowleft}, \Sigma_{sup}^{M_j\circlearrowleft}, \delta_{sup}^{M_j\circlearrowleft}, q_{sup,0}^{M_j\circlearrowleft}, Q_{sup,m}^{M_j\circlearrowleft})$$

G^{M_j} est l'extension du modèle de $G_{sup}^{M_j\circlearrowleft}$ où :

$$G^{M_j} = G_{sup}^{M_j\circlearrowleft} || (||_{A\in(\mathcal{C}^{M_j\leftrightarrows}\setminus\mathcal{C}^{M_j\circlearrowleft})} A) \qquad \blacklozenge$$

La décomposition obtenue à la fin de cette étape est représentée fig.4(a), avec en exemple l'automate de la fig.4(b) représentant le comportement dans le mode nominal.

C.2 Suivi de trajectoire

La corrélation entre les modes est représentée par les événements de commutation générés par les composants et impliquant une commutation des modes qu'ils relient. Nous devons maintenant identifier quelles sont les trajectoires où un événement de commutation peut être généré pour sortir du mode initial et entrer (activer) dans le mode final. En effet, le comportement dynamique du mode initial s'arrête là où la dynamique du mode final commence. Pour identifier ces connexions entre modes, il faut prendre en compte toutes les traces menant à un état où un événement de commutation est généré dans le langage du mode initial, et projeter ces traces dans le mode final pour trouver l'état cohérent pour la commutation. Autrement dit, nous définissons $K_{M_j}(G^{M_j} \rightarrow G^{M_k})$, un langage désiré incluant toutes les traces du mode initial M_j, menant à la première occurrence d'un événement de commutation vers le mode final M_k. Les mots dans $K_{M_j}(G^{M_j} \rightarrow G^{M_k})$ ne doivent pas avoir d'événement de commutation, mais mènent tous à un état où un événement de commutation peut se produire. À partir de ces traces, l'état équivalent dans le mode final est possible à déterminer mathématiquement grâce à l'utilisation de la fonction de projection étendue, introduite par les auteurs de [7]. De cet état peut se produire l'événement de commutation généré depuis l'état du mode initial utilisé pour le calcul de la trajectoire.

Formellement, la fonction de projection étendue $P_{j,k}$ est définie comme suit :

Définition 4
Let $P_{j,k} : \Sigma_j^* \rightarrow \Sigma_k^*$ *tel que* $\forall \sigma \in \Sigma_j$ *et* $\forall s \in \Sigma_j^*$ *:*

$$P_{j,k}(\varepsilon) = \varepsilon$$
$$P_{j,k}(s\sigma) = \begin{cases} P_{j,k}(s)\sigma \text{ si } \sigma \in \Sigma_j \cap \Sigma_k \\ P_{j,k}(s) \quad \text{si } \sigma \in \Sigma_j \backslash \Sigma_k \end{cases}$$

Littéralement, cette fonction ne contraint pas les alphabets comme sur la projection naturelle, et revient à *effacer* de la chaîne s tous les événements σ qui ne sont pas inclus dans l'intersection des ensembles d'événements.

Adaptons maintenant la fonction de projection, développée dans un cas général, sur le système à quatre machines.

Procédure 1
Sur l'exemple, nous considérons les modes nominal, représenté par le modèle G^n*, et dégradé représenté par le modèle* G^{d_1}*. Nous supposons que le mode initial est* G^n*, et que le mode final est* G^{d_1}*. Le mode* d_1 *a été étendu avec le composant* $G^{\mathcal{C}_1}$ *responsable de la génération des événements* f_1 *et* r_1 *provoquant une commutation. Ces modes partagent le composant* $G^{\mathcal{C}_2}$*. L'événement* f_1 *provoque la commutation du mode initial vers le mode final, alors que l'événement* r_1 *provoque la commutation du mode final vers le mode initial. Donc,* $f_1 \in (\Sigma^n_{\rightarrow} et\ \Sigma^{d_1}_{\leftarrow})$*, et* $r_1 \in (\Sigma^n_{\leftarrow} et\ \Sigma^{d_1}_{\rightarrow})$*.*

1. Des deux modes considérés (n *et* d_1 *pour l'exemple), nous identifions toutes trajectoires les reliant. Pour cela :*

(a) Nous calculons le langage $K_n(G^n \rightarrow G^{d_1})$ *qui représente toutes les traces menant à un état* q^n *où une première occurrence d'un événement de commutation peut être généré* G^n*.*

(b) $K_n(G^n \rightarrow G^{d_1})$ *est projeté sur* $L(G^{d_1})$ *pour obtenir* $K_{d_1}(G^n \rightarrow G^{d_1})$*. Ces traces mènent vers des états* q^{d_1} *pour lesquels* $\delta(q^{d_1}, f_1)$ *est défini.*

(c) À ce niveau, il y a deux cas possibles :

i. L'événement de commutation, identifié par une trace unique, existe dans le mode initial et dans le mode final. Dans ce cas, l'événement de commutation est labélisé, dans les deux modes, en lui ajoutant un sous-indice afin d'avoir un nom spécifique qui permettra de l'identifier facilement par la suite.

ii. L'événement de commutation existe dans le mode initial, mais pas dans le mode final. Cela signifie que cette trace désactive le mode initial, car provoque la commutation, mais n'active aucun autre mode. La labélisation ne concerne alors que l'événement de commutation du mode initial. C'est ce type de traces, identifiées par un événement de commutation unique grâce à la labélisation, qui génère une erreur si nous gardons cette trace lors de l'implémentation.

(d) Nous répétons cette opération pour tous les événements de commutation identifiés par $K_n(G^n \rightarrow G^{d_1})$*.*

(e) Cette procédure est de nouveau utilisée pour identifier toutes les premières occurrences de l'événement de commutation r_1 *qui mèneront du mode* G^{d_1} *au mode* G^n*.*

2. Quand toutes les trajectoires reliant les deux modes ont été identifiées et labélisées, la réutilisation de cette procédure pour tous les autres modes reliés entre eux permet d'identifier toutes les trajectoires de commutations du système. La fonction de fusion se termine lorsque tous les premiers événements de commutations des modèles de mode ont été labélisés. ♦

Le second cas de l'étape (c) est réellement important, car c'est ce type de commutation qui donne les états incompatibles dans la gestion de mode. Cela signifie que, dans le cas où au moins une trace unique de ce type existe, le système peut tomber en panne et ne plus garantir le fonctionnement voulu, ce qui est en contradiction avec les spécifications à respecter. Garder comme information toutes les traces menant à un des états incompatibles permet de les interdire avec les spécifications $E^{M_j \leftrightarrows}_{lab}$, qui sont ajoutées aux spécifications intermodales (traitée dans la section III-C.4).

C.3 Fonction de fusion

La fonction de fusion réduit la taille des modèles en fusionnant les états non significatifs. Nous adaptons, comme précédemment, la procédure suivante à l'exemple, afin d'améliorer sa compréhension.

Procédure 2
Considérons l'automate G^n_{lab}*. Les états de* G^n_{lab} *sont nommés, par construction,* $(q^{\mathcal{C}_1}, q^{\mathcal{C}_2}, x_n)$ *où* $q^{\mathcal{C}_1} \in Q^{\mathcal{C}_1}$ *,* $q^{\mathcal{C}_2} \in Q^{\mathcal{C}_2}$ *et* $x_n \in X^{n^{\circlearrowleft}}$*. Supposons que* $Q^{\mathcal{C}_i} = N_i \cup F_i, N_i \cap F_i = \phi$*, où* N_i *représente un ensemble d'état où le composant* i *marche convenablement et, à l'inverse,* F_i *représente un ensemble d'état où le composant est en panne à cause d'un événement correspondant à une défaillance. Dans l'exemple sur le mode nominal,* $N_i = \{A_i, M_i\}$ *et* $F_i = \{F_i\}$*.*

1. Nous déterminons dans G^n_{lab}*, un ensemble d'état à fusionner* $Q^n_{mer} \subset Q^n_{lab}$*. Les états inclus dans* Q^n_{mer} *sont les états qui ne sont pas significatifs pour le mode :*
- *Non significatif pour le mode nominal correspond, par exemple, à tous les états labélisés par* F_1 *ou* F_2*, car le mode nominal n'est pas censé contenir le comportement en panne de* $G^{\mathcal{C}_1}$ *ou de* $G^{\mathcal{C}_2}$*.*
- *Non significatif pour le mode dégradé* d_1*, par exemple, correspond à tous les états qui ne sont pas labélisés par* $(F_1, q^{\mathcal{C}_2}, q^{\mathcal{C}_3})$ *et si* $q^{\mathcal{C}_2} \neq F_2$*. Car dans le mode dégradé, nous ne voulons que le comportement du système quand seul le composant* $G^{\mathcal{C}_1}$ *est en panne.*
- *De la même manière, une table de toutes les combinaisons possibles avec les états des composants pour déterminer un mode est imaginable et réalisable et ne dépend que du cahier des charges. Dans l'exemple comportant quatre modes, les états significatifs du mode nominal sont ceux de la forme* (N_a, N_b) *car représentant le fonctionnement nominal sans panne du système, les autres états étant fusionnés. Pour le mode dégradé* d_1*, seul les états labélisés par* (F_1, N_2) *sont significatifs, représentant une panne de la machine* 1 *seule. Les états labélisés par* (N_1, F_2) *pour le mode dégradé* d_2 *(machine 2 seule en panne) et enfin les états labélisés par* (F_1, F_2) *pour le mode dégradé* d_3*.*

2. Le nouvel état formé par la fusion se nomme $q^{M_j}_{id}$*.*

3. Nous supprimerons toutes les boucles de $q^{M_j}_{id}$ *qui en sortent et en rentrent.*

4. Si l'état initial $q^{M_j}_{0,lab}$ *est inclus dans* $Q^{M_j}_{mer}$*, alors* $q^{M_j}_{id}$ *est le nouvel état initial.*

5. Si un état marqué est inclus dans $Q^{M_j}_{mer}$*, alors* $q^{M_j}_{id}$ *est marqué également.* ♦

En utilisant la fig.4(c) représentant le modèle G^n_{lab} du mode nominal, il est facile d'identifier les états non significatifs. Tous ces états vont alors fusionner pour donner l'état q^n_{id}. Cet état n'est par marqué et n'est pas l'état initial de notre modèle. Ce nouveau modèle appelé G^n_{merge} est représenté sur la fig.4(d) et permet d'avoir un aperçu de la réduction de complexité réalisée grâce à la fonction de fusion.

Remarque 1
Il est bien connu que la fusion d'état dans un automate peut entrainer de l'indéterminisme [6]. C'est pour éviter cela que la labélisation réalisée dans la section III-C.2 a été utilisée précédemment. L'information ainsi gardée permet de discriminer les événements de commutation entre eux. Autrement dit, nous avons anticipé la fonction de fusion d'états en utilisant la labélisation. La labélisation est facilement réalisable grâce à la discrimination effectuée par la fonction de suivi de trajectoire. L'utilisation conjointe de la labélisation avec la fonction de fusion permet ainsi de réduire la complexité des modèles sans perdre l'information concernant la commutation et tout en évitant l'indéterminisme. ♦

À la fin de cette étape, les modèles de modes représentant leur comportement interne et leur comportement commutatif sont presque totalement construits. Ils seront appelés $G^{M_j}_{merge}$ et définis tel que :

- $Q^{M_j}_{merge} = (Q^{M_j}_{lab} \setminus Q^{M_j}_{mer}) \cup \{q^{M_j}_{id}\}$,
- $\Sigma^{M_j}_{merge} = \Sigma^{M_j}_{lab} \setminus \bigcup_{i \in (\mathcal{C}^{M_j \leftrightarrows} \setminus \mathcal{C}^{M_j \circlearrowright})} \Sigma^{C_i}_{\circlearrowright}$
- $\delta^{M_j}_{merge} = \delta^{M_j}_{lab} \setminus \delta^{M_j}_{lab}(q_a, s, q_b)$ avec $q_a, q_b \in Q^{M_j}_{mer}$
- $q^{M_j}_{merge,0} =$

$$\begin{cases} q^{M_j}_{sup,0} & si\ q^{M_j}_{sup,0} \notin Q^{M_j}_{mer} \\ q^{M_j}_{id} & si\ q^{M_j}_{sup,0} \in Q^{M_j}_{mer} \end{cases}$$

- $Q^{M_j}_{merge,m} =$

$$\begin{cases} Q^{M_j}_{sup,m} & si\ Q^{M_j}_{sup,m} \cap Q^{M_j}_{mer} = \phi \\ Q^{M_j}_{sup,m} \setminus Q^{M_j}_{mer} \cup \{q^{M_j}_{id}\} & sinon \end{cases}$$

C.4 Spécifications intermodales

Nous avons dit à la fin de l'introduction que le système ne peut opérer que dans un seul mode à la fois. Ceci afin d'éviter tous conflits entre modes. Nous avons également identifié dans la sectionIII-C.2 des états incompatibles, où un événement de commutation peut se produire, et la trace menant à ces états. Cela signifie que ces événements de commutation peuvent se produire dans le mode initial, mais n'existent dans le mode final. Ces commutations peuvent bloquer le système, ou même causer des dégâts irréversibles si elles se produisent. Ces traces sont donc à interdire.

Pour satisfaire ces spécifications, incluses dans les spécifications intermodales et appelées $E^{M_j \leftrightarrows}$, nous proposons d'utiliser comme modèle de base ceux illustrés sur la fig.2. Ils fournissent une aide pour décrire les modèles des spécifications et assurent la première spécification, c'est-à-dire que pas plus d'un mode n'est actif en même temps. De plus, Il est facile d'ajouter (ou plutôt supprimer) de ces modèles les événements de commutation à interdire grâce aux sous-indices ajoutés précédemment avec la labélisation. En résumé, l'utilisation de ces modèles permet une construction simple et directe des spécifications intermodales.

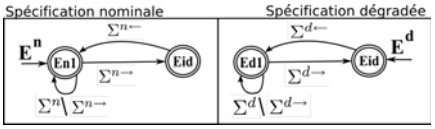

Fig. 2. Modèles de base pour construire les spécifications intermodales

Pour conclure sur l'exemple, le contrôle des modèles $G^{M_j}_{merge}$ par les spécifications de commutation $E^{M_j \leftrightarrows}$ donnent les modèles des procédés sous contrôles des modes, illustrés fig.5(b) pour le mode nominal et fig.5(d) pour le mode dégradé. Ceux-ci correspondent aux modèles de nos modes, représentant leur fonctionnement et assurant les spécifications, notamment de commutation.

IV. Conclusion

Ce papier traite de la cohérence d'états lors d'une commutation entre modes. Il expose une définition d'états incompatibles et donne une solution quand une commutation est requise, en accord avec l'hypothèse que les composants sont considérés dans plusieurs modes à la fois. L'incompatibilité a été considérée comme un état inexistant dans le mode final, impliquant une incohérence d'états liée à un ou plusieurs composants entre le mode initial et le mode final lors d'une commutation. Nous proposons une solution formelle capable de reconnaître ces incohérences et ainsi de détecter les problèmes d'atteignabilité. La gestion de modes par un suivi des configurations modifie la façon de concevoir et la contribution majeure de ce travail porte sur la formalisation d'incompatibilité lors d'un changement de configuration. Nos recherches actuelles portent sur l'ajout de stratégies de commutation, lorsqu'une incompatibilité d'états liée à un événement non contrôlable est détectée, et que le suprême contrôlable ne donne pas satisfaction vis-à-vis des spécifications exigées par le système.

V. Remerciements

Les auteurs souhaitent remercier Stéphane Lafortune pour les remarques apportées dans l'élaboration de ce travail.

Références

[1] D. Harel : Statecharts : A visual formalism for complex systems. *Science of Computer Programming*, 8(3):231–274, June 1987.

[2] F. Jahanian et A.K. Mok : Modechart : A specification language for real-time systems. *IEEE Trans. Softw. Eng.*, 20(12):933–947, 1994.

[3] F. Maraninchi et Y. Rémond : Mode-automata : a new domain-specific construct for the development of safe critical systems. *Science of Computer Programming*, 1(46):219–254, March 2003.

[4] Jean-Pierre Talpin, Christian Brunette, Thierry Gautier et Abdoulaye Gamatie : Polychronous mode automata. *In EMSOFT '06 : Proceedings of the 6th ACM & IEEE International conference on Embedded software*, pages 83–92, New York, NY, USA, 2006. ACM Press.

[5] P.J.G. Ramadge et W.M. Wonham : The control of discrete event systems. *Proceedings of the IEEE*, 77(1):81–98, Jan 1989.

[6] Christos G. Cassandras et St'ephane Lafortune : *Introduction to discrete event systems [Second Edition]*. Springer, 2007.

[7] O. Kamach, L. Piétrac et E. Niel : Multi-model approach to discrete events systems : Application to operating mode management. *Mathematics and Computers in Simulation, Elsevier*, 70(5-6):394–407, 2005.

[8] L. ; Niel E. Kamach, O. ; Pietrac : Forbidden and preforbidden states in the multi-model approach. *Computational Engineering in Systems Applications, IMACS Multiconference on*, 2:1550–1557, 4-6 Oct. 2006.

[9] W. M. Wonham : Supervisory control of discrete-event systems. ece 1636f/1637s 2006-07. course notes, departement of Electrical and Computer Engineering, Univeristy of Toronto, 2006.

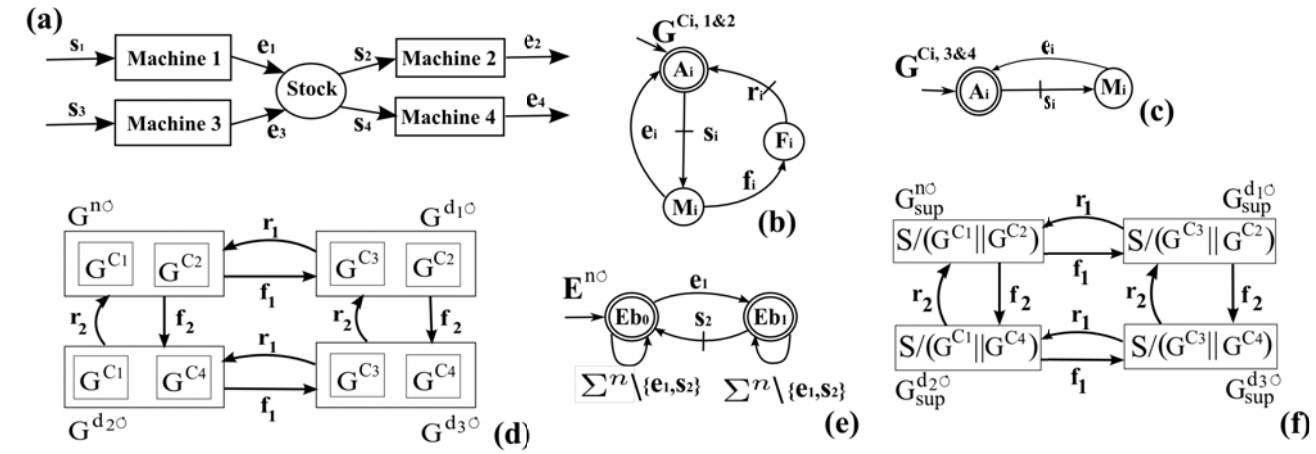

Fig. 3. Exemple d'un système : (a) système ; (b) modèle G_i des 'machines 1 et 2' ; (c) modèle G_i des 'machines 3 et 4' ; (d) décomposition modale du système ; (e) spécifications E^n concernant le stock du mode nominal ; (f) décomposition modale des procédés sous contrôle.

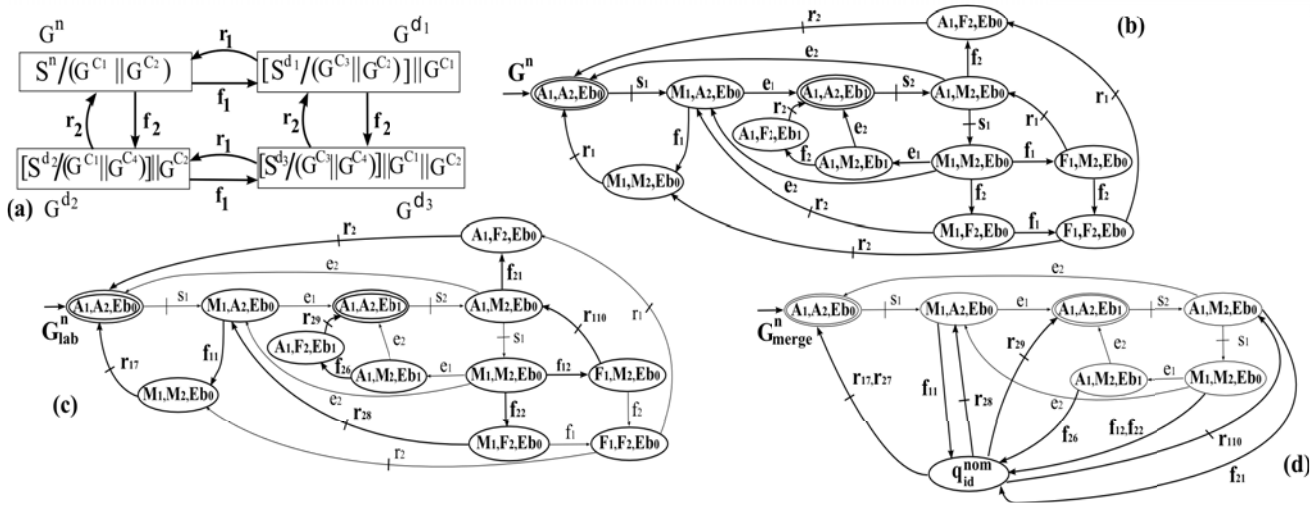

Fig. 4. Exemple d'un système : (a) Décomposition modale des procédés sous contrôle étendus ; (b) Procédé sous contrôle étendu du mode nominal ; (c) le procédé sous contrôle étendu incluant les événements de commutation labélisés ; (d) procédé sous contrôle étendu et labélisé du mode nominal et ayant subi la fusion de ses états non significatifs.

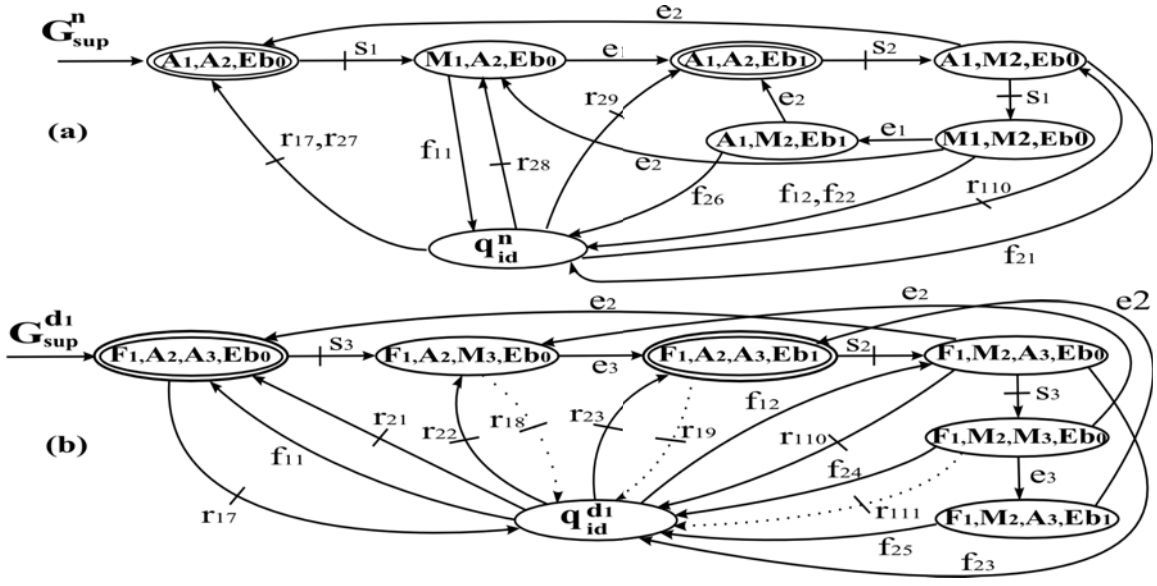

Fig. 5. Exemple d'un système : (a) procédé sous contrôle final du mode nominal ; (b) procédé sous contrôle final du mode dégradé d_1.