Aide à la conception d'architectures opérationnelles de commande de systèmes critiques par analyse d'atteignabilité

Thibault Lemattre, Bruno Denis, Jean-Marc Faure, Jean-François Pétin, Patrick Salaün

Aide à la conception d'architectures opérationnelles de commande de systèmes critiques par analyse d'atteignabilité

Thibault LEMATTRE[1,2], Bruno DENIS[1], Jean-Marc FAURE[1], Jean-François PÉTIN[3], Patrick SALAÜN[2]

[1] Laboratoire Universitaire de Recherche en Production Automatisée
Ecole Normale Supérieure de Cachan, 61 avenue du Président Wilson, 94235 Cachan Cedex, France

[2] Electricité de France, Recherche & Développement
6 quai Watier, 78400 Chatou, France

[3] Centre de Recherche en Automatique de Nancy
UMR 7039 CNRS - Nancy Université, BP 70239, 54506 Vandoeuvre les Nancy, France

{Lemattre,Denis,Faure}@lurpa.ens-cachan.fr,
Jean-Francois.Petin@cran.uhp-nancy.fr, Patrick.Salaun@edf.fr

Résumé— **Ce papier présente une méthode qui facilite la conception de l'architecture opérationnelle d'un système de commande en proposant, à partir de la connaissance des caractéristiques des fonctions que doit assurer ce système et des contrôleurs choisis pour réaliser ces fonctions, une solution d'affectation des fonctions qui satisfait des contraintes de capacités et de répartition tout en minimisant le nombre de contrôleurs. Cette méthode repose sur la vérification d'une propriété d'atteignabilité sur un réseau d'automates communicants. L'intérêt de cette proposition est illustré par le traitement d'un cas non trivial issu de la réalité industrielle.**

Mots-clés— **architectures de commande, architecture opérationnelle, automates communicants, atteignabilité, vérification formelle.**

I. INTRODUCTION

Un système de commande peut être caractérisé par trois types d'architectures :

- l'architecture fonctionnelle, ensemble de fonctions de commande interconnectées qui exprime les besoins. Dans le cas des systèmes de production d'énergie, qui constituent la cible applicative de cette étude, les fonctions de commande ont pour objectif, par exemple, de réguler la température et/ou la pression d'une chaudière, ou de contrôler la position des grappes de régulation ou la concentration en bore lorsque le processus de production est à base d'énergie nucléaire ;
- l'architecture matérielle (ou physique), constituée de contrôleurs (automates programmables industriels ou calculateurs temps réel), chargés d'exécuter les fonctions de commande et générant des signaux à destination du processus à partir des informations recueillies sur ce dernier, et d'un ou plusieurs réseaux de communication permettant les échanges de données entre les contrôleurs ainsi qu'entre les contrôleurs et le système de surveillance/supervision ;
- l'architecture opérationnelle, construite par projection de l'architecture fonctionnelle sur l'architecture matérielle [1].

Cette projection consiste à affecter chacun des éléments de l'architecture fonctionnelle à un contrôleur, tout en respectant des contraintes qui peuvent être relatives :

- aux capacités des contrôleurs en termes de charge CPU et/ou d'entrées/sorties ;
- à la répartition des fonctions, les exigences de sûreté conduisant en effet, pour les systèmes critiques, à définir des classes de fonctions telles que deux fonctions appartenant à des classes différentes peuvent ou non être regroupées dans un même contrôleur ;
- au coût, cette contrainte pouvant s'exprimer en termes de nombre de contrôleurs utilisés, si tous les contrôleurs sont identiques ;
- à la disponibilité de l'architecture opérationnelle ;
- à ses performances temporelles ;
- ...

Dans la suite de ce papier, seules les contraintes relatives aux capacités en termes d'entrées/sorties, à la répartition des fonctions et au nombre de contrôleurs seront retenues. En conséquence, le(s) réseau(x) de communication n'est (ne sont) pas considéré(s). Dans la pratique industrielle actuelle, l'affectation des fonctions est réalisée de façon non automatisée, en se basant sur le savoir-faire des concepteurs, et constitue une tâche fastidieuse et consommatrice de temps. Ce papier vise à supprimer ce problème, en proposant une méthode (Figure 1) qui fournit, à partir de la connaissance des caractéristiques des fonctions de commande et des contrôleurs, une solution d'affectation respectant les contraintes de capacités et de répartition et minimisant le nombre de contrôleurs. Cette proposition est basée sur la vérification d'une propriété d'atteignabilité dans un espace d'état discret. Une telle approche a déjà été utilisée avec succès dans [2] et [3]. Cette contribution se distingue de ces deux références, car le temps n'est pas considéré dans un problème d'affectation statique de fonctions. Il convient enfin de souligner que ce travail constitue un préliminaire à l'évaluation des performances temporelles de l'architecture opérationnelle. Les résultats obtenus dans ce domaine, à l'aide de techniques de simulation ([4],

[5], [6]), de vérification formelle [7], ou par des approches algébriques [8], supposent en effet que les temps de traitement des différents contrôleurs sont connus, ce qui nécessite bien entendu que toutes les fonctions aient été préalablement affectées.
L'organisation de ce papier est la suivante. Les éléments du problème (fonctions de commande, contrôleurs, contraintes d'affectation) sont définis formellement dans la section II. La section III décrit le principe d'obtention de la solution d'affectation par recherche d'atteignabilité, tandis que la mise en oeuvre de ce principe à l'aide d'un outil de vérification formelle fait l'objet de la section IV. L'intérêt de cette contribution est montré en section V par le traitement de deux cas : un cas simple à finalité illustrative, puis un cas issu de la réalité industrielle, en vue du passage à l'échelle. Des perspectives de développement sont données en section VI.

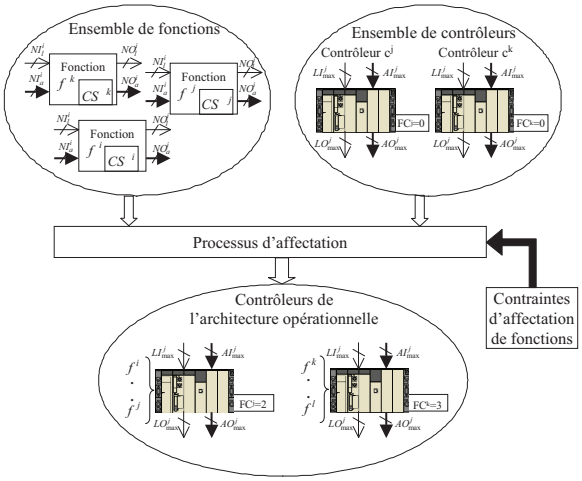


Fig. 1. Objectif de l'étude

II. Modélisation des éléments du problème

L'objectif de cette partie est de présenter l'ensemble des notations utilisées ainsi que d'illustrer le processus d'affectation de fonctions sur un exemple-jouet. Les hypothèses suivantes sont retenues pour cette étude :

- un contrôleur peut héberger de 0 à n fonctions, tant que les contraintes d'affectation sont respectées ;
- une fonction doit être affectée à un et un seul contrôleur.

A. Notations utilisées

Soit F l'ensemble des fonctions f^i, avec $i \in \{1,..,L\}$ et C l'ensemble des contrôleurs c^j, avec $j \in \{1,..,M\}$, L et $M \in \mathbb{N}^*$. L'affectation d'une fonction f^i au contrôleur c^j sera notée : $f^i \leftarrow c^j$.

A.1 Définition formelle d'une fonction

Une fonction $f^i \in F$ est définie comme un 5-uplet $(CS^i, NI_l^i, NI_a^i, NO_l^i, NO_a^i)$ avec :

- $CS^i \in CS$ classement de sûreté ; plus la valeur du classement est faible, plus la fonction est critique. Dans la suite du papier, les fonctions sont rangées en quatre niveaux, donc $CS = \{1,2,3,4\}$;
- $NI_l^i, NI_a^i \in \mathbb{N}$: nombres de données d'entrée, respectivement logiques et analogiques, de la fonction ;
- $NO_l^i, NO_a^i \in \mathbb{N}$: nombres de données de sortie, respectivement logiques et analogiques, de la fonction.

A.2 Définition formelle d'un contrôleur

Un contrôleur $c^j \in C$ est défini comme un 5-uplet $(FC^j, LI_{max}^j, AI_{max}^j, LO_{max}^j, AO_{max}^j)$ avec :

- $FC^j \in FC$ facteur de criticité du contrôleur ; plus la valeur du facteur de criticité est faible, plus le contrôleur doit être fiable. La valeur initiale (aucune fonction affectée au contrôleur c^j) de FC^j est 0, $\forall j$. FC^j est modifié lors du processus d'affectation en respectant la contrainte de répartition énoncée en A.3 ;
- $LI_{max}^j, AI_{max}^j \in \mathbb{N}$ nombres maximaux d'interfaces d'entrée, respectivement logiques et analogiques, du contrôleur ;
- $LO_{max}^j, AO_{max}^j \in \mathbb{N}$ nombres maximaux d'interfaces de sortie, respectivement logiques et analogiques, du contrôleur.

A.3 Contraintes d'affectation

Deux types de contraintes conditionnent l'affectation des fonctions :

1. Contraintes de capacités en termes d'entrées/sorties des contrôleurs,

2. Contraintes de répartition des fonctions.

Ces contraintes sont exprimées respectivement sous forme de contraintes arithmétiques et logiques comme précisé ci-dessous :

- Contraintes de capacités ;
  Soit $F_j = \{f^i \in F | c^j \leftarrow f^i\}$ l'ensemble des fonctions f^i affectées à c^j et $I_j = \{i \in \{1,...,L\} | c^j \leftarrow f^i\}$.
  Il faut alors que : $\forall j \in \{1,...,M\}$,

$$\sum_{i \in I_j} NI_l^i \leq LI_{max}^j \quad (1)$$

$$\sum_{i \in I_j} NI_a^i \leq AI_{max}^j \quad (2)$$

$$\sum_{i \in I_j} NO_l^i \leq LO_{max}^j \quad (3)$$

$$\sum_{i \in I_j} NO_a^i \leq AO_{max}^j \quad (4)$$

- Contraintes de répartition ;
  Une relation $\Re$ de CS vers FC définit les associations possibles entre classement d'une fonction et facteur de sûreté d'un contrôleur :
  $\forall j \in \{1,...,M\}$ tel que $Card(F_j) > 0$, $CS^i \; \Re \; FC^j$.
  Pour des raisons de sûreté, les fonctions doivent être réparties dans les contrôleurs selon leur classement de criticité, les plus critiques ($CS^i = 1$) ne pouvant pas cohabiter dans un même contrôleur avec des fonctions qui le sont moins. Les autres fonctions peuvent être rassemblées dans un seul contrôleur comme suit :
  - fonctions ayant un classement de sureté 2 avec fonctions ayant un classement de sureté 3 ;
  - fonctions ayant un classement de sureté 3 avec fonctions ayant un classement de sureté 4.

  Par conséquent, un contrôleur c^j peut héberger, quand toutes les fonctions sont affectées :
  - soit uniquement des fonctions de classement 1 ;
  - soit des fonctions de classement 2 et des fonctions de classement 3 ;
  - soit des fonctions de classement 3 et des fonctions de classement 4.

La relation $\Re$ est alors définie comme suit :

$$\Re = \{(1,1),(2,2),(3,2),(3,3),(4,3)\} \subset CS \times FC \quad (5)$$

B. Illustration de l'affectation des fonctions

Cet exemple-jouet s'appuie sur un ensemble de cinq fonctions f^1, f^2, f^3, f^4, f^5, dont le classement de sûreté appartient à $\{1,2\} \subset CS$, et qui sont ainsi définies :

- $f^1 = (2,5,4,1,3)$
- $f^2 = (1,5,6,2,4)$
- $f^3 = (2,1,3,6,4)$
- $f^4 = (1,6,8,5,2)$
- $f^5 = (1,3,4,5,2)$

Ces fonctions sont à répartir sur un ensemble de trois contrôleurs c^1, c^2, c^3, dont les capacités sont identiques :
$\forall j \in \{1,2,3\}, LI^j_{max} = AI^j_{max} = LO^j_{max} = AO^j_{max} = 10$

Une solution possible d'affectation des fonctions sur les trois contrôleurs est décrite dans la figure 2. Cette solution a été obtenue en affectant tout d'abord la fonction f^1 au contrôleur c^1, ce qui a fixé la valeur de son facteur de criticité à $FC^1 = 2$. La fonction f^2 a ensuite été affectée au contrôleur c^2, ce qui a fixé la valeur de son facteur de criticité à $FC^2 = 1$. Puis, la fonction f^3 a été affectée au contrôleur c^1 car les contraintes 1, 2, 3, 4 et 5 sont respectées. La fonction f^4 a alors été affectée au contrôleur c^3 car pour le contrôleur c^2, les fonctions f^2 et f^4 ne respectent pas les contraintes 1 et 2. La fonction f^5 a, enfin, été affectée au contrôleur c^2, car pour le contrôleur c^3, la contrainte 2 ne serait pas respectée si f^5 lui était affectée.

Il convient de noter que cette solution n'est pas unique. Comme les contrôleurs possèdent les mêmes nombres d'entrées-sorties, d'autres solutions satisfaisantes peuvent être obtenues par permutation circulaire des contrôleurs ou en permutant simplement deux contrôleurs. Ces solutions sont équivalentes à celle détaillée ci-dessus si aucune autre contrainte n'est introduite.

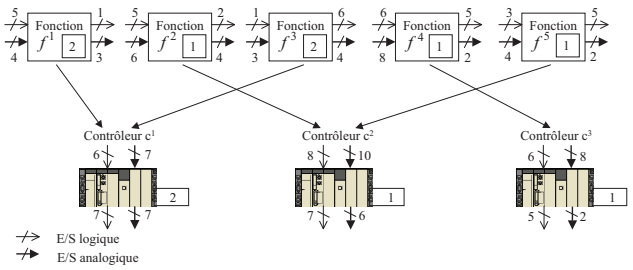


Fig. 2. Exemple d'affectation de cinq fonctions sur trois contrôleurs

III. Méthode proposée

La méthode proposée pour l'affectation automatique des fonctions repose sur deux principes :

- modéliser le problème d'affectation sous la forme d'un ensemble de mécanismes concurrents d'appel-réponse entre des modèles, sous forme d'automates communicants, de demande d'affectation d'une fonction et d'acceptation d'une demande par un contrôleur ;
- rechercher si l'exécution de cet ensemble, réseau d'automates communicants, peut conduire à un état atteignable depuis l'état initial où toutes les fonctions sont affectées.

Cette partie présente tout d'abord le formalisme utilisé pour la construction des modèles de demande et d'acceptation, qui sont ensuite détaillés. L'état du réseau d'automates caractérisant l'affectation de toutes les fonctions est alors défini, ce qui permet d'énoncer la propriété d'atteignabilité recherchée.

A. Définition du formalisme utilisé

Le formalisme retenu est celui d'un réseau d'automates communiquant par variables partagées et synchronisés par des étiquettes de transition. Un automate A est un N-uplet
$A = (S, X, L, T, S_m, s_0, v_0)$, où :

- S est un ensemble de localités ;
- X est un ensemble de n variables entières ;
- L est un ensemble d'étiquettes décomposable en 3 ensembles disjoints L_i, L_o, L_l, où
  - L_i est l'ensemble des étiquettes de réception ;
  - L_o est l'ensemble des étiquettes d'émission ;
  - L_l est l'ensemble des étiquettes locales (internes à un automate).
- T est un ensemble de transitions $(s, l, g, m, s') \in S \times L \times G \times M \times S$, avec G l'ensemble des gardes (conditions sur les variables de X) et M l'ensemble des mises à jour sur les valuations des variables ;
- $S_m \subseteq S$ est un ensemble de localités marquées ;
- $s_0 \in S$ est la localité initiale ;
- $v_0 : X \leftarrow \mathbb{N}$ est la valuation initiale des variables.

Une trace (ou exécution) de A est une succession d'évolutions à partir de l'état initial :
$(s_0, v_0) \xrightarrow{t1} (s_1, v_1) \xrightarrow{t2} (s_2, v_2) ... \xrightarrow{tn} (s_n, v_n)$

Un réseau d'automates $NA = A^1 \| A^2 \| ... \| A^n$ est défini par $NA = (S, X, L, T, S_m, s_0, v_0)$, avec :

- $S \subseteq S^1 \times S^2 \times ... \times S^n$
- $X = X^1 \cup X^2 \cup ... \cup X^n$
- $L = L^1 \cup L^2 \cup ... \cup L^n$
- $T \subseteq S \times L \times G \times M \times S'$, avec G l'ensemble des gardes (conditions sur les variables de X) et M l'ensemble des mises à jour sur les valuations des variables
- $S_m = S^1_m \times S^2_m \times ... \times S^n_m$
- $s_0 = s^1_0 \times s^2_0 \times ... \times s^n_0$
- $v_0 : X \leftarrow \mathbb{N}$ est la valuation initiale des variables.

Une évolution de $NA(s, v) \xrightarrow{t} (s', v')$ est possible si :

- une évolution se produit dans un seul des automates par franchissement d'une transition t portant une étiquette locale, la garde étant satisfaite ;
- deux transitions t^α_k, t^β_m d'un couple d'automates (A^α, A^β) avec t^α_k contenant l'étiquette $l^\alpha_k \in L^\alpha$ et t^β_m contenant l'étiquette $l^\beta_m \in L^\beta$ telles que $l^\alpha_k = l^\beta_m$ sont franchies simultanément, les gardes de ces transitions étant satisfaites.

B. Modèles génériques de demande d'affectation et d'acceptation d'une demande

Les figures 3 et 4 présentent les modèles génériques de demande d'affectation d'une fonction et d'acceptation d'une demande par un contrôleur, notés respectivement δ et α. Les conventions suivantes sont utilisées dans ces modèles :

- les localités initiales sont désignées par un arc source ;
- les localités marquées sont représentées par deux cercles concentriques ;
- les noms des localités sont en gras ;
- les noms des étiquettes sont en italiques et suivis d'un ! (resp. ?) pour les étiquettes d'émission (de réception) ;
- les noms des gardes sont en texte normal ;

– les mises à jour de variables sont soulignées.

De plus, les gardes toujours vraies ainsi que les étiquettes internes ne sont pas représentées par souci de concision.

B.1 Modèle de demande d'affectation

La localité initiale du modèle est « Fonction non affectée ». Une seule transition, qui correspond à l'émission d'une demande d'affectation, est possible à partir de cette localité. Une fois cette demande émise, le modèle attend (localité « Affectation possible ? ») la réponse d'un modèle d'acceptation qui peut être :

– *Refus*, le modèle retourne alors en localité initiale ;
– *Ok*, le modèle évolue vers la localité marquée « Fonction affectée » qui est une localité puits.

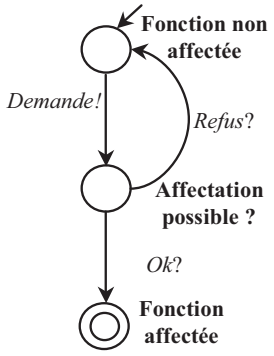


Fig. 3. Modèle générique (δ) de demande d'affectation

B.2 Modèle d'acceptation d'une demande

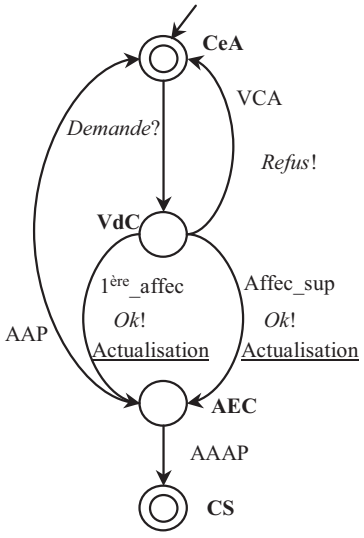
A partir de la localité initiale, qui est également marquée, ce modèle ne peut évoluer vers la localité « Vérification des contraintes (VdC) » qu'à la réception d'une demande d'affectation. Les trois transitions partant de cette dernière localité correspondent à :

– la violation d'une des contraintes d'affectation, l'étiquette *Refus* est alors émise ;
– l'acceptation d'une demande alors qu'aucune autre fonction n'a été préalablement affectée au contrôleur (garde Première affectation vraie (1ère affec)). Les variables $\sum_{i\in I_j} NI_l^i; \sum_{i\in I_j} NI_a^i; \sum_{i\in I_j} NO_l^i; \sum_{i\in I_j} NO_a^i$, sont alors actualisées et le facteur de criticité FC^j prend la valeur du classement de sûreté de la fonction ;
– à l'acceptation d'une demande alors qu'au moins une autre fonction a été préalablement affectée (garde Affectation supplémentaire vraie (Affec sup)). Seules les variables $\sum_{i\in I_j} NI_l^i; \sum_{i\in I_j} NI_a^i; \sum_{i\in I_j} NO_l^i; \sum_{i\in I_j} NO_a^i$, sont actualisées. Le facteur de criticité FC^j n'est pas modifié.

Dans les deux derniers cas, l'étiquette *Ok* est émise.

A partir de la localité « Analyse de l'état du contrôleur (AEC) », deux évolutions sont possibles qui correspondent :

– au fait que toutes les capacités du contrôleur en termes d'entrées/sorties sont atteintes :
$\sum_{i\in I_j} NI_l^i = LI_{max}^j$; $\sum_{i\in I_j} NI_a^i = AI_{max}^j$;
$\sum_{i\in I_j} NO_l^i = LO_{max}^j$; $\sum_{i\in I_j} NO_a^i = AO_{max}^j$;
Le modèle évolue alors vers la localité marquée « Contrôleur Saturé (CS) ».
– au fait qu'au moins une capacité du contrôleur en termes d'entrées/sorties n'est pas atteinte (garde Autre affectation possible vraie (AAP)). Le modèle évolue alors vers la localité initiale « Contrôleur en Attente (CeA) ».



CeA : Contrôleur en Attente ; **VdC** : Vérification des contraintes ;
AEC : Analyse de l'état du contrôleur ; **CS** : Contrôleur saturé ;
VCA : Violation d'une des contraintes d'affectation ;
AAP : Autre affectation possible ; AAAP Aucune autre affectation possible

Fig. 4. Modèle générique (α) d'acceptation d'une demande

C. Modèle instancié pour une architecture fonctionnelle donnée

Ce modèle est un réseau d'automates communicants NA qui comporte :

– autant d'instances $(\delta^1, \delta^2, ..., \delta^L)$ du modèle de la figure 3 qu'il y a de fonctions ;
– M instances $(\alpha^1, \alpha^2, ..., \alpha^M)$ du modèle de la figure 4 ; le choix de M sera discuté dans la section IV.

Il vient donc $NA = \delta^1 \| \delta^2 \| ... \| \delta^L \| \alpha^1 \| \alpha^2 \| ... \| \alpha^M$.

Une évolution synchrone de deux automates n'est possible que si ces deux automates émettent et reçoivent l'un des couples d'étiquettes suivantes :

– *Demande* ! et *Demande* ? ;
– *Ok* ! et *Ok* ? ;
– *Refus* ! et *Refus* ?.

Afin d'éviter des incohérences consistant à ce qu'une instance α^i émette une réponse à destination d'une instance δ^j autre que celle ayant émis la requête d'affection, le mécanisme d'appel-réponse doit être conçu comme une section critique protégée par un sémaphore. La réalisation de cette section critique dépend de l'implantation et ne sera pas discutée plus amplement dans ce papier.

D. Définition de la propriété d'atteignabilité recherchée

Toutes les fonctions sont affectées lorsque la localité marquée est atteinte dans toutes les instances du modèle générique de demande d'affectation. Lorsqu'il en est ainsi, les instances du modèle générique d'acceptation se trouvent soit dans la localité puits, soit dans la localité initiale, qui sont toutes deux marquées. La propriété d'atteignabilité à vérifier peut donc s'énoncer, de

manière informelle, de la manière suivante :
Est-il possible d'atteindre, à partir de l'état initial, un état du réseau d'automates tel que la localité active soit une localité marquée dans tous les automates du réseau ?

IV. Mise en oeuvre à l'aide d'un outil de vérification formelle

Les techniques de vérification formelle par model-checking [9] ont pour objectif de prouver qu'un modèle satisfait une propriété formelle, qui peut être par exemple une propriété d'atteignabilité. Il est donc naturel d'envisager la mise en oeuvre de la méthode proposée à l'aide d'une telle technique. Ceci exige en premier lieu d'énoncer formellement la propriété recherchée, donnée de manière textuelle à la section précédente. En utilisant les quantificateurs de la logique temporelle CTL, cette propriété P s'écrit :

P : EF *Affectation totale*

Affectation totale désignant l'état du réseau tel que la localité active soit une localité marquée pour tous les automates. Cette propriété est vérifiée s'il existe au moins une trace partant de l'état initial du réseau et atteignant *Affectation totale*.

Algorithme 1 Recherche d'une solution d'affectation minimisant le nombre de contrôleurs

Entrées: Caractéristiques des L fonctions
$f^i \in F = (CS^i, NI^i_l, NI^i_a, NO^i_l, NO^i_a), \forall i \in \{1,...,L\}$
Capacités communes à tous les M contrôleurs
$LI^j_{max}, AI^j_{max}, LO^j_{max}, AO^j_{max}, \forall j \in \{1,...,M\}$
Sorties: Nombre minimum N de contrôleurs pour une architecture opérationnelle.
Listes des fonctions affectées à chacun des N contrôleurs
$F_j = \{f^i \in F | c^j \leftarrow f^i\}, \forall j \in \{1,...,M\}$
/* Initialisation : */
$M \leftarrow L; N \leftarrow M$
$NA = \delta^1 \| \delta^2 \| ... \| \delta^L \| \alpha^1 \| \alpha^2 \| ... \| \alpha^M$
/* Construction itérative : */
tant que $NA \models P$
  si $\exists F_j = \emptyset | j \in \{1,...,N\}$ **alors**
    $N \leftarrow M - Card(I) - 1$ avec
    $I = \{I_j, j \in \{1,..,M\} | I_j = \emptyset\}$
  sinon
    $N \leftarrow N - 1$
  fin si
fin tant que
$N \leftarrow N + 1$
/* Afficher : */
– N ;
– F_j, $j \in \{1,...,N\}$.

La recherche d'une solution d'affectation peut donc être réalisée en prouvant que le réseau d'automates NA satisfait la propriété ci-dessus, ce qui sera noté : $NA \models P$. Il est évident que cette preuve dépend du nombre M d'instances du modèle d'acceptation α. Il est par exemple illusoire de chercher à affecter à un seul contrôleur ($M = 1$) deux fonctions ($L = 2$) si ces fonctions ont des classements de sureté incompatibles. On peut cependant remarquer que, si : $\forall i$, $\forall j$, $NI^i_l \leq LI^j_{max}$, $NI^i_a \leq AI^j_{max}$, $NO^i_l \leq LO^j_{max}$, $NO^i_a \leq AO^j_{max}$, le cas $M = L$ conduit à coup sûr à une preuve positive. L'examen de la trace conduisant à l'état *Affectation totale* montre en général dans ce cas que tous les contrôleurs ne sont pas utilisés ; seul un nombre $N < M$ hébergent une ou plusieurs fonctions.

Afin de réduire le nombre de contrôleurs utilisés dans l'architecture opérationnelle, la démarche par preuves itératives de l'algorithme 1 a donc été développée. Cette démarche consiste, lorsqu'une solution a été trouvée, à s'assurer qu'il n'existe pas de solution si on réduit le nombre de contrôleurs.

V. Etudes de cas

A. Choix de l'outil de vérification formelle

Plusieurs outils de vérification formelle par model-checking, tels que NuSMV, SPIN, UPPAAL, peuvent être envisagés pour réaliser l'analyse d'atteignabilité sur laquelle s'appuie la recherche d'une solution d'affectation. L'outil UPPAAL [10] a été retenu car il possède une interface graphique très ergonomique et peut fournir une trace d'exécution en cas de preuve positive. Il importe de souligner que seules ces caractéristiques ont motivé ce choix ; la capacité de cet outil à vérifier des propriétés sur des modèles temporisés ne constitue en aucun cas un critère de sélection, les automates communicants utilisés dans ce travail n'étant pas temporisés.

B. Premier cas d'étude

Ce cas simple vise à illustrer l'approche. L'architecture fonctionnelle comporte vingt fonctions définies dans la table 1 et les caractéristiques des contrôleurs sont les suivantes :
$\forall j \in \{1,...,M\}$, $LI^j_{max} = AI^j_{max} = LO^j_{max} = AO^j_{max} = 32$

TABLE I
Description des fonctions

Fonctions	CS^i	NI^i_l	NI^i_a	NO^i_l	NO^i_a
1	1	3	5	2	4
2	1	4	6	1	3
3	2	3	7	2	1
4	2	3	4	4	3
5	2	7	5	6	2
6	2	7	2	8	6
7	1	4	5	2	4
8	2	3	6	4	5
9	1	3	7	4	2
10	1	7	2	6	4
11	3	3	5	2	4
12	4	4	6	1	3
13	3	3	7	2	1
14	4	3	4	4	3
15	3	7	5	6	2
16	2	7	2	8	6
17	4	4	5	2	4
18	3	3	6	4	5
19	4	3	7	4	2
20	1	7	2	6	4

Une première analyse d'atteignabilité avec un nombre initial de contrôleurs $M = 20$ fournit une solution où seuls $N = 5$ contrôleurs hébergent au moins une fonction (15 contrôleurs ne sont pas utilisés). En effectuant une nouvelle analyse avec un

nombre initial de contrôleurs $M = 4$, une solution plus compacte est trouvée, qui comporte 4 contrôleurs réellement utilisés. Il n'est cependant pas possible de réduire encore le nombre de contrôleurs, l'analyse avec $M = 3$ ne fournissant aucune solution. La solution d'affectation finale pour $N = 4$ contrôleurs est détaillée dans la table II.

TABLE II
CARACTÉRISTIQUES DES CONTRÔLEURS ET RÉPARTITION DE L'ENSEMBLE DES FONCTIONS DE LA TABLE I

c^j	FC^j	F_j	$\sum_{i\in I_j} NI_l^i$	$\sum_{i\in I_j} NI_a^i$	$\sum_{i\in I_j} NO_l^i$	$\sum_{i\in I_j} NO_a^i$
c^1	3	$\{f^{12}, f^{13}\}$	7	13	3	4
c^2	2	$\{f^3, f^4, f^5, f^6, f^8, f^{16}\}$	30	26	32	23
c^3	3	$\{f^{11}, f^{14}, f^{15}, f^{17}, f^{18}, f^{19}\}$	23	32	22	20
c^4	1	$\{f^1, f^2, f^7, f^9, f^{10}, f^{20}\}$	28	27	21	21

Les durées des différentes analyses nécessaires à l'obtention de cette solution figurent dans la table III.

TABLE III
DURÉE DE LA RECHERCHE D'ATTEIGNABILITÉ POUR L'ENSEMBLE DES FONCTIONS DE LA TABLE I

Nombre initial de contrôleurs	20	5	4
Durée	NM	NM	6 s

NM : non mesurable (trop faible pour être mesurée)

C. Deuxième cas d'étude

Afin d'évaluer les possibilités de passage à l'échelle, une étude basée sur 200 fonctions a été ensuite entreprise. Ces deux cents fonctions sont toutes différentes les unes des autres, leurs caractéristiques étant telles que :
$\forall i \in \{0,...,L\}, CS^i \in \{1,2,3,4\}, NI_l^i \in \{0,...,9\}$,
$NI_a^i \in \{0,...,9\}, NO_l^i \in \{0,...,9\}, NO_a^i \in \{0,...,9\}$.
Les contrôleurs utilisés sont tous identiques et ont comme caractéristiques :
$\forall j \in \{1,...,M\}, LI_{max}^j = AI_{max}^j = LO_{max}^j = AO_{max}^j = 32$.
Une première recherche d'atteignabilité a été effectuée, qui fournit une solution d'affectation où des contrôleurs n'hébergent aucune fonction ; le nombre de contrôleurs réellement utiles est alors $N = 37$. L'analyse effectuée avec $M = 36$ fournit également une solution. Le nombre de contrôleurs nécessaires à la réalisation de l'architecture opérationnelle ne peut cependant plus être réduit, l'analyse avec $M = 35$ contrôleurs ne fournissant aucune solution. La solution d'affectation utilise donc au minimum $N = 36$ contrôleurs.

La table IV indique les durées des différentes recherches d'atteignabilité effectuées dans cette étude de cas. Ces valeurs montrent que l'approche proposée est tout à fait envisageable dans un contexte industriel de conception d'architectures opérationnelles.

TABLE IV
DURÉE DE LA RECHERCHE D'ATTEIGNABILITÉ AVEC L=200

Nombre initial de contrôleurs	200	37	36
Durée	110 s	46 s	50 s

VI. CONCLUSION ET PERSPECTIVES

Ce papier a montré qu'il est possible d'obtenir une solution d'affectation de fonctions de commande sur des contrôleurs qui respecte des contraintes de capacités et de répartition et minimise le nombre de contrôleurs nécessaires. Cette contribution s'appuie sur une analyse d'atteignabilité dans un réseau d'automates communicants à variables qui représentent des mécanismes concurrents d'appel-réponse. La mise en oeuvre de cette proposition, à l'aide d'un outil de vérification formelle, sur un exemple de taille non triviale montre ses potentialités d'application.

Les travaux en cours concernent la prise en compte d'autres contraintes de capacités, comme la charge CPU, et de répartition. L'introduction de différents types de contrôleurs et la recherche d'un ensemble de solutions d'affectation, et non d'une seule solution, sont également envisagées. Les perspectives à plus long terme visent à intégrer des contraintes relatives aux performances temporelles attendues de l'architecture opérationnelle, ce qui conduira à introduire dans la modélisation des caractéristiques du réseau de communication.

RÉFÉRENCES

[1] F. SIMONOT LION et J-P. ELLOY : An Architecture Description Language for In-Vehicle Embedded System Development. *In 15th Triennial World Congress of the International Federation of Automatic Control*, Barcelona, Spain, 2002. Elsevier Science.

[2] Gerd BEHRMANN, Ed BRINKSMA, Martijn HENDRIKS et Angelika MADER : Production scheduling by reachability analysis - a case study. *Parallel and Distributed Processing Symposium, International*, 3:140–147, 2005.

[3] S. SUBBIAH et S. ENGELL : Short-Term Scheduling of Multi-Product Batch Plants with Sequence-Dependent Changeovers Using Timed Automata Models. *In 20th European Symposium on Computer Aided Process Engineering*, volume 28, pages 1201–1206, 2010.

[4] G. MARSAL, B. DENIS, J-M. FAURE et G. FREY : Evaluation of Response Time in Ethernet-based Automation Systems. *In Proceedings of the 11th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA'06*, Prague, Czech Republic, September 2006.

[5] P. MEUNIER, B. DENIS et J-J. LESAGE : Temporal performance evaluation of control architecture in automation systems. *In Proceedings of 6th EUROSIM Congress on Modelling and Simulation*, Ljubljana, Slovénia, September 2007.

[6] D. A. ZAITSEV : An Evaluation of Network Response Time using a Coloured Petri Net Model of Switched LAN. *In Proceedings of 5th Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN Tools*, pages 157–167, Aarhus, Denmark, october 2004.

[7] S. RUEL, O. DE SMET et J-M. FAURE : Finding the bounds of response time of networked automation systems by iterative proofs. *In Proceedings of the 13th IFAC Symposium on Information Control Problems in Manufacturing (INCOM 2009)*, pages 1365–1370, Moscow, Russia, June 2009.

[8] B. ADDAD, S. AMARI et J-J. LESAGE : Analytic Calculus of Response Time in Networked Automation Systems. *IEEE Transactions on Automation Science and Engineering*, Vol. 7-4:858–869, April 2010.

[9] B. BÉRARD, M. BIDOIT, A. FINKEL, F. LAROUSSINIE, A. PETIT, L. PETRUCCI et P. SCHNOEBELEN : *Systems and Software Verification*. Springer, 2001.

[10] G. BEHRMANN, J. BENGTSSON, A. DAVID, K. LARSEN, P. PETTERSSON et W. YI : UPPAAL implementation secrets. *In Proc. of 7th International Symposium on Formal Techniques in Real-Time and Fault Tolerant Systems (FTRTFT)*, Oldenburg, Germany, september 2002.