Synthèse de Lois de commande pour la configuration et la reconfiguration des systèmes industriels complexes

Sébastien Henry

Institut National Polytechnique de Grenoble

No. attribué par la bibliothèque

THESE

pour obtenir le grade de

DOCTEUR DE L'INPG

Spécialité : Automatique-Productique

préparée au Laboratoire d'Automatique de Grenoble

dans le cadre de l'École Doctorale :
Électronique, Électrotechnique, Automatique, Traitement du Signal

présentée et soutenue publiquement

par

Sébastien Henry

le 07 octobre 2005

Titre :

Synthèse de Lois de Commande pour la Configuration et la Reconfiguration des Systèmes Industriels Complexes

Directeur de thèse : Éric ZAMAÏ

JURY

Pierre LADET	Président
Étienne CRAYE	Rapporteur
Nidhal REZG	Rapporteur
Audine SUBIAS	Examinateur
Pascal BERRUET	Examinateur
Michel COMBACAU	Invité
Eric ZAMAÏ	Directeur de thèse
Mireille JACOMINO	Co-Encadrant

à mes Parents, Marie et Lucas

Avant-Propos

Le travail présenté dans ce mémoire a été préparé au Laboratoire d'Automatique de Grenoble (LAG), dans le cadre d'une allocation du Ministère de l'Education Nationale, de la Recherche et de la Technologie. Je remercie Monsieur Alain Barraud, directeur du LAG, de m'avoir accueilli et permis d'effectuer mes travaux de recherche dans d'excellentes conditions.

En premier lieu, je tiens à exprimer toute ma gratitude à mon directeur de thèse, Monsieur Eric Zamaï, Maître de Conférences à l'Institut National Polytechnique de Grenoble, tout d'abord pour avoir accepté de m'encadrer, et ensuite pour la qualité de cet encadrement de part sa disponibilité, ses conseils et son aide durant ces trois années de thèse. Pour tout cela et les longues discussions autour d'un café avec une "chocolatine", sans lesquelles un très grand nombre d'idées n'auraient sans doute jamais vu le jour, je te renouvelle toute ma reconnaissance.

Je tiens également à exprimer toute ma gratitude à Madame Mireille Jacomino, Professeur à l'Institut National Polytechnique de Grenoble, qui a co-encadré ces travaux. Merci pour s'être toujours rendue disponible quand il le fallait, et pour tous les commentaires et toutes les réflexions toujours très justes qui ont grandement contribué à ces travaux.

Je remercie Monsieur Pierre Ladet, Professeur à l'Institut National Polytechnique de Grenoble, pour avoir accepté de présider mon jury.

Je remercie tout particulièrement Messieurs Étienne Craye, Professeur à l'École Centrale de Lille, et Nidhal Rezg, Professeur à l'Université de Metz, pour l'intérêt qu'ils ont porté à mes travaux de thèse et pour avoir accepté la charge d'en être les rapporteurs.

Je tiens également à remercier Madame Audine Subias, Maître de Conférences à l'Institut National des Sciences Appliquées de Toulouse, et Monsieur Pascal Berruet, Maître de Conférences à l'Université de Bretagne Sud, pour l'honneur qu'ils m'ont fait en acceptant d'être examinateurs de ce travail.

Je remercie également Monsieur Michel Combacau, Professeur à l'Université Paul Sabatier de Toulouse, pour sa lecture attentive du mémoire, et d'avoir accepté de participer à ce jury.

Je souhaite exprimer mes remerciements aux membres de l'équipe COSP pour leur soutien et leurs conseils. Je pense à Bernard Descotes-Genon, Marie Laure Espinouse, Alexia Gouin, Maria

Di Mascolo, Alexis Aubry, Ha Duy Long, et tout particulièrement à ceux qui m'ont supporté dans leur bureau Elisabeth Chuiton et André Rossi. Enfin, je tiens à remercier chaleureusement Eric Deschamps pour avoir beaucoup apporté à ces travaux et pour sa lecture attentive du mémoire.

Mes remerciements vont également à l'ensemble du personnel des services informatique et administratif du LAG, en particulier à Pascal Bellemain avec qui travailler a été un réel plaisir, et Olivier Chabert que j'ai souvent dérangé.

Parallèlement à mes travaux de recherche, le CIES de Grenoble m'a accordé un poste de moniteur à l'ENSIEG. Je remercie Monsieurs Gerard Cognet, directeur du CIES de Grenoble. Mes remerciements vont également à Monsieur Roland Vidil, Directeur de l'ENSIEG, et aux enseignants de l'ENSIEG, en particulier Messieurs Eric Escande et Patrick Gounon, mes tuteurs au CIES.

Je souhaite remercier toutes les personnes sans qui je n'aurai pas pu commencer cette thèse, en particulier Monsieur Marc Lavigne, ancien chef des travaux du Lycée Catherine et Raymond Janot à Sens, pour m'avoir facilité le suivi d'un DEA en parallèle de mon poste de TZR et avoir soutenu ma demande de disponibilité.

Mes remerciements s'adressent tout particulièrement à ma famille, mes parents pour leur amour et leur soutien sans faille, ma sœur qui avait ouvert la voie, Marie pour avoir accepté de me suivre dans cette aventure et à Lucas pour le bonheur qu'il m'apporte tous les jours. Je n'oublie pas non plus le soutien de mes beaux parents.

Je profite de ces quelques lignes pour remercier les personnes qui ont accepté la très lourde tâche de relire mon manuscrit, en particulier Messieurs Giovanni Zamaï et Gregory Roche. Un grand merci également à tous mes copains Fred, Chris, Sylvain, Yann, Gilou, Marco, Vincent, Cyrille, Small, Yannick, Cedric, Cyril, Teem, Stéphane, Alex, Nico, Francine, Sandra, Oreka, Alexine, Sabine, Valérie, Clemence, Isabelle, Edwige, Virginie, Delphine, Audrey, Leslie, et les autres...

Table des matières

Introduction générale

Depuis la naissance de l'industrie manufacturière, et en particulier des premières méthodes qui ont été proposées pour automatiser ses processus de production, le contexte de développement des produits et services associés s'est considérablement durci. Partie initialement de productions locales basées sur des petites et moyennes entreprises, la production se retrouve aujourd'hui "éclatée" sur de nombreux sites nationnaux et internationnaux. La raison principale en est une recherche tout azimut de réduction des coûts, et en particulier ceux de la main d'œuvre ouvrière. Cependant, force est de constater aujourd'hui que les gains réalisés à court terme, seront vraisemblablement difficiles à maintenir dans l'avenir. La délocalisation serait-elle un pari risqué ?

Pour tenter d'apporter quelques éléments de réponse à cette question, il est important de comprendre que pour maintenir une entreprise au plus haut niveau de compétitivité, la seule réduction des coûts de la main d'œuvre ne suffit pas. En effet, rester compétitif, c'est prendre conscience du fort niveau d'incertitude installé aux antipodes du système de pilotage de l'entreprise. Elle se retrouve en effet écartelée entre une variabilité excessive de la demande (délais de fabrication réduits, modification des spécifications de dernière minute, ...), des coûts de transports fortement perturbés par le cours du pétrole, et enfin une fiabilité des ateliers de production toute relative (prés de 60% du temps de production est passé en mode dégradé).

Bien entendu, pour faire face à de telles difficultés, des méthodes et des outils visant à automatiser la réactivité à ces aléas de fonctionnement ont été mis en place, mais pas partout... On parlera par exemple de méthodes de type ERP et MES pour les trois premiers niveaux du CIM. Ces méthodes mettent à disposition un ensemble de fonctionnalités capables d'aider l'expert dans ses tâches de réponse à des appels d'offre, de planification ou encore d'ordonnancement réactif. En revanche, pour les trois derniers niveaux du CIM, force est de constater que le niveau d'automatisation des processus réactifs aux aléas de fonctionnement est relativement bas. Par exemple, il n'existe pas, à notre connaissance, de fonctionnalité capable de concevoir automatiquement, voire même semi-automatiquement, les programmes (base de recettes) des automates programmables. A ce jour, seules des équipes d'ingénieurs méthode réalisent sur leur unique niveau d'expertise les bases de recettes du système de pilotage en réponse à des ordres de fabrication, et seules des équipes de maintenance sont capables de gérer l'incertain lié aux parties opératives, ne serait-ce que pour envisager des reprises de cycle.

Force est donc de constater que délocaliser présuppose être en mesure de délocaliser également toutes les compétences humaines de l'entreprise afin de maintenir son niveau de réactivité, fortement lié, comme nous venons de le voir au niveau d'expertise de ses personnels. Cependant, il faut prendre conscience du fait qu'il est d'une part difficile de délocaliser tout un personnel,

ne serait-ce que pour des raisons familiales, et d'autre part, pour ceux acceptant une telle aventure, de garantir leur pérennité dans le poste (risque fort de perte de la connaissance de l'expert).

Toutes ces considérations ne font qu'accroître l'intérêt que nous devons porter à la capitalisation des connaissances de l'expert et aux méthodes d'exploitation de cette connaissance afin de l'aider à prendre les bonnes décisions, quelque soit son niveau d'expertise.

Le travail que nous présentons dans ce document propose d'apporter sa contribution au domaine du pilotage réactif d'ateliers de productions manufacturières. Comme nous avons pu le voir, pour maintenir leur compétitivité dans un contexte concurrentiel déjà difficile et durci encore par les délocalisations, les entreprises doivent faire face à de plus en plus d'incertitudes sur les demandes, les ateliers, et même sur la pérennité des experts dans leur poste.

Tout ceci encourage le développement de systèmes d'aide à la décision. Pour permettre au procédé de continuer à remplir sa mission suite à un aléa, ces systèmes doivent par exemple être en mesure de générer en temps réel de nouvelles configurations des systèmes de commande qui y sont associés. Les mécanismes décisionnels liés à une telle activité doivent s'appuyer sur des informations précises à la fois sur le procédé, la gamme ou encore le produit. Dans ce cadre, une approche globale partant de l'élaboration d'une méthodologie de modélisation des parties opératives jusqu'à la spécification d'algorithmes de synthèse de lois de commande optimales a été proposée. Ainsi, en sus d'un apport significatif en terme de capitalisation des connaissances au sein d'une entreprise, cette approche permet à ce jour non seulement de réduire considérablement le temps de conception des bases de recettes, mais également d'envisager son utilisation dynamique dans le cadre d'un recouvrement suite à l'occurrence de défaillances. L'ensemble des résultats à ce jour obtenus a été validé " en manuel " sur de nombreux procédés, et "in situ" sur le procédé pilote SAPHIR du Laboratoire d'Automatique de Grenoble.

Ce mémoire est organisé en quatre parties dont les thèmes sont donnés ci-après :

La première partie présente de manière générale la problématique et la démarche que nous proposons pour d'une part aider l'expert à capitaliser les connaissances de la partie opérative et d'autre part pour exploiter cette connaissance dans un but de synthèse de lois de commande. Dans cet objectif, le contexte général de l'étude est précisé. Ainsi, partant d'une vision générale du système automatisé de production, nous amenons progressivement le lecteur sur le terrain précis de l'étude. Ce dernier, clairement localisé sur les capacités décisionnelles des systèmes de Supervision, Surveillance et Commande de chaînes fonctionnelles, est alors détaillé. Orienté par cette fermeture de contexte, un état de l'art basé sur des approches capables de concevoir des lois de commande est présenté. Ainsi, et ce pour des domaines différents, méthodes de planification en intelligence artificielle, techniques d'ordonnancement et de génération de gammes opératoires, théorie de la supervision et démarche industrielle sont présentées et analysées. Fort des avantages et inconvénients présentés, cette partie s'achève par la présentation de la problématique que nous nous proposons de traiter.

La partie II est entièrement dédiée à la présentation du formalisme de modélisation que nous proposons pour aider l'expert à capitaliser les connaissances de la partie opérative qui doit être commandée, surveillée et supervisée. Ainsi, les caractéristiques principales du modèle sont tout d'abord présentées notamment en terme d'informations à modéliser et de structuration à préconiser. Sur la base de telles propositions, le formalisme dérivé du concept d'opération utilisé dans le domaine de la Planification Automatique est présenté. Ce concept d'opération est alors étendu aux besoins de notre problématique en considérant notamment les spécificités des systèmes de production ancrées sur des problèmes de transformation, de transitique et de préparation de machines. Le formalisme qui en découle, structuré, intuitif et adapté à la modélisation des systèmes complexes, est également conforté par une démarche de modélisation visant à guider l'expert dans sa phase d'appropriation de la connaissance du système contrôlé considéré.

La partie III est quant à elle consacrée à la proposition de la démarche globale de synthèse de lois de commande optimales. Naturellement fondée sur la structure du formalisme de modélisation proposé dans la partie II, les principes généraux de la démarche de synthèse sont tout d'abord amenés. Ils se caractérisent d'une part par une proximité forte avec les techniques de recherche de chemins dans un graphe, et d'autre part par la recherche d'un compromis permettant de s'affranchir de la complexité des graphes d'états à manipuler sans pour autant mettre à mal la performance du système de pilotage réactif. Dans ce but, un mécanisme de décomposition du problème en problèmes de difficulté moindre est proposé pour nous amener à décrire un premier algorithme de synthèse de lois de commande cycliques dans un contexte mono-flux de produits. Après quoi, le mécanisme de synthèse est généralisé à la problématique multi-flux de produits.

Dans la partie IV, nous développons un exemple d'application sur un atelier réel. Le procédé pilote considéré (SAPHIR) est celui du Laboratoire d'Automatique de Grenoble. Après avoir montré l'adéquation d'un tel procédé à l'évaluation de techniques de synthèse de lois de commande à la fois dans un contexte de configuration de bases de recettes mais également dans celui, particulier, d'une reprise de cycle, ce chapitre applique, pas à pas, notre approche. Ses principaux avantages tels que l'assistance à la modélisation, la proposition d'un formalisme de modélisation très structuré, intuitif et proche de l'expert, les possibilités offertes en terme de validation, l'aspect générique de la démarche de synthèse, son niveau de couverture par rapport à la problématique industrielle et enfin sa mise en œuvre au sein d'un atelier logiciel y sont exposés.

Première partie

Cadre de l'étude

Chapitre 1

Contexte

1 Introduction

Ce chapitre est consacré à la présentation du contexte de notre travail et à la définition des concepts de base de notre étude. Volontairement localisé au niveau du pilotage temps réel des ateliers flexibles de production manufacturières, ce chapitre va progressivement nous amener à fermer le contexte de notre travail et à définir les concepts fondamentaux à retenir pour faciliter par la suite, la lecture du document.

Nous commencerons tout d'abord par présenter les caractéristiques des Systèmes Automatisés de Production (SAP). Cette présentation servira à mettre en évidence la complexité à laquelle les SAP doivent faire face, notamment lors des phases de configuration et de reconfiguration des systèmes de commande. Ensuite, et au delà de la prise en considération des aspects organisationnels et décisionnels du système de pilotage, une introduction générale à la problématique de la réactivité aux aléas de fonctionnement est amenée. Elle conduit, en particulier, à présenter les grandes catégories d'aléas et les processus réactifs qui peuvent leur être associés. Fort de cette présentation générale, ce chapitre nous amène alors au cœur de l'approche globale de pilotage réactif développée au Laboratoire d'Automatique de Grenoble et dans laquelle nos travaux prennent place. Au delà d'une présentation de ses caractéristiques principales, une étude critique est proposée afin de conférer à cette approche une meilleure adéquation de sa structure décisionnelle à la réactivité.

2 Les Systèmes Automatisés de Production (SAP)

Les Systèmes Automatisés de Production (SAP) sont une classe particulière de Systèmes de Production (SP) dont la majeure partie des tâches est automatisée. Historiquement, l'automatisation a commencé par celles des tâches les plus pénibles ou dangereuses pour l'opérateur humain. Elle s'est ensuite poursuivie afin d'améliorer la productivité et la qualité des produits dans la perspective d'accroître les bénéfices de l'entreprise. Nos travaux se positionnant au cœur de ces SAP, nous nous proposons ici, de préciser davantage leur rôle, organisation et structure.

2.1 Rôle

Qu'il soit automatisé ou non, la fonction principale d'un système de production reste la même. Aussi, face à la recherche de la satisfaction du client exprimée généralement en terme de délais de livraison, de quantité, et de qualité de produit (formes géométriques (ISO-1101, 2004), état de surface), il vise à transformer le flux de produits entrant (matières premières) dans le système de production en un flux de produits sortant tel que son état de sortie concorde avec les demandes clients (AFNOR, 1991). Soulignons également que dans tous les cas, la transformation d'un produit par le SAP doit conduire à lui conférer, sur le plan économique, une valeur ajoutée vitale à l'entreprise (bénéfices).

De cette description volontairement condensée de la fonction générale d'un SAP, deux remarques au moins doivent être faites afin d'en montrer toute la difficulté.

D'une part, réaliser cette fonction nécessite de prendre en compte un ensemble de contraintes visant à assurer l'intégrité des biens et des personnes. Ces contraintes, généralement décrites en terme de sécurité et d'écologie, sont imposées et définies par des normes (Mendez et al., 2003). Celles-ci peuvent être très nombreuses. Elles sont fortement dépendantes du procédé de fabrication utilisé (usinage, moulage, procédé chimique, etc) et des matières d'œuvre utilisées (alimentaires, corrosives, explosives, etc).

D'autre part, afin de répondre à la fois aux besoins du client et à ceux internes à l'entreprise, des critères de productivité et de qualité doivent être respectés ; ils sont en effet les leviers pour diminuer les coûts de production et pour améliorer la satisfaction client par le respect de sa demande. Rappelons que la qualité ne se limite pas à la qualité technique du produit. Mais le terme qualité est ici utilisé dans son sens le plus large dénommé parfois qualité globale (Clavier, 1997). Elle peut se décliner selon trois principes : "ce qu'ils voulaient, quand ils le voulaient, au prix convenu" correspondant aux trois dimensions classiques de la qualité coût-délai-performance (ISO-9000, 2000).

La complexité du SAP naît essentiellement du "mariage délicat" de ces contraintes qui doivent au moins être respectées et des critères qui doivent être au mieux suivis afin de garantir la pérennité de l'entreprise. Une première solution à ce problème est amenée par la partie physique même du SAP.

2.2 Organisation Physique et Flexibilité

Face à cette complexité et suivant les transformations possibles (assemblage, mise en forme, contrôle dimensionnel), la partie physique du SAP peut être organisée selon différents niveaux. De manière générale, il est classique de trouver dans la littérature (Tawegoum, 1995) une organisation successivement décomposée en usines, ateliers, machines, stocks, postes, et stations le tout connecté à des services de transport assumant non seulement l'approvisionnement en matières premières mais également la livraison des produits finis.

Dans le détail, les usines sont divisées en ateliers dédiés à un type de produit ou à une fonction spécifique comme par exemple la peinture ou l'usinage. Les ateliers sont connectés entre eux par des systèmes de transport dédiés tels que des chariots filoguidés. Plusieurs cellules regroupant trois à quatre machines (robot de soudage, machine à commande numérique, machines à mesurer tridimensionnelle) et des stocks tampons constituent un atelier. Les

machines sont reliées par un système de transport de type convoyeur ou robot de manutention. Un sous-ensemble d'une machine permettant d'effectuer un ensemble d'opérations sur le produit maintenu en position est appelé station. Un poste est une partie d'une station associée à une fonction spécifique sur le produit (fonction : opérative, d'identification, de mesure, etc). Et enfin la plus petite entité, la chaîne fonctionnelle, réalise une fonction élémentaire. Nous reviendrons plus en détail sur cette notion de chaîne fonctionnelle lors de l'étude structurelle d'un SAP.

Dans un contexte incertain, lié en partie à la forte variabilité de la demande client, cette organisation physique ne peut à elle seule garantir le maintien des performances (respect des critères de qualité et de productivité). Pour cette raison, un agencement des différentes entités physiques et des redondances existantes est généralement proposé afin de gagner en flexibilité de la partie opérative (Berruet, 1998). Ainsi, cette flexibilité physique apporte d'une part plusieurs possibilités pour fabriquer un même produit et d'autre part la capacité de fabriquer de nouvelles variantes de produits existants ou de nouveaux produits. L'objectif principal de ce type de flexibilité est donc d'offrir aux clients une grande variété de produits en utilisant les mêmes ressources matérielles. Cependant, et en anticipant un peu sur la suite de ce document, notons que cette flexibilité joue également un rôle primordial pour répondre aux incertitudes liées aux aléas de fonctionnement de la partie opérative.

La seule flexibilité physique ne permet pas de garantir la flexibilité de l'ensemble du système automatisé de production. Le système de pilotage doit lui aussi être en mesure de s'adapter d'une part à la demande du client, et d'autre part aux défaillances matérielles. Nous parlerons alors de flexibilité décisionnelle. Mais préalablement à l'étude de ces aspects décisionnels, prenons le temps d'explorer plus avant la structure interne du SAP.

2.3 Structure générale d'un SAP

D'une manière générale, les SAP ont une structure interne composée de quatre éléments (cf. Figure 1.1). Au delà des éléments de base qui caractérisent un système de production (flux de produits et partie opérative), deux autres éléments sont à considérer en vue de son automatisation, à savoir une interface et un système de pilotage.

Le flux de produits représente l'ensemble des entités en cours de transformation.

La partie opérative (PO) regroupe l'ensemble des organes physiques qui interagissent sur le flux de produits pour modifier son état au sens le plus large (état physique et position spatiale). Elle se compose des actionneurs et effecteurs.

Le système de pilotage (Trentesaux et Sénéchal, 2002) dont la fonction est de définir, d'imposer et de surveiller l'évolution du flux de produits et de la partie opérative afin de les amener conjointement d'une situation à une autre concordant avec les demandes client tout en satisfaisant les contraintes de sécurité et d'écologie, et de plus en optimisant les critères de productivité et de qualité. Il se compose d'une partie matérielle (calculateurs, réseaux de communication) et d'une partie logicielle (système d'exploitation, interpréteur de commande, ...) dont la prépondérance est naturellement croissante (adaptabilité, reconfigurabilité, maintenabilité).

L'interface, constituée de pré-actionneurs et de capteurs, transmet les ordres du système de pilotage à la PO et informe ce dernier de l'état de la PO et/ou du flux de produits.

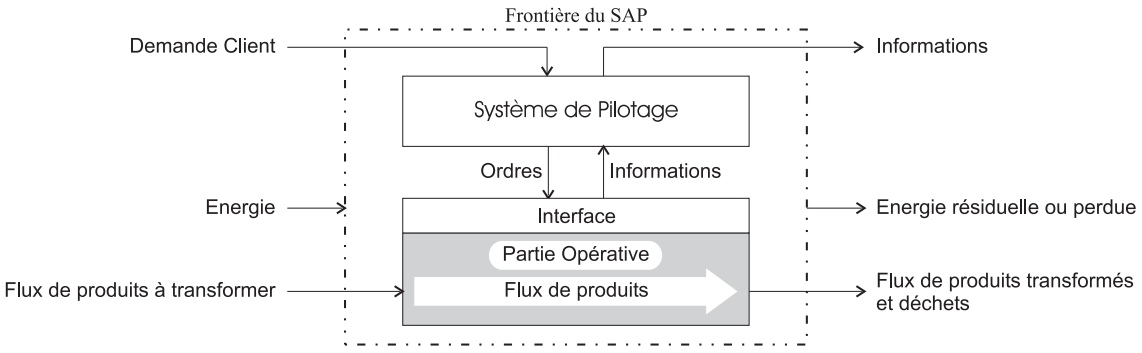


FIG. 1.1 – Entrées/Sorties d'un SAP avec sa structure interne.

Compte-tenu de ces éléments de base, et rappelons-le de la complexité croissante en terme de pré-actionneurs, actionneurs, effecteurs, et capteurs dont le nombre a considérablement augmenté, il est devenu difficile d'en maîtriser la complexité. Pour tenter de répondre à ce problème crucial, de nombreuses architectures de pilotage ont été proposées (Kramer et Senehi, 1993). Une des solutions la plus usitée consiste à organiser le système de pilotage en plusieurs niveaux de complexité moindre.

2.4 Le Système de pilotage

Durant les années 1970 et 1980, le concept de "Computer Integrated Manufacturing" (CIM) a été proposé (CIM, 1989). Le modèle CIM, en forme de pyramide, définit une décomposition hiérarchique et modulaire d'un SAP en cinq niveaux. A partir de ce concept, de nombreux modèles ont été proposés (Doumeingts et al., 1995) (Unger, 2001). Parmi eux, nous retiendrons le modèle à cinq niveaux complété d'un niveau supplémentaire (Macedo et al., 2004), en conformité avec la récente norme S95 (ISA-S95, 2000) (Veille, 2000). Ces niveaux (Kjaer, 2003) sont les suivants :

Niveau 5 : Information Stratégique (Direction générale de l'entreprise), Production de l'entreprise, planification (Affectation de la charge aux usines) ;

Niveau 4 : Production de l'usine, planification (Direction opérationnelle usine) ;

Niveau 3 : Coordination inter-unités (Supervision globale) ;

Niveau 2 : Contrôle de l'unité (contrôle et supervision) ;

Niveau 1 : Contrôle de process (Automates, Contrôle local) ;

Niveau 0 : Partie Opérative (Transformations physiques).

Lorsque la complexité du niveau 0 devient trop importante, en sus de la hiérarchisation globale du système de pilotage, un cloisonnement, au sein de chaque niveau, en modules de pilotage qui n'ont pas de liens directs est proposé (Combacau, 1991). Il en résulte une hiérarchie décisionnelle dans laquelle un module n'est lié qu'à un seul module du niveau d'abstraction supérieur qui coordonne plusieurs modules de niveau inférieur (cf. Figure 1.2). Au niveau le plus bas du système de pilotage, les modules sont appelés *chaînes fonctionnelles* (Merlaud et al., 1995).

La structuration en modules de pilotage implique une grande quantité d'informations circulant au travers de cette architecture. Un mécanisme de communication permettant d'exploiter justement l'information qu'elle véhicule doit donc être mis en place. Le principe généralement

retenu est communément qualifié de RPC (Remote Procedure Call), traduit généralement par l'expression française : communication en mode appel/réponse. Chaque module envoie des ordres aux modules qui se trouvent dans le niveau immédiatement inférieur (*requêtes de commande*). Ces requêtes sont désagrégées, et envoyées aux niveaux inférieurs.

Généralement, lorsqu'un module *i* envoie une requête de commande, il attend un compte-rendu attestant que la requête a bien été effectuée par le module du niveau i-1. Néanmoins, en présence de dysfonctionnements, un module peut remettre en cause les ordres reçus de son niveau supérieur. Dans ce cas, le principe de fonctionnement de la hiérarchie doit imposer que seul le module qui a émis la requête soit en mesure de pouvoir la modifier. Si tel n'est pas le cas, le niveau *i-1* risque de violer des contraintes inconnues pour lui (Combacau, 1991).

Les principes généraux inhérents aux architectures de pilotage ayant été présentés, et considérant que les modules de pilotage assurant la coordination de plusieurs chaînes fonctionnelles sont précisément le niveau auquel cette thèse vise à apporter sa contribution, prenons le temps de les détailler.

2.5 Les chaînes fonctionnelles

Une chaîne fonctionnelle (Gendreau et al., 2000) se caractérise par un agencement fonctionnel de constituants sous forme de chaînes, en regroupant tous les éléments de la partie opérative, de l'interface et du système de pilotage concourant à la réalisation d'une fonction opérative élémentaire. Elle se compose en général de trois parties, voir Figure 1.2.

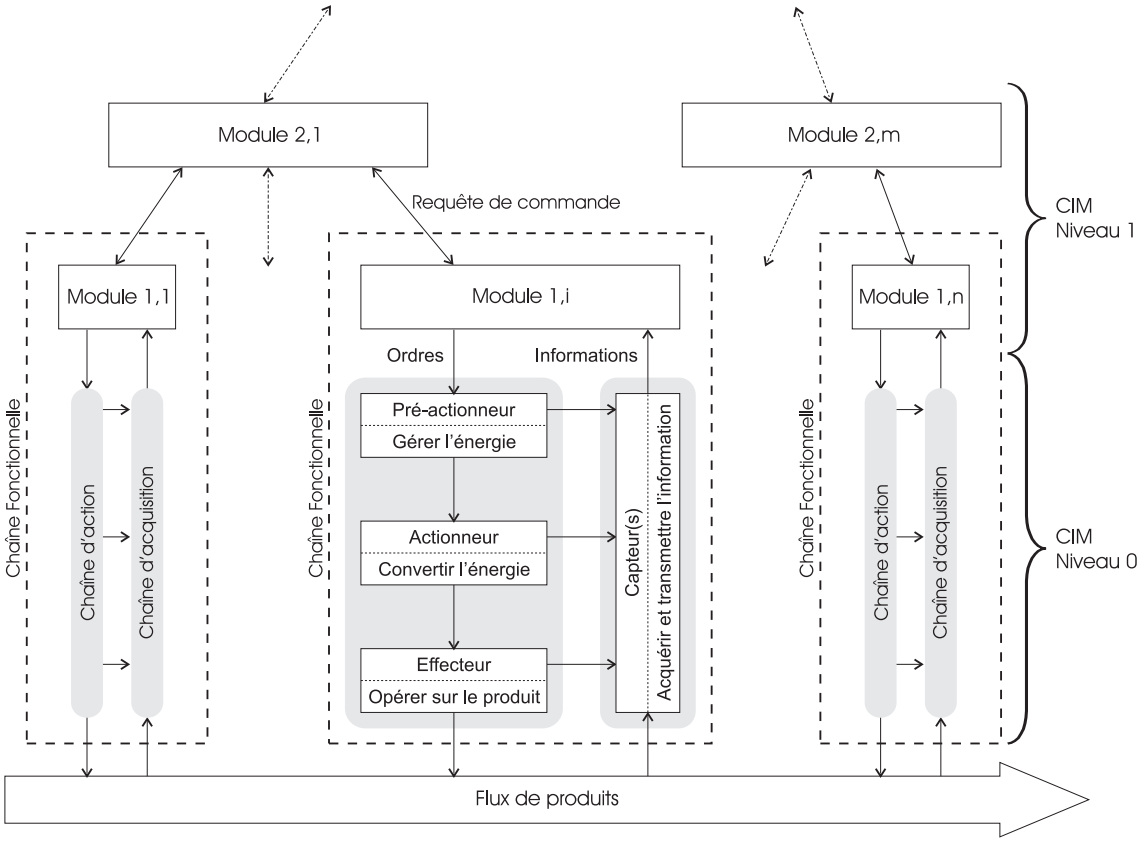


FIG. 1.2 – Structure de coordination des chaînes fonctionnelles.

La chaîne d'action : elle assure, à partir d'un ordre du système de pilotage, l'animation de la partie opérative. Elle se compose de préactionneur(s), d'actionneur(s) et d'effecteur(s). L'ensemble des chaînes d'actions constitue avec le flux de produits le système contrôlé du point de vue du système de pilotage.

La chaîne d'acquisition : elle regroupe l'ensemble des composants permettant d'acquérir, d'élaborer et de transmettre les informations exploitables par le système de pilotage. Il peut s'agir d'une chaîne d'acquisition d'informations sur le flux de produits ou sur la chaîne d'actions, ou bien d'informations liées à l'environnement de la chaîne fonctionnelle considérée.

Le module de pilotage : il appartient au plus bas niveau du système de pilotage. Il élabore des ordres à destination de la chaîne d'actions en fonction des objectifs fixés par le module de niveau supérieur et des informations fournies par la chaîne d'acquisition.

Une chaîne fonctionnelle peut être associée à deux ou plusieurs actions (exemples : "sortir tige" ou "rentrer tige" pour un vérin, "marche avant", "marche arrière", "tourner d'un quart de tour vers l'avant" pour un moteur à deux sens de marche). Elle peut comporter plusieurs capteurs (deux capteurs de fin de course par exemple) ou encore n'en comporter aucun. Elle se réduit alors à la chaîne d'actions. De même, elle peut comporter plusieurs actionneurs (exemple : deux vérins de levage symétriques pour une forte charge) ou encore n'en comporter aucun. Elle se réduit alors à la chaîne d'acquisition (exemple : fonction de pesée basée sur une jauge de contrainte). Ainsi, en fonction des éléments composant une chaîne fonctionnelle, une requête ou/et un compte rendu, respectivement envoyée et reçu par le niveau coordination, seront accompagnés de données. C'est par exemple, pour une chaîne fonctionnelle limitée à une chaîne d'acquisition, les données recueillies. Pour une chaîne d'action, les données seront, par exemple, la consigne de position dans le cas d'un vérin électrique. La Figure 1.3 représente les quatre situations existantes.

De cette notion de chaîne fonctionnelle, le niveau 1 du CIM peut se décomposer au minimum en deux sous-niveaux d'un point de vue décisionnel. En effet, d'après sa définition, le niveau 1 du système de pilotage doit piloter d'une part chacune des chaînes fonctionnelles du SAP, et d'autre part coordonner plusieurs chaînes fonctionnelles (Figure 1.2).
L'architecture opérationnelle étant désormais présentée, nous allons maintenant nous intéresser à sa configuration nominale afin de réaliser un objectif de production.

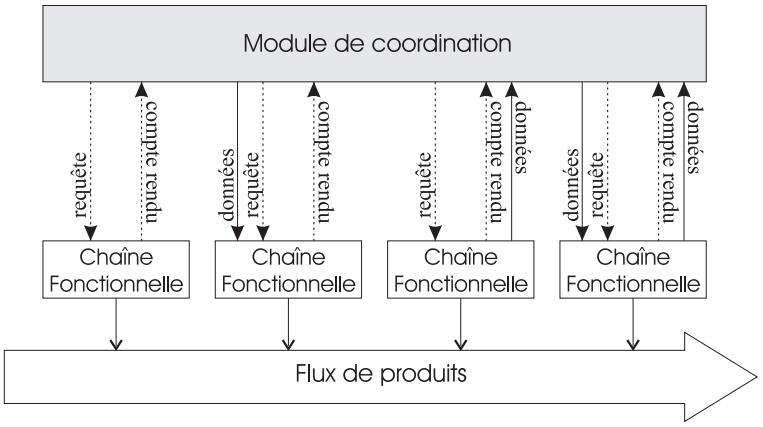


FIG. 1.3 – Communication entre un module de coordination et une chaîne fonctionnelle.

2.6 Configuration du système de pilotage

Comme nous avons pu le constater dans les sections précédentes, l'architecture de la partie opérative et du système de pilotage offre suffisamment de flexibilité pour envisager de répondre à différentes sollicitations clients définissant les objectifs de production. Nous allons donc nous intéresser maintenant au processus qui mène à une configuration logicielle du système de pilotage afin que le SAP réponde à la demande client.

Jusqu'à aujourd'hui, le processus de configuration qui décrit finalement le comportement adéquat que doit avoir le SAP commence par un long processus de négociation entre les divers acteurs du CIM, du client jusqu'au niveau de coordination. Tout au long de ce processus, il n'y a bien entendu aucune fabrication effective de produits mais uniquement un travail visant à s'assurer d'une part que le SAP à les capacités de réaliser la demande tant d'un point de vue technique que temporel, et d'autre part de mettre au point les recettes de production. Pour le niveau de coordination, il s'agira par exemple de concevoir les programmes des automates dans un des langages de la norme IEC 61131-3. A ce niveau et contrairement au niveau supérieur du modèle CIM (cf. Figure 1.4), la phase de configuration n'est pas automatisée et elle est essentiellement basée sur l'expertise humaine.

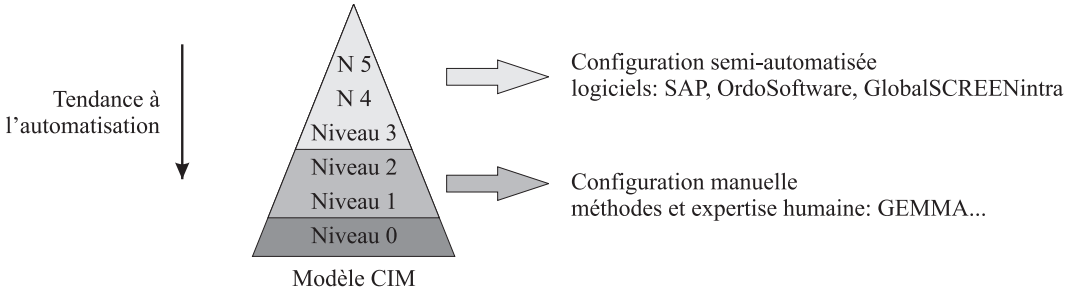


FIG. 1.4 – Tendance à l'automatisation du processus de configuration.

Toute la difficulté d'une telle configuration résulte de l'interaction de trois domaines d'expertises et de connaissances tels qu'exprimés dans la Figure 1.5 adaptée de (Nagel et Tomiyama, 2004). Elle représente les interactions entre les trois cycles de vie : produit, process, système de pilotage. Ces cycles de vie se composent des étapes d'identification et de spécification du Besoin utilisateur (B), de Conception (C) avec la spécification du point de vue concepteur, de Réalisation (R), d'Utilisation (U) et de Fin de vie (F).

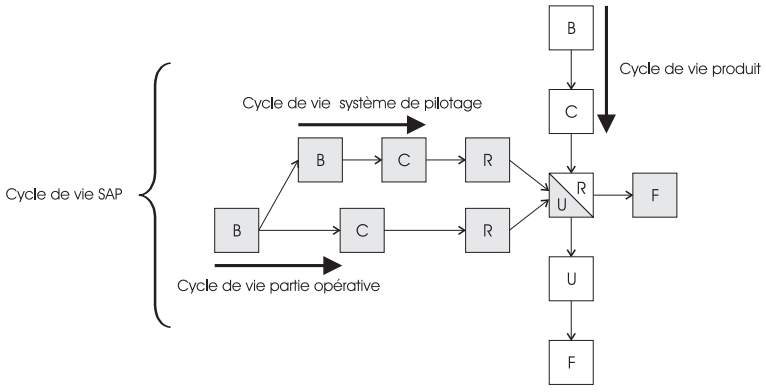


FIG. 1.5 – Interaction des cycles de vie produits, partie opérative et système de pilotage.

Les interactions entre le produit, la partie opérative et le système de pilotage sont aujourd'hui telles que les choix faits à chacune des étapes de conception peuvent avoir un impact fort sur les autres. En réponse à cet enjeu, l'ingénierie simultanée, qui s'est d'abord focalisée sur la conception des produits et de la partie opérative, a aujourd'hui une très forte tendance à inclure aussi ceux du système pilotage. DELMIA Automation, une filiale de Dassault Système, leader mondial de la conception assistée par ordinateur, propose la validation complète des programmes des automates programmables industriels dans un environnement 3D virtuel connecté aux outils de conception des produits et de la partie opérative. Nous pouvons également citer le logiciel SIMSED qui est un logiciel de simulation des systèmes à événements discrets conçu par la société SYDEL en partenariat avec le LESTER[1]. Il s'agit d'un logiciel de conception et de simulation des systèmes de transitique incluant un moteur physique.

Malheureusement, il n'existe pas aujourd'hui de modèle de connaissance reconnu permettant une réelle connexion entre la conception produit/partie opérative et la conception du système de pilotage.

Afin d'assurer au maximum la continuité du service offert par un SAP, le système de pilotage doit être réactif aux aléas de fonctionnement. L'obtention de cette réactivité du système de pilotage est présentée dans la section suivante.

3 La Réactivité aux Aléas de Fonctionnement

La considération d'un système réel ne permet pas de faire abstraction des aléas de fonctionnement. En effet, lorsqu'une requête de commande est envoyée à une chaîne fonctionnelle, rien ne garantit que l'évolution des états (parties opératives / produits) observés correspondra systématiquement à l'évolution prévue par le système de pilotage. Afin de rendre ce système capable de réagir face à de telles situations, il doit intégrer en sus de la fonction Commande, des fonctions de Supervision et de Surveillance (Combacau et al., 2000). Cependant, avant de présenter de manière plus approfondie ces fonctionnalités, attardons nous un moment sur le concept d'aléas de fonctionnement.

3.1 Les Aléas de Fonctionnement

Un aléa de fonctionnement se définit comme un événement qui vient perturber le fonctionnement du SAP en remettant donc en cause la réalisation de son objectif. Bien entendu, face à une telle situation, il est naturel d'attendre du système automatisé une certaine réactivité visant à réduire l'erreur constatée et donc à maintenir l'objectif de production fixé. Cependant, comme nous allons le voir dans la suite de ce paragraphe, ce comportement réactif présuppose de doter le système de pilotage de fonctionnalités plus ou moins complexes, capables de traiter l'imprévu, avec *confiance*. Pour s'en convaincre, étudions les cinq types d'aléas qui peuvent être mis en exergue selon leur localisation dans la structure générale du SAP (cf. Figure 1.1) ; nous identifions en effet :

1. ceux qui affectent la partie opérative (casse d'outils, de courroies, etc...) ;

[1] Laboratoire d'Électronique des Systèmes TEmps Réel

2. ceux qui pénalisent l'interface (défaillance[2] capteur, actionneur, pré-actionneur) ;
3. ceux qui touchent la partie matérielle du système de pilotage (panne d'alimentation du calculateur, saturation de la bande passante du réseau de communication, ...) ;
4. ceux qui perturbent les entrées du SAP (variation des demandes client, coupure alimentation générale, non conformité des propriétés de la matière première) ;
5. et enfin ceux qui remettent en cause la cohérence de la partie logicielle du système de pilotage.

S'appuyant sur des prélèvements de données issues du monde industriel (Sourise et Boudillon, 1997), la Figure 1.6 donne une répartition assez fidèle des aléas de type 1, 2 et 3.

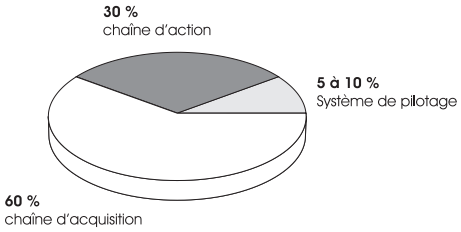


FIG. 1.6 – Origine des aléas de type 1, 2 et 3, adapté de (Sourise et Boudillon, 1997).

Parmi les aléas de type 4, l'impact de la variation de la demande est particulier. En effet, ces variations concernent aussi bien les quantités, les délais et les spécifications du produit. Du point de vue des quantités et des délais, l'impact se limite généralement aux niveaux supérieurs (3, 4 et 5) du modèle CIM. Mais pour la personnalisation des produits, l'impact touche tous les niveaux. La modification des spécifications produit engendre quasiment toujours une remise en cause des lois de commande définies aux niveaux 1 et 2.

Les aléas de type 5 sont liés aux parties logicielles qui ont des particularités les différenciant des composants matériels. En effet, les logiciels sont immatériels, facilement modifiables notamment par des agents de maintenance. De plus, ils ne connaissent pas les phénomènes de vieillissement et d'usure. Les défauts logiciels sont donc toujours d'origine humaine et sont introduits tout au long du cycle de vie du logiciel (cf. Figure 1.7). Les moyens pour atteindre la sûreté logiciel se focalisent donc uniquement sur le développement de méthodes et de démarches de spécification et de conception de logiciels (Laprie, 1995), (Frey et Litz, 2000).

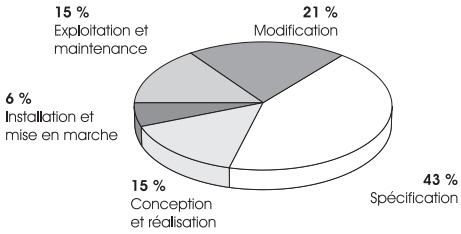


FIG. 1.7 – Origine des défaillances/Cycle de vie logiciel, issu de (Sourise et Boudillon, 1997).

[2]Une défaillance est par définition la cessation de l'aptitude à réaliser une fonction (Zwingelstein, 1999) (Villemeur, 1988)

Nous retiendrons enfin que la proportion d'aléas de fonctionnement qui affectent la partie matérielle du système de pilotage (automates, réseaux de terrain, réseaux d'atelier, ...) est relativement faible (entre 5 et 10 %). Ces éléments, testés et répondant à des normes drastiques (certification classe A, indices de protection, norme DIN 19245 relative aux réseaux de terrain [http ://www2.din.de],...) sont quant à eux très fiables.

Cependant, dans un tel contexte et en faisant raisonnablement abstraction des défaillances de la partie matérielle et logicielle du système de pilotage, ces situations doivent conduire, au moins à deux types de réactivités de la part du système de pilotage :

Réactivité aux aléas qui affectent les capacités nominales de la partie opérative ou de l'interface : deux stratégies peuvent être considérées ici. D'une part, si les délais de réparation sont incompatibles avec les délais de livraison, une reconfiguration de la partie logicielle en exploitant, si elle existe, la flexibilité native de la partie opérative (utilisation des services encore effectifs offerts par la PO) peut être envisagée. D'autre part, si les délais de réparation sont acceptables, la réparation de l'élément défaillant peut être demandée avec *in fine* reprise du cycle. Cependant, quelle que soit la stratégie considérée, il n'en demeure pas moins que pour répondre aux questions "quand réagir", "quoi réparer" ou encore "que reconfigurer", il est nécessaire au moins de détecter la déviation de comportement, de localiser l'origine exacte de la défaillance, d'évaluer les services encore offerts par la partie opérative ou l'interface, ou encore d'être en mesure de concevoir une nouvelle configuration de la partie logicielle du système de pilotage (élaboration de nouvelles lois de commande).

Réactivité aux aléas qui affectent les demandes client : dans ce cas, comme nous avons pu le voir plus haut, la seule réactivité envisageable consiste à reconsidérer toute ou partie de la configuration logicielle du système de pilotage, à savoir donc, générer de nouvelles lois de commande adaptées aux nouvelles spécifications.

Ainsi, pour être réactif aux aléas venant d'être présentés, que ce soit pour des raisons matérielles ou de variations des demandes, le système de pilotage doit disposer d'un sous-ensemble de fonctionnalités capables de lui conférer, outre ses capacités de commande, des capacités d'observation, d'analyse, de décision et de génération de lois de commande. Afin de préciser davantage le contexte dans lequel ces travaux se placent, la section suivante propose de préciser les fonctionnalités de supervision et de surveillance qui doivent être intégrées au cœur du système de pilotage afin d'améliorer sa réactivité aux aléas de fonctionnement.

3.2 Les fonctions de Supervision, Surveillance et Commande (SSC)

La présentation proposée des fonctions de SSC dans cette section est entièrement basée sur une étude terminologique (Combacau et al., 2000) (Niel et Craye, 2002) issue du GT ASSF [3] du GRP [4] devenu aujourd'hui le GT INCOS [5] du pôle STP [6] du GDR MACS [7].

Fonction de commande : son rôle est de faire exécuter un ensemble d'opérations (élémentaires ou non suivant le niveau d'abstraction auquel on se place) au système contrôlé (interface, partie opérative et flux de produits) en envoyant des ordres en réponse aux

[3] Groupe de Travail Automatisation des Systèmes Sûrs de Fonctionnement
[4] Groupement de Recherche en Productique
[5] Groupe de Travail INgénierie de la COmmande et de la Supervision des SED
[6] Science et Technique de la Production
[7] Groupement de Recherche, Modélisation, Analyse et Conduite des Systèmes dynamiques

demandes (requêtes de niveau supérieur dans le contexte d'une architecture hiérarchique et modulaire). Dans cette définition, la commande regroupe toutes les fonctions qui agissent directement sur le système contrôlé. Il est habituel de distinguer quatre fonctions de commande : commande dans un but de production, reprise pour un retour en production, urgence, et enfin maintenance corrective pour sa partie automatisée. Ces fonctions ont toutes le même besoin qui est de disposer d'une loi de commande définissant les opérations à exécuter en fonction des requêtes de commande, de l'état de la partie opérative et du flux de produits.

Fonctions de surveillance : son rôle est de recueillir en permanence tous les signaux en provenance du système contrôlé (chaînes fonctionnelles et flux de produits) et de la commande. De plus, elle reconstitue l'état réel du système contrôlé et fait toutes les inférences nécessaires pour produire les données utilisées afin de dresser des historiques de fonctionnement et le cas échéant, pour mettre en œuvre un processus de traitement de défaillance. Dans cette définition, la surveillance n'agit ni sur le système contrôlé ni sur la commande. Elle a donc un rôle passif vis-à-vis de ces deux systèmes. Elle regroupe généralement les fonctions : suivi, détection, diagnostic, pronostic. La majorité des fonctions de surveillance fait appel à des mécanismes de prise de décision comme par exemple le diagnostic (Kempowsky et al., 2004) lorsque plusieurs causes d'une défaillance sont plausibles.

Fonctions de supervision : son rôle est de contrôler et de surveiller l'exécution d'une opération ou d'un travail effectué par d'autres. La supervision recouvre les aspects fonctionnement normal et anormal suite à un aléa. Par conséquent, elle est chargée de gérer suivant le contexte les moyens à mettre en œuvre en terme de fonctions de SSC, de générer et de paramétrer les lois de commande par conception ou extraction d'une base de recettes. Dans le cadre de leur élaboration, et en fonction du niveau de réactivité requis, fixé essentiellement par le contexte considéré (nouvelle demande client, défaillance de la partie opérative), ces lois de commande sont générées (cf. § 2.6) en prenant en compte tout ou partie des critères liés à la productivité ou à la qualité. En effet, durant la phase de configuration des bases de recettes, la contrainte temporelle n'est pas aussi forte qu'en phase d'exploitation !

Les résultats de l'étude terminologique ayant été brièvement présentés, un dernier point doit être précisé quant aux solutions de mise en œuvre de ces fonctions. Deux au moins sont envisageables : mise en œuvre intégrée ou séparée. Une approche entièrement intégrée n'est pas réactive aux aléas non prévus. La contre-partie de la réactivité d'une approche complètement séparée est un temps de réaction indéterminé. Une solution consiste à jouer sur les avantages des deux approches précédentes. On parlera alors de "surveillance-supervision mixte" (Combacau, 1991). Ici, la commande intègre en partie la fonction détection alors que les autres fonctions de surveillance-supervision sont séparées de la commande.

Dans le cadre de ces travaux, nous retiendrons cette dernière hypothèse de travail. Les conséquences directes d'une telle prise de position impliquent naturellement que toutes les lois de commande correspondant à toutes les situations ne peuvent pas être définies à l'avance. Aussi, l'approche présentée dans ce mémoire vise à apporter sa contribution à l'élaboration de fonctions de supervision ayant la capacité de concevoir, au besoin, de nouvelles lois de commande.

Les aspects essentiels permettant de cadrer le contexte de notre étude sur le plan de la réactivité aux aléas de fonctionnement ayant été présentés, nous nous proposons maintenant de

localiser précisément nos apports au sein de l'approche globale de Supervision, Surveillance et Commande développée au Laboratoire d'Automatique de Grenoble.

4 L'approche développée au LAG

Les approches existantes de SSC sont nombreuses d'une part au niveau national avec celles proposées par exemple au LAGIS (Dangoumau, 2000) (Berruet et al., 2000), au LAI (Rezg, 1996) (Khatab, 2000), au CRAN (Gouyon, 2004), au LESTER (Mouchard, 2002), au LAAS (Chaillet, 1995) (Zamaï, 1997), et au LAG (Mendez, 2002), et d'autre part au niveau international avec des approches basées parfois sur des systèmes holoniques (Fletcher et al., 2003) ou multi-agents (Culita, 2001) (Heragu et al., 2002) (Odrey et Mejia, 2003). Les travaux présentés dans ce mémoire ont été développés dans un cadre générique. Mais, ils seront en premier lieu intégrés à l'approche de SSC développée au LAG. Pour cette raison, nous présentons ici dans le détail cette dernière.

L'approche développée actuellement au LAG a été initiée au LAAS. La réactivité de l'approche est non seulement basée sur les fonctions de SSC présentées précédemment mais aussi sur l'existence de marges temporelles (Combacau, 1991) offertes à chacun des niveaux de pilotage. Sans ces marges temporelles, une défaillance viendrait rendre impossible le respect des délais de fabrication et remettre en cause toutes les décisions prises à tous les niveaux du système de pilotage.

"Dans cette approche de SSC, la surveillance-supervision n'est plus vue seulement comme un palliatif à la commande" (Zamaï, 1997), mais elle est considérée au même niveau que les autres fonctions. Dans le contexte d'une architecture hiérarchique et modulaire de pilotage présentée au § 2.4, l'approche est basée sur un module générique du point de vue de ses fonctions et de leur gestion ; ce module est appelé CERBERE. Le besoin important de gestion de l'information et d'une structuration des fonctions de SSC autour de cette dernière a été démontré dans une étude menée dans le cadre des travaux de E. Zamaï (Zamaï et al., 1997). La conséquence de cette étude est une structure interne d'un module CERBERE qui est basée sur deux blocs fonctionnels : un bloc de routage de l'information et un bloc de traitement de l'information, présentés dans la Figure 1.8.

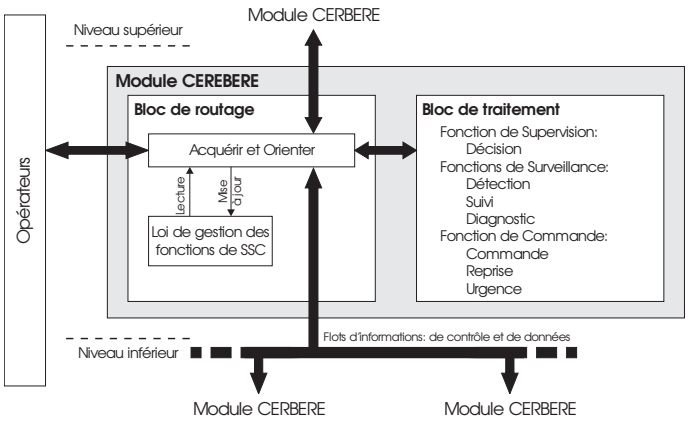


FIG. 1.8 – Représentation de la structure interne d'un module CERBERE du système de pilotage.

Le bloc de traitement contient l'ensemble des fonctions de SSC destinées au traitement de l'information. Nous y retrouvons une fonction de supervision, quatre fonctions de surveillance et trois fonctions de commande.

Le bloc de routage est chargé de l'acquisition et de l'orientation de l'information transitant par un module. Il est le seul à être connecté avec l'extérieur (opérateur humain, module du niveau supérieur ou du niveau inférieur). Le bloc de routage dispose d'un modèle de gestion des fonctions de SSC afin d'orienter l'information suivant le contexte (requêtes de commande, état de la partie opérative et du flux de produits). Ce modèle est formalisé par un modèle à états, où un état représente les fonctions qui sont en cours d'exécution et donc susceptibles d'être concernées par l'information à orienter. L'obtention de ce modèle est réalisée par une technique de synthèse à partir d'un modèle de référence (97 états et plus de 1000 transitions) (Mendez, 2002).

Dans le cadre de cette thèse, comme nous l'avons laissé entendre précédemment, nous allons nous intéresser plus particulièrement à la configuration et à la reconfiguration du système de commande afin d'améliorer la réactivé du SAP aux aléas de fonctionnement. Afin de spécifier la forme des lois de commande attendues par le système de commande, le contexte d'exécution de ces lois de commande à générer est maintenant présenté.

5 Fermeture de contexte

Il s'agit ici de présenter de manière plus fine le contexte de l'étude qui nous permettra entre autre d'orienter l'étude bibliographique et la description précise de notre problématique. Ce contexte est caractérisé à partir d'une analyse décisionnelle. Mener une telle étude présuppose de disposer d'un référentiel. Comme nous l'avons déjà précisé, nos travaux s'intégreront, dans un premier temps, au sein de l'approche CERBERE. Ainsi, nous nous proposons d'appliquer ce référentiel d'analyse à cette approche afin d'établir le périmètre exact de notre intervention. Il sera alors naturellement dégagé de cette étude toutes les hypothèses concernant l'environnement de notre travail.

5.1 Analyse décisionnelle

Comme nous avons pu le préciser dans le début de ce chapitre, nos travaux tentent d'apporter leur contribution à la configuration et à la reconfiguration du système de commande. Au delà de la composante matérielle qui le caractérise (type de calculateur, cartes d'entrées/sorties, ...), nous nous proposons ici de fermer le contexte de la partie décisionnelle du système de pilotage qui lui confère des capacités de conception ou de sélection de lois de commande.

Dans cet objectif, nous nous proposons de reprendre ici les résultats d'une analyse décisionnelle des systèmes de Supervision Surveillance et Commande présentés dans (Henry et al., 2003). Ces travaux, initialement développés dans le but de mettre en exergue toutes les formes de décisions au sein d'approches de SSC (pour la détection (Boufaied, 2003), le diagnostic (Zwingelstein, 1995), la commande, etc...), proposent un référentiel d'étude mettant en évidence deux moteurs génériques sur lesquels sont basés les fonctions de SSC (cf. Figure 1.9) :

- **le moteur d'exécution :** la première classe de fonctions limitées à une tâche d'exécution doit disposer d'un modèle déterministe (absence de conflit décisionnel). Les sorties d'une telle fonction sont données par ce modèle suivant son état courant et ses entrées. C'est le

cas des fonctions de commande de l'approche CERBERE considérées dans les travaux de (Mendez, 2002).

– **le moteur de décision :** à l'inverse, le modèle dont disposent les autres fonctions est dit non-déterministe au sens où il existe des conflits décisionnels à résoudre. C'est le cas par exemple lors de la résolution de problèmes d'ordonnancement d'ateliers basés sur une modélisation par réseau de Petri (Julia et al., 1998) ; plusieurs transitions sont franchissables en sortie d'une place. Dans cette situation, en plus de ce modèle, la fonction doit disposer d'un ou plusieurs critères afin de décider.

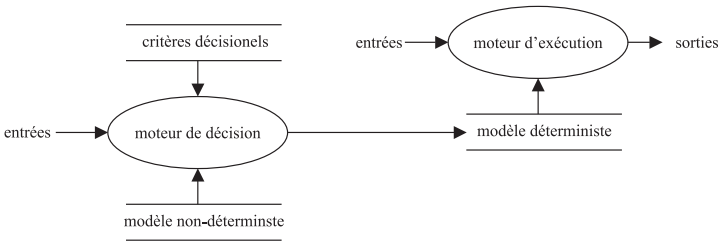


FIG. 1.9 – Référentiel pour l'analyse décisionnelle d'une approche de SSC.

Directement appliquée à un module de pilotage de l'approche CERBERE (Mendez, 2002), et en particulier à ses fonctions de supervision et de commande, la structure fonctionnelle représentée dans la Figure 1.10 est obtenue.

Une telle analyse fondée sur le point de vue décisionnel conduit naturellement à distribuer des rôles très précis à chacun des acteurs du système.

En particulier, ici, seule la fonction décision est apte à prendre toutes les décisions requises pour, à partir de critères (qualité, productivité) et de contraintes (modèles de la partie opérative représentant en particulier les contraintes de sécurité et d'écologie) générer ou sélectionner un modèle de commande, de reprise, ou d'urgence déterministe. Chacun de ces modèles pouvant ensuite être exécuté par l'une des fonctions de commande, reprise ou urgence.

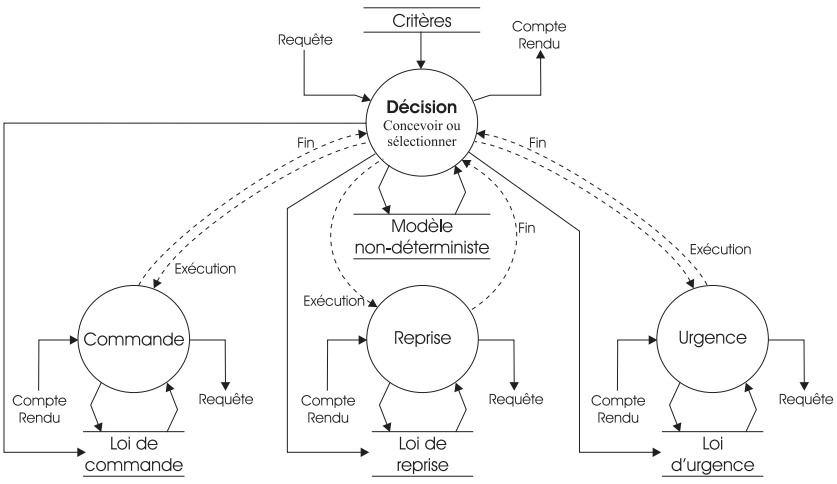


FIG. 1.10 – Analyse décisionnelle appliquée à l'approche CERBERE.

5.2 Conséquences sur l'architecture CERBERE

Une telle analyse nous amène ici à reconsidérer au moins trois points de l'approche CERBERE.

En premier lieu, compte tenu du fait que les fonctions de commande, de reprise ou d'urgence sont d'une part réduites à un rôle unique de moteur d'exécution, et que d'autre part, elles ne doivent en aucun cas générer d'ordres contradictoires vers le sous-système contrôlé (Zamaï, 1997) afin d'éviter d'éventuels conflits opérationnels, nous proposons de les factoriser en une seule fonction. Cette dernière sera appelée **commander**, au sens terminologique du terme (Combacau et al., 2000), et exécutera, au besoin, l'un des modèles déterministe de commande, de reprise ou d'urgence fourni par la fonction décision. Un mécanisme de commutation de modèles, déclenché par la fonction décision, devra donc sensibiliser cette fonction de commande afin de lui permettre de changer de modèle à exécuter. Notons également qu'une telle considération s'accompagne de la simplification des trois classes de requêtes de commande, d'urgence ou de reprise en une seule requête de commande.

En second lieu, et faisant suite à cette première reconsidération, une mise à jour du modèle de référence pour la surveillance, la commande et la supervision doit être envisagée. En effet, les 97 états (Mendez, 2002) du modèle s'appuient sur l'existence des fonctions de commande, reprise ou urgence. La simplification proposée entraîne donc une réduction du modèle. Cependant, il est important de remarquer que cette réduction n'implique pas une perte d'information. En effet, la connaissance de l'exécution d'un modèle de commande, d'un modèle d'urgence ou encore de reprise est rendue implicite par la prise en compte actuelle des modes de marches et d'arrêts (ADEPA, 1981) (Mendez et al., 2003) (en particulier ceux caractérisant un fonctionnement normal, une marche dégradée ou encore l'urgence) au sein du modèle de référence.

Enfin, et parce que nous avons montré que quel que soit le type d'aléa (variation de la demande client ou défaillance de la partie opérative) le système de pilotage devait être en mesure de générer de nouvelles lois de commande, une modification capitale doit être apportée à l'algorithme de routage des informations. En effet, l'approche développée successivement dans (Zamaï, 1997) puis (Mendez, 2002) supposait un fonctionnement dans un contexte unique d'exploitation. Ceci supposait que tous les modèles de commande en fonctionnement normal étaient préalablement générés et affectés, en manuel, à chaque module de pilotage. De ce fait, et en exploitation normale, une requête de commande issue d'un module père était naturellement dirigée vers la fonction de commande du module fils. Dans notre cas, et parce que la fonction décision doit offrir des capacités de génération ou de sélection de lois de commande, que ce soit en contexte de configuration (hors ligne), ou de reconfiguration (en ligne), un nouveau type de requête devra être considéré en sus des classiques requêtes de commande ; il s'agit des **requêtes de configuration**. Ainsi, en phase de configuration du SAP, l'occurrence d'une demande client entraînera en premier lieu sa désagrégation au sein de la structure hiérarchique et modulaire de pilotage en requêtes de configuration générées par les fonctions décisions de chaque module de pilotage concerné. Une fois entièrement configuré (remontée des comptes rendus de fin de configuration), le système de pilotage pourra, à la date prévue, lancer les requêtes de commande. Ainsi, que ce soit une requête de configuration (hors ligne) ou bien de commande (en ligne) l'algorithme de routage de tout module de pilotage doit l'orienter systématiquement vers la

fonction décision, qui se chargera de lancer soit une activité de génération de loi de commande (répondant donc à une requête de configuration), ou bien une activité de sélection d'un modèle de commande (répondant ici à une requête de commande) dans la base de recettes élaborées en phase de configuration.

Dans le cadre de ces travaux, nous nous focaliserons sur les capacités de la fonction décision à générer de nouvelles lois de commande, de reprise, voir même d'urgence en réaction aux demandes client et aux aléas de fonctionnement de la partie opérative. Cependant, et comme nous allons le voir dans la section suivante, il ne serait pas judicieux d'aller plus loin dans l'exposé de ces travaux sans au préalable avoir défini le contexte d'utilisation des lois de commande que nous souhaitons automatiquement générer.

5.3 Mise en œuvre de la fonction commande

Si nous restreignons l'étude de la génération de lois de commande dans un objectif unique d'exécution, il serait assez lucide d'envisager, à terme, une compilation de ces lois et ce malgré le contexte de commutation fréquente de modèles dans lequel nous nous plaçons (commutation liée aux aléas de fonctionnement). En effet, un programme de commande écrit dans un langage dit "compilé" va être traduit par un programme annexe (le compilateur) afin de générer un nouveau fichier qui sera lui autonome ; on dit d'ailleurs que ce fichier est exécutable. Compte tenu de la puissance actuelle des calculateurs, de la simplicité de compilation (http ://www-verimag.imag.fr/) des lois de commande de chaînes fonctionnelles, et enfin du niveau auquel ces travaux se placent (coordination des chaînes fonctionnelles) pour lequel les contraintes de temps de réponse ne sont pas aussi dures qu'au niveau des boucles de régulation, l'insertion d'une fonction de compilation entre la phase de génération de lois de commande et son exécution par la fonction commande semble tout à fait réaliste. Enfin, il faut remarquer que la traduction étant faite dans un langage proche de celui de la machine, son exécution est plus rapide améliorant ainsi la caractéristique temps réel du système de commande.

Cependant, il ne serait pas judicieux de prendre en compte que cet unique point de vue. En effet, nos travaux s'intégrant dans une approche de Supervision, de Surveillance et de Commande, il est aisé de démontrer que les lois de commande peuvent être amenées à être utilisées à d'autres fins que la commande (Chaillet, 1995). Prenons le cas par exemple de la fonction diagnostic. Suite à l'occurrence d'un symptôme de défaillance, le rôle du système de diagnostic revient à rechercher la (les) cause(s) ayant entraîné(es) la défaillance. Pour cela, il doit avoir au moins la connaissance non seulement de l'état réel de la partie opérative, mais également celui de la séquence de commande ayant abouti à l'anomalie. Si une telle loi de commande venait à être compilée (donc dans un langage exclusivement compréhensible par la machine), il est clair que cela ne simplifierait pas l'accès à cette connaissance pour le système de diagnostic. Si de plus nous admettons qu'un tel système de diagnostic ne peut être entièrement automatisé, et donc doit naturellement collaborer avec l'opérateur humain, il devient alors évident qu'une technique de compilation de la connaissance doit être, si possible, rejetée.

Compte tenu de ces observations, une mise en œuvre interprétée du système de commande semble être davantage appropriée aux besoins d'une approche globale de SSC. Il en découle la réalisation d'un interpréteur de commande dont le rôle sera de traduire au fur et à mesure

les instructions de commande. L'interpréteur sera quant à lui compilé une seule fois pour le calculateur cible considéré.

La section suivante conclue ce chapitre suite à la présentation du contexte d'utilisation des lois de commande à générer.

6 Conclusion

Dans ce chapitre, nous avons présenté la problématique générale du pilotage des Systèmes Automatisés de Production. Nous avons successivement précisé leur rôle, leur structure organisationelle (celle du CIM) et précisé, au sein de cette architecture, le niveau dans lequel se placent ces travaux, à savoir, la coordination des chaînes fonctionnelles. Dans ce contexte, où nous avons précisé les difficultés liées à la configuration (hors ligne donc) des parties logicielles des systèmes de commande (intersection de différentes disciplines et connaissances), nous avons ensuite mené une étude de l'impact d'éventuels aléas de fonctionnement sur les modules de coordination de chaînes fonctionnelles. De cette étude, fortement appuyée sur les résultats du Groupe de Travail ASSF du GDR-MACS, nous avons présenté un bilan fonctionnel des compétences que devait intégrer tout module de pilotage réactif. Fort de cette étude, nous avons ensuite mené une analyse critique de l'approche dans laquelle nous allons apporter notre contribution. Cette approche, développée au Laboratoire d'Automatique de Grenoble, et baptisée CERBERE, vise à développer un module de pilotage réactif, intégrant à terme, toutes les fonctionnalités requises pour piloter une partie opérative que ce soit en fonctionnement normal ou anormal. Les résultats de l'analyse critique effectuée nous ont permis de proposer des améliorations significatives de la structure décisionnelle existante, permettant maintenant d'envisager plus sereinement l'intégration de fonctions capables de spécifier, au besoin, de nouveaux comportements du système de commande.

Le chapitre suivant se propose alors de mener une étude bibliographique largement orientée sur les aptitudes de certaines approches à générer des comportements, de la partie opérative et du flux de produits, adaptés non seulement au besoin de l'utilisateur mais également à la capacité des systèmes contrôlés.

Chapitre 2

État de l'Art

1 Introduction

Dans le chapitre précédent, nous avons été amenés à préciser le périmètre de nos travaux de recherche. Ils se localisent en particulier au niveau des modules de coordination de chaînes fonctionnelles et visent à apporter leurs contributions à la spécification des comportements du système de commande considéré. Nous parlerons désormais de lois de commande.

Fort de ce contexte, nous nous proposons maintenant de dresser un état de l'art du domaine. Nous réalisons ainsi, dans ce chapitre, une étude des principales approches offrant des aptitudes certaines à la conception de lois de commande. Cinq approches sont ainsi présentées : la planification en intelligence artificielle, l'ordonnancement, la génération de gamme opératoire, les approches basées sur la théorie de Ramadge et Wonham, et pour terminer la démarche actuellement appliquée dans l'industrie. Comme nous le verrons dans les sections suivantes, bien que différentes de par leur domaine d'application et donc des verrous technologiques induits, toutes ces approches s'accordent sur au moins trois points fondamentaux : le besoin d'un modèle du système contrôlé, la spécification formelle d'une demande à laquelle répondre, la conception d'un algorithme, plus ou moins complexe, capable de générer une solution, si possible, optimale.

2 La Planification en Intelligence Artificielle

L'intelligence artificielle (Luger, 2002) est avant tout un domaine de recherche faisant appel à des techniques informatiques et à des recherches développées en sciences humaines (de Boulay, 2001). La recherche en intelligence artificielle avait au départ comme objectif de comprendre le raisonnement humain ou d'un individu en simulant ce raisonnement sur un ordinateur. Par la suite, un deuxième objectif a été précisé, qui consiste à développer des systèmes intelligents (Nilson, 1998). La différence principale entre un système d'intelligence artificielle et un système informatique classique est que la connaissance est codée explicitement et reste accessible. Les domaines d'applications de l'intelligence artificielle sont très nombreux : le diagnostic, l'aide à la décision, la robotique, la représentation et la capitalisation des connaissances.

De ces domaines, une branche particulière de l'intelligence artificielle a émergé, la planification automatique (Ghallab et al., 2004). Elle concerne la synthèse de plans d'opérations par des techniques de modélisation, de conception et de programmation des processus de sélection des

opérations à exécuter. De manière générale, un plan est une spécification d'opérations à exécuter en vue de répondre à une demande donnée sous la forme d'un état objectif. Pour cela, le raisonnement se base sur les effets possibles des opérations sur le système devant atteindre l'état objectif et les conditions requises sur l'état de ce dernier pour exécuter chaque opération.

Un problème de planification automatique se caractérise par trois composantes :

- un domaine qui est modélisé par un système à événements discrets. Il est défini par un 4-uplet $\Sigma = (S, A, E, \gamma)$ où : S est un ensemble d'états, A est un ensemble d'opérations, E est un ensemble d'événements, et $\gamma : S \times A \times E \rightarrow 2^S$ est la fonction de transition d'états. Les événements sont liés soit à l'action de l'environnement sur le système considéré, soit à une défaillance.
- la demande, dont la formulation la plus simple est un état objectif, mais dans bien des problèmes, cette formulation est plus complexe. Il s'agit par exemple de satisfaire certaines conditions sur la séquence d'états parcourue par le système : éviter des états ou bien imposer le passage par d'autres états dans un ordre précis.
- un état initial appartenant à S.

Les questions auxquelles la planification automatique tente d'apporter des réponses sont essentiellement de deux ordres. Cela concerne d'abord la façon de représenter les états et les opérations sans avoir recours à une représentation explicite et énumérative des états S. Ensuite, la deuxième question est de savoir comment aboutir, le plus rapidement possible, à une solution correcte sans une représentation explicite de l'espace d'état. Ceci implique de définir des algorithmes, des heuristiques et des techniques d'orientation dans l'espace d'états pour trouver une solution.

Les problèmes de planification automatique sont classifiés en fonction des hypothèses qu'ils respectent vis-à-vis de celles de la planification dite classique. Les hypothèses d'un problème classique de planification sont les suivantes :

- **H0 :** le système Σ a un nombre fini d'états.
- **H1 :** quel que soit l'état du système Σ, il est observable.
- **H2 :** le système Σ est déterministe. Si une opération est exécutable depuis un état S, l'exécution de l'opération conduit le système Σ dans un seul et unique état, de même pour l'occurrence d'un événement.
- **H3 :** le système Σ est statique, l'ensemble des événements est vide. Une évolution du système résulte forcément de l'exécution d'une opération.
- **H4 :** l'objectif se limite à des contraintes sur l'état à atteindre. Il n'existe pas de contrainte sur les états entre l'état initial et l'état final.
- **H5 :** un plan, solution d'un problème de planification, est une séquence finie, ordonnée et linéaire d'opérations.
- **H6 :** les opérations et les événements n'ont pas de durée. Le passage d'un état à un autre est instantané.
- **H7 :** un plan est complètement défini avant d'être appliqué. Il n'est pas construit dynamiquement.

La planification classique est une formulation très restrictive du processus de planification au sens large. Bien des problèmes pratiques ne respectent pas ces hypothèses. Malgré cela, la grande majorité des recherches concerne la planification classique. La première conséquence est une faible intégration de la planification dans les systèmes de pilotage des systèmes de production (Ghallab et al., 2004). Ces techniques s'appliqueraient d'abord aux niveaux 3

et 4 du modèle CIM. Elles viseraient à définir les opérations à réaliser dont le résultat est aujourd'hui donné sous la forme de gammes de fabrication (Sornaz et Khoshnevis, 2003) ou de gamme de montage (Ben-Arieh et al., 2004). A ce sujet, il ne faut pas confondre ces techniques avec les activités du service communément appelé planification[1] dans les systèmes de production.

L'hypothèse H5 est la plus restrictive à des fins conception de lois de commande au niveau coordination. En effet, elle doit être levée en priorité pour permettre de concevoir des lois de commande incluant des parallélismes et des synchronisations. Les premiers travaux dans le domaine des systèmes de production ont étendu la notion d'opération qui habituellement comporte la description de l'effet et de la pré-condition à satisfaire pour son exécution. Cette extension a ajouté une condition à satisfaire durant l'opération. Les travaux ne cherchaient pas à générer un plan, mais seulement à vérifier le respect de l'ensemble des pré-conditions et conditions pour un plan contenant des parallélismes (Sandewall et Rönnquist, 1986).

De ces travaux, plusieurs approches (Klein, 1999), (Aylett, 2001) ont été développées pour la conception de lois de commande destinées à des procédés batch composés principalement de pompes, valves, réservoirs et mélangeurs. Le principal reproche fait à ces approches (Castillo, 2000) est de ne pas donner un plan suffisamment détaillé pouvant directement être considéré comme une loi de commande au niveau coordination. De plus, la particularité des procédés batch est qu'il n'existe pas d'interaction entre les ressources.

Une application à un procédé manufacturier est proposée dans (Castillo et al., 2000), (Castillo et al., 2001), mais elle ne fait pas non plus apparaître de contraintes d'interactions entre les ressources. La modélisation proposée ne tient pas compte des événements liés à l'action de l'environnement comme par exemple l'arrivée d'une pièce dans un stock, ou l'évacuation d'un palet par un opérateur humain suite à la réalisation d'un montage sur ce dernier. Initialement développées pour les procédés batch, toutes les opérations ont une opération complémentaire du point de vue des effets sur le flux de produits, par exemple à une opération d'ouverture d'une vanne correspond une opération de fermeture. De par les contraintes modélisées, la spécification d'une loi de commande comporte forcément les deux opérations complémentaires. Le niveau de détail recherché pour correspondre à une loi de commande est basé sur ce mécanisme. Même si cette hypothèse ne semble pas trop restrictive pour une loi de commande en fonctionnement normal, elle est en revanche un point fortement bloquant dans un mode de reprise. Supposons en effet que suite à une défaillance, un vérin soit en position sortie, si pour revenir en fonctionnement normal le vérin doit être rentré, alors la loi de commande pour la reprise fera uniquement apparaître l'opération de rentrée du vérin.

Pour la partie algorithmique, la modélisation implique l'utilisation d'un algorithme basé sur un mécanisme de chaînage arrière ou régression conduisant par exemple à l'anomalie de Sussman (Weld, 1994). Ce mécanisme ne permet pas de prendre en compte les divergences et convergences en OU qui résultent généralement de l'action de l'environnement. Mais cette limitation vis-à-vis des divergences en OU ne résulte pas uniquement de l'algorithme, mais également de l'utilisation d'un graphe de précédence comme outil de représentation de la solution. Enfin, l'approche ne permet pas la spécification de lois de commande cycliques. En résumé, même si cette approche semble très prometteuse, elle se limite à la spécification de lois de commande

[1] Le service de planification définit les opérations à réaliser à partir des quantités commandées et des nomenclatures des produits définissant pour chacun d'eux les opérations à réaliser.

cycliques en fonctionnement normal où pour toutes les opérations, l'opération complémentaire existe. Enfin, l'optimisation de la solution n'est pas possible par un algorithme de planification non-déterministe dans le sens où l'algorithme ne dispose pas de critères afin de décider de l'opération à exécuter quand plusieurs sont possibles. C'est la première trouvée qui est choisie.

Plus classiquement dans le domaine de la planification, lever l'hypothèse H5 n'aboutit généralement pas à une solution dont l'application fera apparaître effectivement des parallélismes. Il s'agit plus généralement de définir un plan partiel dans lequel seules les décisions nécessaires au respect des contraintes et de la demande sont prises (Weld, 1994). On peut considérer un plan partiel comme une conjonction de contraintes qui définit un ensemble de plans "candidats". Le processus de recherche par un planificateur dit non-linéaire va réduire cet ensemble jusqu'à ce que tout plan incohérent ait été écarté. Les plans restants sont alors des solutions du problème. Un plan partiel représente une famille de plans (Trinquart, 2004). L'affectation ensuite des ressources et la définition des dates d'exécution sont le résultat de l'application de techniques d'ordonnancement présentées dans la section suivante.

3 L'Ordonnancement

L'ordonnancement est un domaine très vaste dont les applications sont très nombreuses comme l'allocation dynamique des ressources dans les systèmes d'exploitation d'ordinateurs, l'organisation d'activités de service, la définition de grands projets. De manière générale, l'ordonnancement peut se définir ainsi (Baker, 1974) :

L'ordonnancement est la distribution des ressources dans le temps pour effectuer un ou plusieurs travaux composés chacun d'une ou plusieurs opérations.

Comme le laisse entrevoir cette définition, le problème d'ordonnancement comporte deux sous problèmes : le premier consiste à affecter les opérations aux ressources, et le second a pour objet de fixer le calendrier de l'exécution des opérations pour chaque ressource, c'est un ordonnancement.

Les problèmes d'ordonnancement les plus proches de notre problématique sont ceux d'ordonnancement d'ateliers visant à spécifier un comportement du flux de produits. Ils s'appliquent plus particulièrement aux niveaux 3 et 4 de l'architecture CIM. La complexité du problème général d'ordonnancement d'atelier est telle que des simplifications sont presque toujours nécessaires. Elles sont présentées avant de nous focaliser sur les caractéristiques des modèles de la partie opérative et du flux de produits, de la demande et des algorithmes utilisés classiquement en ordonnancement d'ateliers.

En ordonnancement, un système de production est vu comme l'union de deux systèmes : un système de transformation composé de toutes les ressources agissant sur l'état physique des produits ou contrôlant leur état (MOCN, MMT, etc) et un système de transitique composé de toutes les ressources permettant de modifier l'état spatial des produits. Face à la complexité de ce problème, une résolution analytique nécessite de simplifier le problème par l'hypothèse de découplage des problèmes de transformation et de transitique (Brauner et al., 2004). Un des deux systèmes est considéré prépondérant par rapport à l'autre dans le sens où c'est lui qui est le plus contraignant pour les critères considérés. Cette hypothèse implique que l'autre système est capable d'absorber l'ensemble des demandes qui lui seront faites. Dans ces deux

domaines, les problèmes sont formalisés de la même façon. Ainsi, nous présentons maintenant les caractéristiques du modèle de la partie opérative et du flux de produits, de la demande et des algorithmes en montrant les similitudes mais aussi les différences avec la conception de lois de commande au niveau coordination.

Le modèle du système contrôlé (partie opérative et flux de produits) retenu en ordonnancement d'atelier définit les caractéristiques de chacune des ressources (disjonctives et renouvelables)(Esquirol et Lopez, 1999). De plus, pour chacune des ressources, les opérations qu'elles réalisent sont connues. Les opérations sont caractérisées par une durée opératoire et leurs effets sur le flux de produits donnés simplement par le nom de l'opération. Cette modélisation ne faisant pas apparaître l'état des ressources et du flux de produits, hormis pour la disponibilité des ressources, les contraintes d'interactions physiques (PO/PO, produits/PO et produits/produits) ne sont pas faciles à exprimer. Considérons la collision possible entre deux robots, la contrainte visant à éviter cette collision ne peut pas être écrite simplement par une proposition sur les variables d'états des robots, mais elle nécessite de rechercher l'ensemble des exclusions mutuelles entre les opérations offertes par les deux robots. La principale conséquence est un risque majeur d'erreur lors de la modélisation. Le modèle de la physique de l'atelier représente les capacités de ce dernier (tout ce qu'il peut faire), la spécification d'un comportement (ce qui doit être fait) parmi tous ceux offerts dépendra de la demande.

La demande impose les travaux à réaliser et donne les critères à optimiser. Aujourd'hui, elle prend aussi en compte la notion de garantie de performance (Rossi, 2003) dans un contexte incertain. Ce concept est lié à la distance entre le modèle utilisé pour l'optimisation et le système réel. Cette distance ne permet jamais d'atteindre les valeurs des critères optimisés. Ainsi, plutôt que de chercher un extremum pour un jeu de données exactes, il est recherché une solution pour laquelle les critères restent dans un intervalle donné en fonction d'une incertitude sur la demande.

A l'inverse de la conception de lois de commande au niveau coordination et de la problématique de la planification, il n'existe pas de problème de choix des opérations à réaliser car les travaux dont la décomposition en opérations est connue sont imposés par l'objectif. Même quand toutes les opérations ne sont pas données par la décomposition des travaux, elles se déduisent simplement. Par exemple, dans les problèmes avec prise en compte des opérations de préparation, il est toujours considéré une seule et unique opération de préparation pour modifier l'état d'une ressource telle qu'elle puisse exécuter l'opération de production suivante (Artigues, 1997). Si plusieurs séquences d'opérations de préparation existaient pour passer dans l'état souhaité de la ressource, toutes ces séquences ne pourraient pas être trouvées et devraient toutes être modélisées. Or, au niveau du modèle CIM auquel se place notre problématique, le détail requis pour la définition des opérations implique bien souvent l'existence de plusieurs opérations pour passer d'une opération de transformation à une autre. Finalement, la distance entre la demande considérée en ordonnancement et celle d'une loi de commande au niveau coordination est trop importante pour envisager une utilisation directe d'une méthode d'ordonnancement.

Néanmoins, il serait envisageable de coupler des techniques de planification présentées dans la section précédente avec des techniques d'ordonnancement. Ceci suppose de pouvoir dissocier la recherche des opérations requises, de l'allocation des ressources et de la définition des dates d'exécution. Si la vision limitée aux opérations de transformation au niveau 3 et 4 du modèle CIM permet cette dissociation basée sur les données fournies par les spécifications des

produits, elle n'est pas possible au niveau 1 dont l'existence des opérations de transitique et de préparation n'est liée à rien d'autre qu'à l'affectation des opérations de production. Après avoir étudié la formulation de la demande, les caractéristiques principales des algorithmes utilisés sont présentées.

Par la vision ressources auxquelles sont affectées des opérations dans le temps, les algorithmes d'ordonnancement spécifient naturellement un comportement avec des parallélismes et des synchronisations. De plus, les méthodes formelles de résolution utilisées impliquent toujours un respect des contraintes modélisées.

Mais, il n'existe pas encore de méthode de résolution à la fois générale et de faible complexité algorithmique (Esquirol et Lopez, 1999). La nature du problème d'ordonnancement peut se trouver complètement modifié suite à un aléa. Par conséquent, la résolution nécessitera un nouvel algorithme. Ceci peut s'avérer bloquant dans le contexte de la réactivité aux aléas si un opérateur est requis pour fournir l'algorithme à appliquer en fonction du problème.

La complexité des problèmes généraux liés aux systèmes flexibles de production manufacturière sont équivalents à un problème d'open shop (Esquirol et Lopez, 1999). Comme nous venons de le voir, les solutions apportées en ordonnancement se focalisent plus particulièrement sur la définition du calendrier d'exécution en supposant disposer de l'affectation. Ainsi dans la section suivante, il est proposé une approche visant à répondre à ce problème d'affectation des opérations de chacun des travaux séparément, en considérant à la fois le système de transformation et de transitique.

4 Génération de Gammes Opératoires

Plusieurs auteurs traitent du problème de la génération de gammes opératoires (Amar, 1994) (Sornaz et Khoshnevis, 2003). Mais pour mieux comprendre l'intérêt d'une telle approche pour la réactivité aux aléas, nous l'étudierons dans le cadre d'une approche de SSC, et en particulier l'approche CASPAIM développée au LAGIS. Ainsi, avant d'étudier la génération des gammes opératoires, l'approche CASPAIM est présentée.

La méthodologie CASPAIM (Conception Assistée des Systèmes Automatisés de Production en Industrie Manufacturière) couvre tous les aspects de la commande, de la surveillance et de la supervision. Elle se focalise essentiellement sur les niveaux bas de l'architecture CIM (niveau 1 et 2). L'architecture de l'approche, présentée dans la Figure 2.1, se compose de quatre modules en interactions : la supervision, la surveillance, la commande et la maintenance. Les modules de supervision et de maintenance sont en liaison avec les niveaux supérieurs (planification et ordonnancement). Au niveau le plus bas, seul le module de surveillance est connecté à la partie opérative. La solution fournie par le niveau ordonnancement est considérée non déterministe au niveau de l'affectation des produits et du routage. Ces degrés de liberté sont ainsi pris en compte très tard et résolus en exploitation. Le rôle de la supervision est de paramétrer cette commande afin de "coller" au plus près à l'objectif de production. Dans ce but, la supervision dispose d'un module de pilotage, de recouvrement et de gestion des modes de marches et d'arrêts. Suite à une défaillance de la partie opérative, le module de recouvrement (Berruet, 1998) décide des actions à mener pour disposer d'au moins une gamme opératoire pour chacun des produits à

fabriquer. Ces actions définissent l'état à atteindre d'un point de vue des modes de marches et d'arrêts dont la commutation pour chacune des ressources est assurée par la gestion des modes (Dangoumau, 2000). A partir des gammes opératoires disponibles, le pilotage basé sur des techniques d'ordonnancement établit dynamiquement le cheminement des pièces dans le système, en fonction de la charge de ce dernier (Tawegoum, 1995). Une gamme opératoire, élément de base pour le pilotage, spécifie le comportement du flux de produits pour un type de produit. Spécifier ce comportement est l'étape amont à la spécification du comportement de la partie opérative. Pour cette raison, nous présentons maintenant la génération de gammes opératoires.

Dans un premier temps, des gammes logiques sont définies à partir des spécifications des produits et des fonctions de transformation communes à tout atelier. Ces fonctions (tourner, fraiser, percer, etc) ne sont pas affectées à une machine car, à ce stade, la physique de l'atelier n'est pas connue (Cruette, 1991). La gamme logique d'un produit (Cruette, 1991) représente la séquence des différentes fonctions de transformations à réaliser, afin d'obtenir le produit fini à partir de son état brut en entré du SAP. La flexibilité de gamme est prise en compte à ce niveau.

Dans un deuxième temps, les gammes logiques sont directement traduites en gammes opératoires (Amar, 1994) grâce à la prise en compte des moyens de production et de transport d'un système de production. Basé sur ce principe, une modélisation particulière a ensuite été proposé, le Graphe d'Accessibilité Opérationnelle (GAO) (Berruet, 1998) dont une extension en réseaux de Petri pour l'expression d'exclusions mutuelles (Belabbas et Berruet, 2004) est proposée. Le passage de la gamme logique à la gamme opératoire consiste à affecter les fonctions de transformation et à introduire les opérations de transitiques. A partir des gammes logiques de produits à fabriquer et du GAO, les notions d'accessibilité d'une opération de transformation depuis l'entrée ou vers la sortie du système de production ont été introduites. Ainsi pour chacune des gammes logiques, l'accessibilité de chacune des machines de transformation est déterminée avant même de construire explicitement une gamme opératoire. Dans un contexte de réactivité aux aléas de fonctionnement, cette notion d'accessibilité offre très rapidement pour la

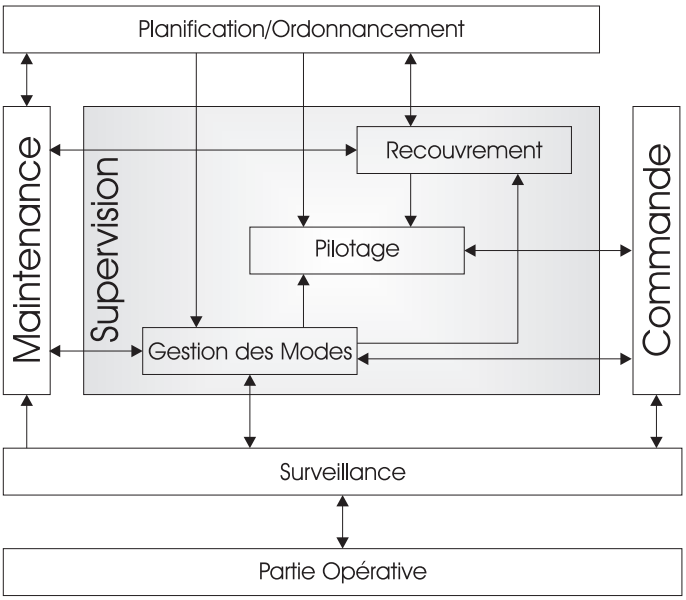


Fig. 2.1 – Architecture de l'approche CASPAIM.

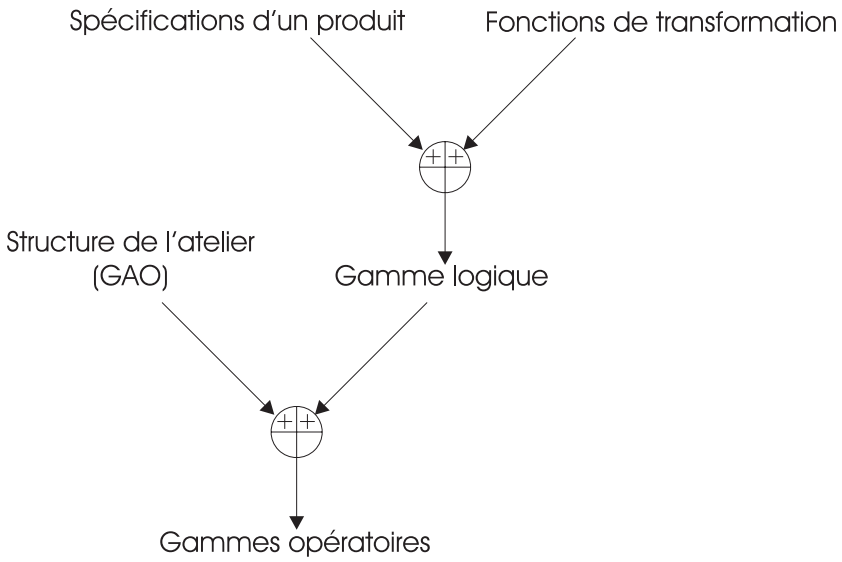


FIG. 2.2 – Principe de génération des gammes opératoires.

prise de décisions une connaissance des capacités du SAP à réaliser les produits demandés. En fonction des flexibilités physiques offertes par l'atelier, il existe généralement plusieurs gammes opératoires pour une gamme logique.

Mais le niveau de détail avec lequel sont vus la partie opérative et le flux de produits n'est pas suffisant pour la conception de lois de commande traitée dans ce manuscrit. La vision donnée par le GAO de la partie opérative est très proche de celle de l'ordonnancement d'atelier pour les niveaux 3 et 4 du modèle CIM. En effet, les interactions entre les ressources ne sont pas modélisées, comme par exemple l'interaction possible entre deux robots accédant à une zone commune. De plus, l'approche définit de manière indépendante les gammes opératoires de chacun des produits. Par conséquent, les flux de produits résultant de chacune des gammes opératoires ne sont pas coordonnés. Le risque est alors de conduire le système dans un état bloquant ou bien de le saturer. Suivant la façon dont l'objectif du SAP est formulé, il est envisageable de baser le module de pilotage sur deux approches complémentaires à celle-ci, une approche basée sur les techniques d'ordonnancement qui a été retenue dans l'approche CASPAIM et le contrôle par supervision présenté dans la section suivante.

Tout d'abord, si la demande contient initialement l'ensemble des travaux à réaliser (produits à fabriquer avec les quantités et les délais), alors en fonction de cette demande, des gammes opératoires et du modèle de l'atelier, il reste à fixer le calendrier de l'exécution des opérations pour chaque ressource ; c'est un problème d'ordonnancement. Dans ce cas, le flux de produits entrant sera imposé par l'ordonnancement calculé.

A l'inverse pour une demande évoluant très rapidement et arrivant de façon quasi continue, le flux de produits entrant ne sera pas contrôlé dans le sens où un produit sera injecté, sans considération des autres, dans le SAP chaque fois qu'une commande arrivera. Dans ce contexte, les produits arrivent de manière aléatoire dans le système de production. Pour éviter les états bloquants, une approche de contrôle par supervision basée sur la théorie de Ramadge et Wonham peut être une solution. Cette approche est présentée dans la section suivante.

5 Les Approches basées sur la Théorie de Ramadge et Wonham

La théorie proposée par Ramadge et Wonham (Ramadge et Wonham, 1987), à l'origine de nombreux travaux visant à piloter un système de production, s'intéresse à l'existence et à la synthèse de superviseurs. Le principe de cette théorie est la séparation claire entre le modèle du système à contrôler, appelé générateur, et le modèle des spécifications que doit respecter ce système. En l'absence d'un superviseur, le générateur est dit non contrôlé. Les événements générés sont spontanés si aucun mécanisme ne vient contraindre l'occurrence des événements.

L'ensemble des événements générés est appelé alphabet du générateur et noté Σ. Cet alphabet est partitionné en deux ensembles disjoints : les événements contrôlables Σ_c et les événements incontrôlables Σ_{uc}. Le superviseur ne génère aucun événement. Il observe tout ou partie des événements générés et interdit l'occurrence de certains événements de manière à respecter les spécifications.

La notion de spécification peut présenter deux caractères particuliers (Chafik, 2000) :

- spécifications à caractère négatif par le fait qu'elles indiquent ce que ne doit pas faire le générateur. Dans le contexte des systèmes de production, elles assurent le respect des contraintes de sécurité et d'écologie ;
- spécifications à caractère positif par le fait qu'elles indiquent ce que doit faire le générateur. Elles expriment les contraintes relatives à la demande pour l'état final du flux de produits et de la partie opérative pour une application aux systèmes de production.

La théorie est basée sur les langages formels et une modélisation du générateur, des spécifications et du superviseur par des automates à états finis. Pour palier au problème d'explosion du nombre d'états, d'autres approches basées sur les réseaux de Petri ont été proposées dans la littérature (Holloway et Krogh, 1990), (Ghaffari et al., 2003).

Pour le pilotage des systèmes de production, cette théorie est utilisée de deux manières différentes.

La première utilisation vise à l'obtention de lois de commande devant imposer un comportement d'une part à la partie opérative et d'autre part aux flux de produits. Ces deux ensembles d'éléments sont vus comme le générateur, (Brandim, 1996), (Nourelfath, 1997), (Chafik, 2000) et pour partie dans (Gouyon, 2004). Le principal problème auquel ces travaux se trouvent confrontés est l'écart sémantique entre une loi de commande qui impose un comportement et un superviseur

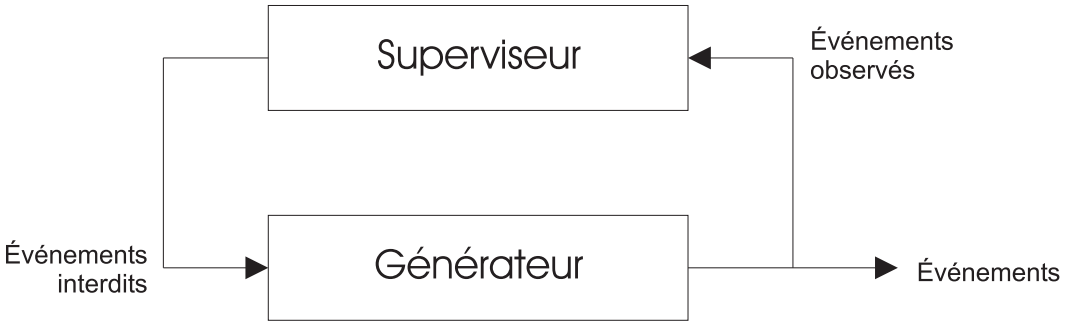


FIG. 2.3 – Schéma de contrôle par supervision selon la théorie de Ramadge et Wonham.

qui limite le comportement d'un générateur spontané d'événements (Zaytoon, 2002). Cet écart conduit à devoir formuler des hypothèses très restrictives ne permettant pas l'application dans un contexte industriel complexe. A cette limitation s'ajoute la difficulté d'écriture des spécifications, et en particulier des spécifications positives.

En effet, l'écriture de ces spécifications revient dans bien des cas à être capable d'écrire la loi de commande qui est une spécification du comportement souhaité de la partie opérative et du flux de produits. Qui plus est, le formalisme basé sur les automates à états ne facilite pas l'écriture des spécifications à la charge de l'expert. Toujours dans une approche de génération d'une loi de commande et partant du principe que les spécifications positives sont les plus difficiles à écrire, la théorie de Ramadge et Wonhan a été utilisée pour vérifier une loi de commande spécifiée en Grafcet (Zaytoon et Carré-Ménétrier, 1999). Mais, là aussi, la spécification du comportement souhaité de la partie opérative est initialement donnée par un opérateur humain qui sera également chargé de trouver les modifications à apporter tant que les spécifications négatives ne seront pas respectées. Par conséquent, de telles techniques ne sont pas envisageables dans le cadre de notre problématique de conception de lois de commande dans un contexte de réactivité aux aléas.

La seconde utilisation, le contrôle par supervision, (Charbonnier et al., 1999), (Lee et Lee, 2002), (Qiu et al., 2003), (Achour et Rezg, 2004) de la théorie de Ramadge et Wonham ne vise pas à imposer un comportement à une partie opérative. Mais elle supervise un ensemble de ressources qui disposent chacune de leur propre système de commande. Ce sont tout aussi bien des chaînes fonctionnelles que des machines comme un robot, un tour, une fraiseuse, etc. L'ensemble des ressources avec leur système de commande propre génère spontanément des événements qui sont les débuts (contrôlables) et les fins (incontrôlables) des opérations offertes par les entités. D'autres événements incontrôlables sont liés à l'arrivée ou au départ de pièces dans les stocks d'entrée et de sortie. Dans un fonctionnement sans superviseur, les entités évoluent de façon complètement indépendantes les unes des autres sans prise en compte des contraintes liées à leurs interactions. Dans ce contexte, il est nécessaire en fonction de l'état du générateur d'interdire des événements contrôlables pouvant conduire dans un état bloquant ou violant les contraintes de sécurité et d'écologie. Dans le contexte de cette utilisation, le système de production est piloté par le flux de produits. Chacun des produits qui entre dans le SAP contient la séquence des opérations qu'il doit subir. Par analogie, le générateur peut être vu comme l'ensemble des véhicules en circulation dans une agglomération et le code de la route est alors équivalent aux spécifications. Il est par exemple interdit de s'engager dans un carrefour sans s'être assuré de pouvoir en sortir afin d'éviter un état de blocage. Pour ce genre de système, les entités le composant (ressource et produit) ont pour particularité d'avoir chacune leur propre objectif. Il y a une absence totale de coordination globale du système en fonction des objectifs de chacun. Ainsi, ces systèmes ne sont pas optimisés et ils ne sont alors jamais utilisés au maximum de leur capacité. Ils sont sur-dimensionnés vis-à-vis des besoins de l'entreprise.

En conclusion, l'une et l'autre des utilisations faites de la théorie de Ramadge et Wonham n'apportent pas une réponse satisfaisante à la conception de lois de commande au niveau coordination dans le contexte d'exécution présenté au § 5 page 31. Après avoir présenté différentes démarches issues du monde de la recherche et contribuant à la conception de lois de commande, nous nous proposons, avant de clore ce chapitre, de donner une image instantanée de la démarche industrielle actuelle.

6 Démarche Industrielle de Conception d'une Loi de Commande

Toutes les approches présentées précédemment offrent une démarche formelle de spécification d'un ou d'un ensemble de comportements de la partie opérative et du flux de produits. L'aspect formel des démarches est lié au respect par le comportement spécifié de l'ensemble des contraintes formalisées individuellement dans le modèle du système contrôlé et la demande. Mais toutes ces démarches formelles ne répondent pas au problème industriel complexe de conception de lois de commande destinées aux niveaux 1 et 2 de l'architecture CIM. De par l'inadéquation de ces approches, la démarche de conception, actuellement appliquée en entreprise, est centrée sur les compétences humaines. Le principal acteur de la conception d'une loi de commande reste l'expert avec ses qualités en terme de capacité à prendre des décisions en définissant de nouveaux critères si nécessaire, et ses faiblesses notamment par ses capacités limitées de mémoire qui sont généralement sources d'erreurs face aux problèmes complexes.

Néanmoins, l'expert peut s'appuyer sur des méthodologies visant à le guider dans sa démarche de conception, comme le Guide des Modes de Marches et d'Arrêts (GEMMA) (ADEPA, 1981), (Moreno, 1997). D'autres démarches fonctionnelles ou orientées objets ont été proposées (Sargent, 1998) (Zaytoon, 2002). Toutes ces démarches sont des guides méthodologiques pour appréhender la complexité en décomposant un problème général en sous-problèmes. Elles ne sont pas formelles et elles restent centrées sur l'expert dont la démarche de conception d'une loi de commande est maintenant présentée.

La Figure 2.4 représente le processus actuel de conception d'une loi de commande par un expert. Ce processus est entièrement basé sur la capacité de l'expert à se créer une image correcte des évolutions, de la partie opérative et du flux de produits, satisfaisant les contraintes de sécurité et d'écologie. De la justesse de cette image dépend l'intégrité des biens et des personnes par le respect des contraintes de sécurité et d'écologie. L'expert ne formalisant jamais son image du système contrôlé, une validation de cette dernière est par conséquent impossible. En supposant

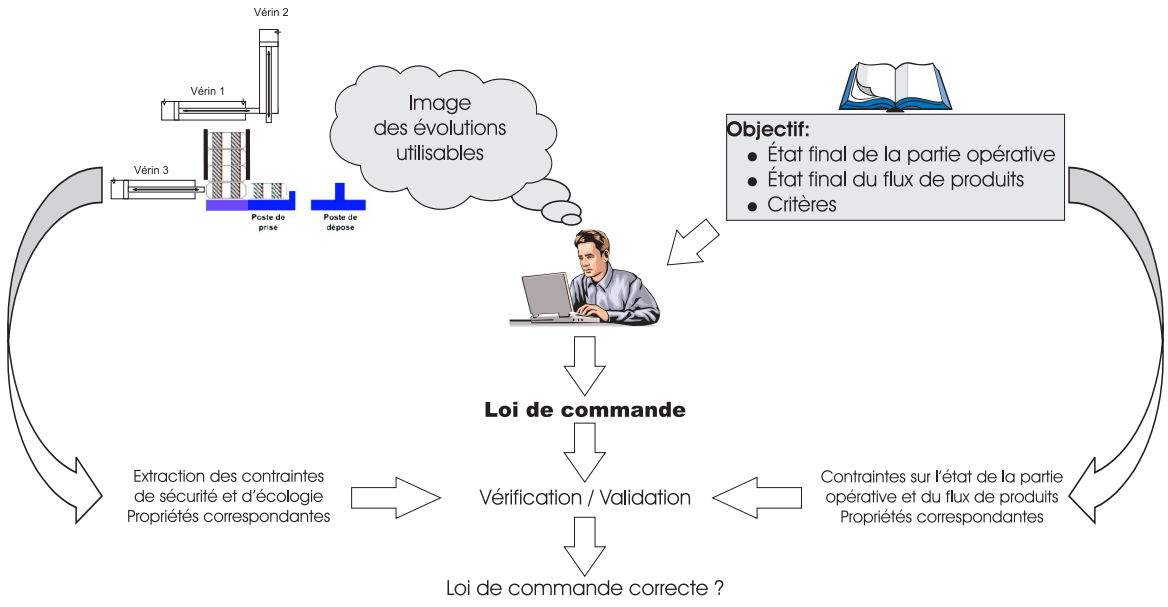


FIG. 2.4 – Démarche industrielle actuelle de conception d'une loi de commande.

que l'expert dispose à un moment donné d'une image correcte du système contrôlé, il se pose alors le problème de la dégradation dans le temps de cette image. En effet, la conservation correcte de l'image est directement liée aux capacités intellectuelles et de mémoire de l'expert qui ne peuvent pas être garanties.

Puis, en fonction de la demande et de sa représentation du système contrôlé, l'expert conçoit une loi de commande en spécifiant les relations logiques entre les entrées et les sorties du système de commande telles que le comportement du flux de produits et de la partie opérative satisfasse la demande. Face à la complexité du problème de conception, il adopte généralement une démarche d'intégration progressive de l'ensemble des données. Pour cela, il spécifie des comportements avec différents points de vue. Dans une structure de pilotage où les niveaux coordination et chaînes fonctionnelles sont fusionnés, ces points de vue sont successivement une vision produits, puis partie opérative suivie d'une vision pré-actionneur et enfin entrées/sorties de l'automate programmable (Gendreau et al., 2000).

Même en supposant que l'expert dispose d'une image correcte des évolutions satisfaisant les contraintes de sécurité et d'écologie, la complexité du problème à résoudre est bien souvent telle qu'il lui est impossible de concevoir une loi de commande exempte de fautes. Dans une démarche d'élimination des fautes, une étape de vérification et de validation d'une loi de commande (Frey et Litz, 2000) est requise soit par simulation, soit par des approches formelles telles que le model-checking (mac Millan, 1993) (Schnoebelen, 1999) ou le theorem-proving (Emerson, 1990). La simulation trouve sa limite principale dans le problème de couverture des états accessibles. Il est en effet impossible de garantir que l'ensemble des situations aient été testées, qui plus est, le coût croît très fortement en fonction du nombre de situations testées. Pour les méthodes formelles, la difficulté majeure est leur mise en application dans un contexte industriel. Les formalismes et les langages (la logique temporelle par exemple) à la base de ces approches sont très difficiles à manipuler (Schnoebelen, 1999) et ils posent bien souvent le problème de l'écriture correcte des propriétés à vérifier (Henry et Faure, 2003).

Dans un contexte industriel, les conséquences de la démarche de conception d'une loi de commande sont nombreuses :

- par rapport à un ou plusieurs critères, l'optimisation de la loi de commande est laissée à l'expert qui se base sur une démarche empirique liée à son expérience et sur la phase de mises au point par des essais successifs. Par conséquent, il n'y a aucune garantie de performance.
- l'étape de vérification et de validation demande un temps comparable au temps de conception de la première version d'une loi de commande avant correction. Ceci est vrai aussi bien pour l'écriture des propriétés dans le cadre des approches formelles que pour la définition du protocole de simulation et la réalisation des simulations.
- la durée cumulée résultante de la spécification, de la vérification/validation et de la mise au point est conséquente.
- avec cette contrainte sur la durée de développement d'une loi commande ne permettant pas une conception rapide, la seule stratégie envisageable aux niveaux 1 et 2 de l'architecture CIM suite à une défaillance est la réparation de l'organe défectueux pour pouvoir revenir dans un fonctionnement identique. Ceci nécessite généralement un arrêt, non prévu, de la production pour l'intervention de la maintenance qui engendre généralement une forte dégradation des performances.

– seules les lois de commande sont formalisées. La capitalisation des connaissances se limite donc aux lois de commande et ne concerne pas les capacités du SAP qui sont pourtant le point de départ pour une re-conception. Si l'expert quitte l'entreprise, un travail important de ré-appropriation des connaissances sur les capacités de la partie opérative est alors nécessaire à un nouvel expert.

En conséquence, une méthode formelle de conception d'une loi de commande impliquerait des lois de commande plus sûres, une optimisation des critères de qualité et de productivité, une forte diminution des moyens financiers engagés pour la conception, et une capitalisation des connaissances. Au delà de ces quatre points, une conception automatique d'une loi de commande avec au moins un service analogue intégré à une approche de SSC améliorerait très nettement la réactivité aux aléas de fonctionnement.

7 Conclusion

Au travers de ce chapitre, nous avons réalisé un tour d'horizon des approches qui ont contribué significativement à la problématique de la conception de lois de commande. Volontairement distribuée sur différents domaines d'application, cette étude bibliographique nous a permis d'étudier l'adéquation des approches présentées aux besoins d'un module de coordination de chaînes fonctionnelles.

Ainsi, avantages et inconvénients de chacune des approches présentées ont été mis en exergue. Nous retiendrons en particulier que les approches de type Planification Automatique, bien que intéressantes sur le plan de la modélisation et de la formalisation de la demande, génèrent des modèles de comportements non optimaux, ce qui, dans le contexte manufacturier, est à rejeter. Les approches de type Ordonnancement souffrent d'un niveau d'abstraction du modèle du système contrôlé beaucoup trop important ; seules des opérations de transformation ou de transitique sont prises en compte généralement. Hors, comme nous le verrons dans la partie II de ce mémoire, au niveau de la coordination de chaînes fonctionnelles, ces deux types d'opérations, au moins, doivent être conjointement pris en compte. Cependant, il est à retenir que les approches de type Ordonnancement intègrent des méthodes d'optimisation fortement intéressantes pour garantir les performances d'une loi de commande (solution de l'ordonnancement). Les approches traitant de la problématique de Génération de Gammes Opératoires amènent des solutions amont aux techniques d'Ordonnancement en proposant des solutions en terme d'affectations. L'avantage essentiel de telles techniques repose sur leur capacité à évaluer la faisabilité d'une loi de commande, sans pour autant la générer. Ceci représente un intérêt considérable à la fois en terme de réponse à un appel d'offre ou encore en phase de pronostic suite à l'occurrence d'une défaillance. Cependant, ce type d'approches souffre du même problème décrit précédemment pour les approches d'ordonnancement ; les modèles spécifiés et utilisés sont beaucoup trop abstraits pour être utilisés au niveau de la coordination des chaînes fonctionnelles. Les approches de type Ramadge & Wonham proposent quant à elles des solutions très efficaces de synthèse de lois de commande essentiellement applicables au problème du pilotage par le flux de produit ; donc sans optimisation globale. Elles trouvent également un intérêt dans le cadre de la commande locale, en prise directe avec les capteurs et actionneurs. Cependant, force est de constater qu'elles montrent rapidement leurs limites dès que le système à commander devient complexe. C'est en particulier le cas du niveau coordination des chaînes fonctionnelles. D'autres approches traitent de la problèmatique de la conception de lois de commande à différents niveaux du modèle CIM

et auraient pu trouver leur place dans ce chapitre (Mader et al., 2001), (Roussel et al., 2004). Nous citerons également les travaux de Holloway (Holloway et al., 2000) qui développent une approche formelle pour la conception des lois de commande d'une chaîne fonctionnelle. Ces travaux ne sont malheureusement pas transposables au niveau supérieur pour lequel les auteurs renvoient à une approche basée sur la théorie de Ramadge et Wonham, présentée dans ce chapitre.

Au delà des avantages et inconvénients présentés par chacune de ces approches, il est important de retenir qu'elles s'accordent toutes, y compris celles usités en contexte industriel, à reconnaître le besoin d'un modèle du système contrôlé, d'une spécification formelle de la demande, et de la recherche d'un algorithme capable de concevoir, hors ligne ou en ligne une solution. Fort de ces considérations, le chapitre suivant va s'attacher à définir notre problématique.

Chapitre 3

Problématique et Démarche de l'Approche Proposée

1 Introduction

Dans le chapitre précédent, nous avons présenté un ensemble de travaux qui s'insèrent dans le domaine général de la conception de lois de commande. Une étude critique présentant les avantages et les inconvénients de chacune de ces approches a été faite.

Fort de cette prise de recul, nous proposons ici un chapitre assez synthétique permettant de définir précisément la problématique à laquelle nous allons apporter une réponse dans ce mémoire ainsi que la démarche de résolution que nous avons envisagée.

2 Problématique de la Conception de Lois de Commande

Aboutir à une formulation précise de la problématique est conditionné par la connaissance préalable d'une part des objectifs et des caractéristiques d'une loi de commande, et d'autre part des données initiales du problème de conception.

2.1 Objectifs et caractéristiques d'une loi de commande

Quelle que soit la nature du système de commande considéré (continu ou discret), l'exécution d'une loi de commande revient à amener le système contrôlé d'un état stable particulier à un autre état stable en respectant des contraintes et en optimisant des critères. Afin de bien mesurer l'impact de cette définition sur les objectifs du système de commande dans les deux derniers niveaux du CIM, prenons le temps d'examiner la Figure 3.1.

Comme nous pouvons le voir, cette définition, replacée au niveau d'une chaîne fonctionnelle, revient à amener un actionneur/effecteur d'un état stable à un autre état stable en optimisant certains critères comme le temps ou encore la stabilité. Ce changement d'état caractérise un service offert au niveau supérieur ; ce dernier pouvant être demandé sous la forme d'une requête à laquelle un compte rendu de réalisation du service est associé. Ce niveau du système de pilotage est aussi appelé niveau régulation (Gentil et Zamaï, 2003), il se focalise sur la commande de l'actionneur et ne considère donc pas les interactions avec son environnement. Les modèles généralement utilisés à ce niveau varient selon la nécessité de connaître ou non à tout moment tous les états du système contrôlé ; on utilisera respectivement des systèmes

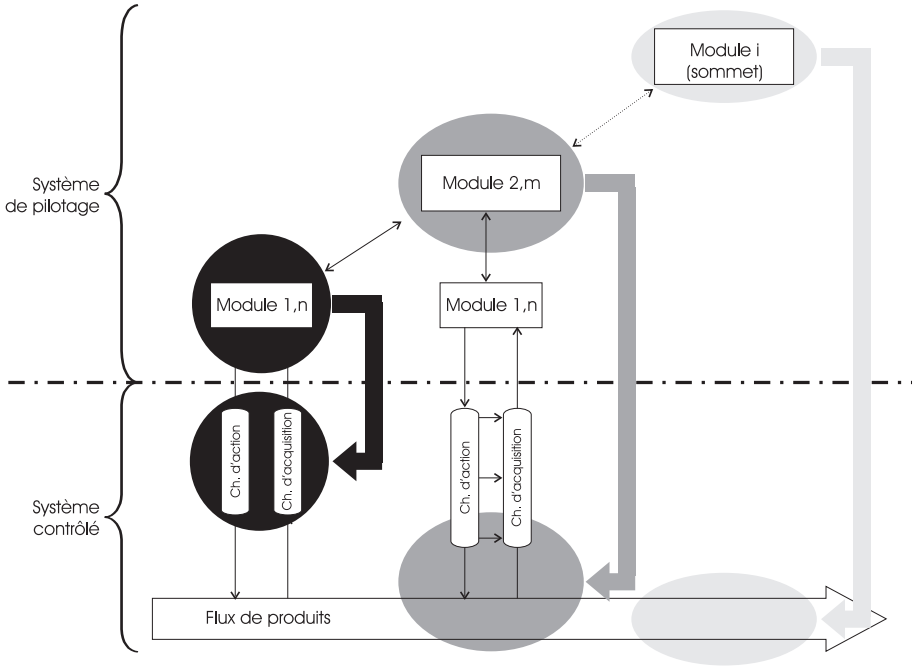


FIG. 3.1 – Les objectifs du système de commande

d'équations algébriques, différentielles, ou encore des fonctions de transfert pour spécifier le comportement des systèmes de commande continus, des modèles basés sur des lois périodiques d'observation (échantillonnage) pour les systèmes de commande discrétisés, des modèles issus du concept activité/transition où les points d'observation ne suivent pas forcément une loi périodique connue pour les systèmes de commande à événements discrets.

Au niveau de la coordination de plusieurs chaînes fonctionnelles, une loi de commande gère justement l'appel aux services de chacune des chaînes fonctionnelles afin d'agir sur l'état de la partie opérative voire également sur celui des produits. Elle dispose pour cela d'un ensemble de grandeurs de commande (requête d'appel aux services, compte rendu de fin des services, informations fournies par l'environnement) qu'elle exploite afin d'imposer certaines évolutions à la partie opérative et aux produits. Ces évolutions qui visent à répondre à une demande doivent de plus satisfaire un ensemble de contraintes de sécurité et d'écologie (cf. Figure 3.2).

Ces contraintes visent à garantir l'intégrité du système contrôlé et des opérateurs. Elles interdisent notamment des collisions entre les actionneurs (cas de deux vérins en L) ou bien encore entre les pièces. Quant à une demande à ce niveau, elle peut être de nature assez différente selon qu'elle spécifie uniquement des états de la partie opérative à atteindre (cas d'une mise à l'arrêt en fin de cycle de la partie opérative, cas d'une marche de vérification à vide, etc...) ou conjointement ceux du flux de produit (cas par exemple d'une production normale de pièces). Elle intègre en plus des critères d'optimisation tels que la qualité ou encore la productivité (réduction des temps de cycle, minimisation de la consommation énergétique, de la quantité de déchets, etc).

Compte tenu du niveau d'abstraction de la coordination des chaînes fonctionnelles, la connaissance permanente de tous les états de la partie opérative et des produits est soit non pertinente, soit inaccessible par manque de capteurs sur l'état des produits par exemple. De

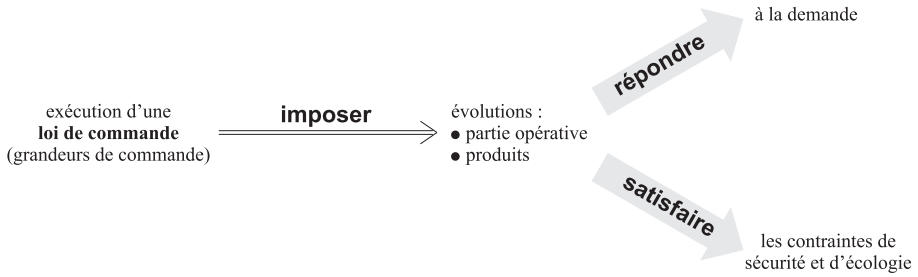


FIG. 3.2 – Objectifs d'une loi de commande du point de vue utilisateur.

surcroît, les événements significatifs issus des chaînes fonctionnelles sont entièrement caractérisés par le mode de communication retenu (mode appel/réponse, cf. § 2.4 page 22) et sont donc apériodiques. L'ensemble de ces considérations nous conduit à conclure quant à la nature des lois de commande de ce niveau hiérarchique, à savoir, à événements discrets.

Les caractéristiques essentielles des lois de commande que nous considérons dans ces travaux étant maintenant présentées, la section suivante détaille l'ensemble des données initiales du problème de conception de lois de commande.

2.2 Les données initiales du problème de conception

En sus de la demande à laquelle la loi de commande doit permettre de répondre ainsi que des contraintes de sécurité et d'écologie à satisfaire, les données initiales incluent d'une part les grandeurs de commande par l'intermédiaire desquelles le module coordination va agir sur l'état de la partie opérative et des produits, et d'autre part le temps imparti pour répondre à la demande. Les quatre paramètres de la conception de lois de commande sont maintenant étudiés successivement en commençant par la demande, puis les grandeurs de commande suivis des contraintes de sécurité et d'écologie, et enfin le temps imparti.

2.2.1 La demande

Une loi de commande impose certaines évolutions à la partie opérative et aux produits afin de répondre à une demande (cf. Figure 3.2). Générée par la fonction *Décider*, une demande est constituée d'une part d'un état initial des chaînes fonctionnelles et des produits, et d'autre part d'une requête. Cette dernière émane soit du niveau supérieur lors d'une configuration, soit d'un besoin interne suite à une défaillance lors d'un processus de reconfiguration. Quelle que soit son origine, cette requête se compose au plus de trois éléments :

L'état final des produits, qui correspond, en production, à l'état de sortie des produits du sous-système constitué des chaînes fonctionnelles contrôlées par le module coordination. Cet état résulte d'une part de la physique de ce sous-système qui définit par exemple les stocks de sortie, et d'autre part des spécifications des produits. En fonction des spécifications, des états intermédiaires, liés parfois par des contraintes de précédence, peuvent en plus être imposés.

L'état final de la partie opérative qui est imposé afin de respecter des contraintes de fonctionnement dans un contexte de fonctionnement normal, de reprise ou même d'urgence. Par exemple dans un fonctionnement normal, des pistolets de peinture devront être propres lors de l'arrêt du système de production. Dans un contexte d'urgence, il s'agira d'imposer une position de repli comme par exemple pour une fraiseuse quatre axes pour laquelle la broche devra être placée en position haute.

Les performances avec lesquelles l'état final du flux de produits et de la partie opérative doivent être atteints. Au niveau coordination, les performances sont généralement exprimées en terme de temps de cycle ou de coût de production. Une phase de configuration visera à l'optimisation des performances tandis qu'une phase de reconfiguration cherchera à atteindre les performances nécessaires afin de répondre dans le temps imparti à la demande en cours.

Suite à une requête de configuration, une réponse donnée au niveau supérieur, qui se limiterait à une affirmation (ou à une négation) à atteindre l'état final des produits et de la partie opérative, rendrait impossible le calcul des coûts et des temps de production ; servant ensuite à fixer les prix de vente et les délais de livraison aux clients. Basé sur les performances établies lors de la phase de configuration, une requête de commande va consister à majorer ces performances d'une marge décisionnelle afin de laisser au niveau inférieur des degrés de liberté. Au niveau coordination, ces marges se limitent le plus souvent à des marges temporelles (Combacau, 1991).

Suite à l'occurrence d'une défaillance, la requête de commande en cours impose certaines performances. Ainsi dans un contexte de reconfiguration, la connaissance des performances d'une loi de commande permet d'être en mesure de pronostiquer au plus tôt la capacité à répondre encore à la demande et ainsi confiner la défaillance au niveau coordination, ou au contraire, pronostiquer l'incapacité du module coordination et ainsi propager au plus tôt la défaillance au niveau supérieur (Boufaied, 2000).

Afin de répondre à une demande telle que définie dans cette section, le niveau coordination dispose d'un ensemble de grandeurs de commande sur lesquelles se focalise la section suivante.

2.2.2 Les grandeurs de commande

Les grandeurs de commande sont pour le niveau coordination d'une part le moyen d'imposer des évolutions à la partie opérative et aux produits, et d'autre part une source d'informations sur l'état du système contrôlé et de l'environnement. En effet, les grandeurs de commande se composent des requêtes d'appel aux services offerts par les chaînes fonctionnelles, des comptes rendus de fin de réalisation des services appelés, et enfin des informations fournies par l'environnement sur ces actions.

Ainsi, les moyens d'action dont dispose un module de coordination sont définis par les requêtes associées aux services offerts par les chaînes fonctionnelles. Un compte rendu de fin informe le niveau coordination de la réalisation effective d'un service. En plus de ces informations liées aux comptes rendus de fin, le module de coordination dispose parfois d'informations fournies par l'environnement non seulement sur l'arrivée des produit, mais également sur leur

état d'entrée dans le système de production. De même, l'environnement informe parfois de l'évacuation des produits.

Mais par l'intermédiaire des grandeurs de commande, un module de coordination ne peut pas répondre à une demande sans se soucier des contraintes de sécurité et d'écologie présentées dans la section suivante.

2.2.3 Les contraintes de sécurité et d'écologie

Une réponse à une demande ne peut pas être donnée sans assurer l'intégrité du système contrôlé, des opérateurs et de l'environnement au sens large. Ainsi, l'utilisation des moyens d'action dont dispose le module de coordination doit satisfaire un ensemble de contraintes de sécurité et d'écologie (Mendez et al., 2004).

L'intégrité des biens et des personnes étant directement dépendante des interactions entre les éléments physiques qui composent le système de production, il est nécessaire de connaître ces éléments afin de pouvoir ensuite limiter leurs interactions par des contraintes. Ces interactions ne se limitant pas aux chaînes fonctionnelles contrôlées par le module de coordination, il s'avère nécessaire de considérer également l'environnement. Cette prise en compte de l'environnement est à l'origine de la redondance partielle lors de la distribution de modèle du procédé proposé dans (da Silveira, 2003).

En effet, la partie du système de production pilotée par un module de coordination accède à des espaces partagés avec les chaînes fonctionnelles pilotées par d'autres modules de coordination (cf. Figure 3.3). Finalement, les contraintes porteront sur les trois catégories d'éléments suivants : les chaînes fonctionnelles contrôlées, les produits contrôlés et l'environnement constitué de chaînes fonctionnelles pilotées par d'autres modules de coordination.

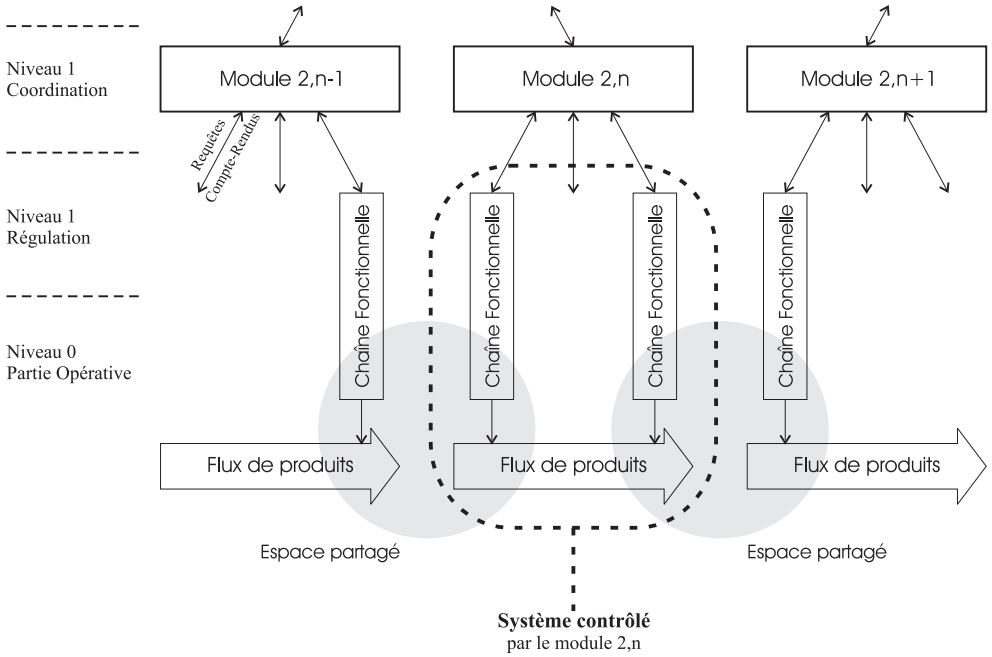


FIG. 3.3 – Interaction du système contrôlé avec son environnement.

Au final, les contraintes de sécurité et d'écologie à satisfaire seront généralement spécifiées par un ensemble d'états interdits comme par exemple deux vérins en L qui ne peuvent pas être conjointement en position sortie.

En sus de la demande, des moyens d'action et des contraintes de sécurité et d'écologie, la conception d'une loi de commande est également contrainte par le temps imparti à la conception et à l'exécution comme le présente la section suivante.

2.2.4 Le temps imparti

Dans un contexte de configuration comme de reconfiguration, les performances d'une loi de commande sont des données nécessaires non seulement pour le module de coordination mais également pour son supérieur hiérarchique.

En phase de configuration, l'intérêt majeur consiste à étudier les capacités du SAP à pouvoir répondre à un appel d'offre. Pour cela, une démarche de conception des bases de recettes (lois de commandes) est réalisée. En fonction des résultats obtenus, des processus de négociation peuvent être envisagés avec le client afin d'ajuster les délais de livraison et les prix de vente.

Nous voyons bien ici que la contrainte ne porte pas sur le temps imparti pour la conception et l'exécution d'une loi de commande, mais davantage sur les performances des lois de commande. Ainsi, la marge temporelle allouée à la phase de conception des bases de recette n'est pas fortement contrainte ; il s'agit avant tout de répondre justement à l'appel d'offre.

Dans ce contexte, une automatisation de la conception permettrait d'une part une meilleure réactivité à la demande des clients par une réponse plus rapide à des demandes spécifiques, et d'autre part une amélioration des temps de cycle, sources pour l'entreprise d'une meilleure maîtrise des coûts et des délais de production par leur connaissance précise au plus tôt.

En phase de reconfiguration, le temps imparti est connu via les performances fixées par la requête de commande en cours au moment de la défaillance. La contrainte temporelle qui découle de ce temps imparti est forte puisque la violer remettrait en cause une décision prise par le niveau supérieur, et aurait pour conséquence une propagation de défaillance (Chaillet, 1995).

La marge temporelle allouée pour le temps de conception et d'exécution de la nouvelle loi de commande est fonction des degrés de liberté laissés par le supérieur hiérarchique, ceci afin d'autoriser des décisions locales pour permettre un confinement des défaillances (Combacau, 1991). A l'inverse du temps imparti qui est connu via les marges temporelles, la répartition de ce temps entre la conception et l'exécution d'une loi de commande est très difficile à évaluer. Ainsi dans un contexte de reconfiguration, les stratégies pourront être variables suivant la flexibilité du système considéré et la capacité à concevoir plus ou moins rapidement une loi de commande. Deux stratégies s'opposent : allouer le temps nécessaire pour minimiser le temps de cycle de la loi de commande, ou à l'inverse minimiser le temps de conception en se limitant à la recherche d'une solution afin de disposer d'un temps maximum pour l'exécution.

Après avoir étudié les caractéristiques d'une loi de commande ainsi que les données initiales du problème de conception, la problématique de la conception de lois de commande est formulée dans la section suivante.

2.3 Formulation de la problématique

Au vu des éléments présentés précédemment et avec le formalisme SADT, la Figure 3.4 illustre les entrées et la sortie d'une fonction *Concevoir une loi de commande*. De par les objectifs d'une loi de commande définis au § 2.1, la conception de lois de commande consiste en fonction du temps imparti à exploiter les grandeurs de commande de manière à répondre à la demande tout en satisfaisant les contraintes de sécurité et d'écologie.

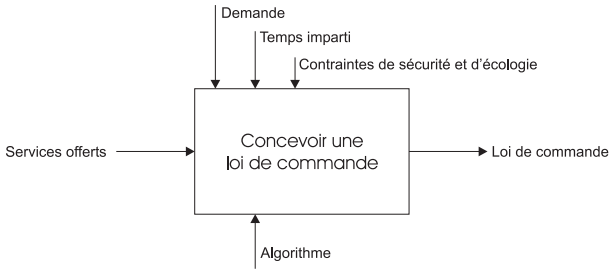


FIG. 3.4 – Données initiales du problème de conception.

En lisant à l'inverse, de la droite vers la gauche, la Figure 3.2 définissant les objectifs d'une loi de commande, la conception d'une loi de commande revient alors à traduire sous forme de grandeurs de commande, par des séquences de requêtes et de comptes rendus, d'une part la demande et d'autre part les contraintes de sécurité et d'écologie ; tout ceci en fonction du temps imparti. Toujours par une lecture inverse de la Figure 3.2, cette traduction nécessite de déterminer les évolutions de la partie opérative et des produits qui répondent à la demande et satisfont les contraintes de sécurité et d'écologie. Dans la Figure 3.5 qui illustre la problématique, seule la dimension temporelle de la problématique de conception de lois de commande n'apparaît pas.

Nous proposons dans la section suivante de donner une vue générale de l'approche proposée.

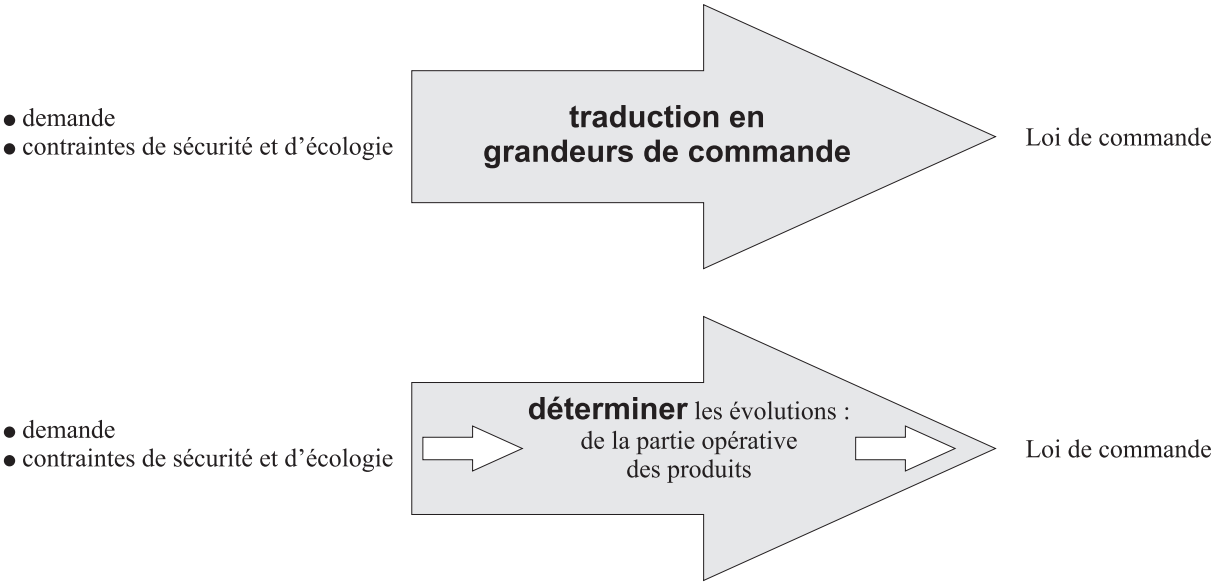


FIG. 3.5 – Problématique de la conception de lois de commande.

3 Démarche de l'Approche Proposée

La démarche proposée, dans la Figure 3.6, en réponse à la problématique de lois de commande, énoncées ci-dessous, se base d'une part sur les données initiales du problème de conception de lois de commande, et d'autre part sur le résultat attendu, à savoir, une loi de commande avec ses objectifs.

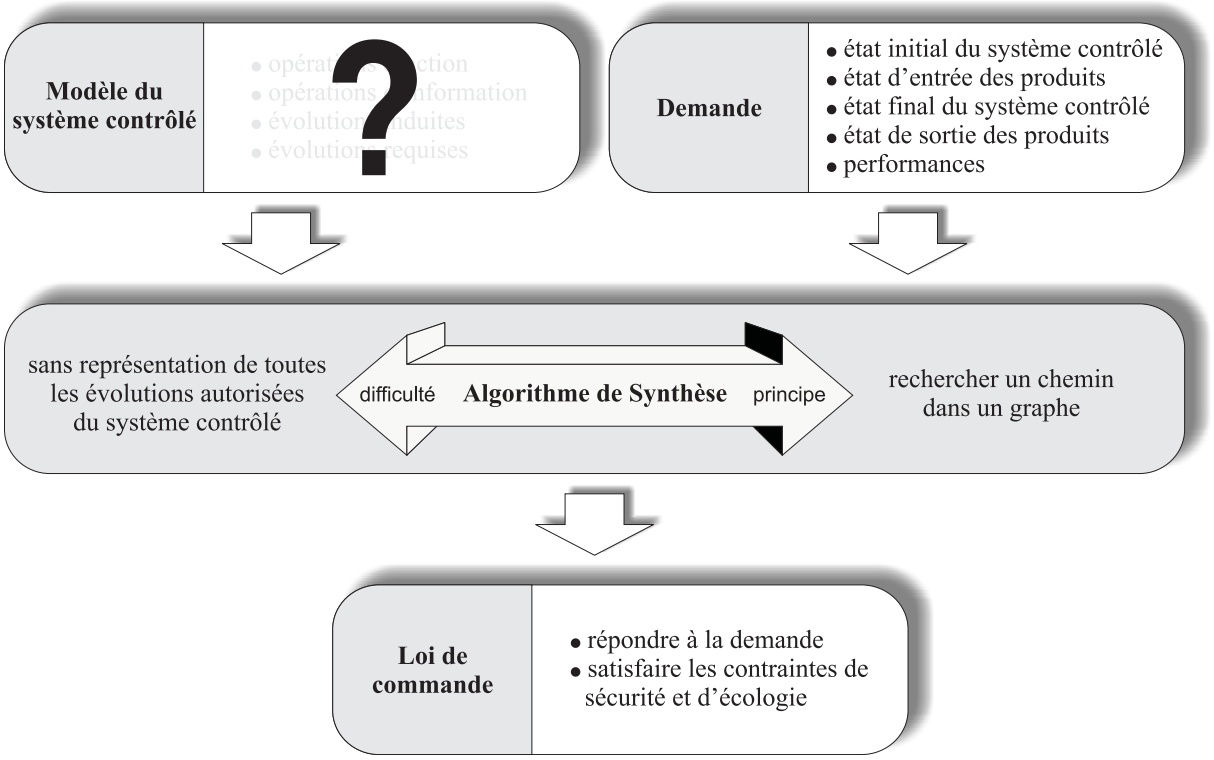


FIG. 3.6 – Démarche de l'approche proposée.

Au vu de la problématique, le rôle de l'algorithme de synthèse de lois de commande est de traduire la demande et les contraintes de sécurité et d'écologie sous la forme de grandeurs de commande : spécification des requêtes à envoyer en fonction des comptes rendus et des informations en provenance de l'environnement.

Dans cette optique et de par la nécessité de définir les évolutions du système contrôlé, nous adoptons une démarche proche de celle utilisée en planification automatique, à savoir, la recherche d'un chemin dans un graphe. Cette recherche de chemin est envisageable soit sans construire le graphe des évolutions satisfaisant les contraintes de sécurité et d'écologie, mais au détriment des performances, soit en générant complètement ce graphe, mais impliquant alors un problème de représentation de l'ensemble des évolutions du système contrôlé satisfaisant les contraintes de sécurité et d'écologie. Toute l'originalité de l'approche proposée du point de vue de l'algorithme de synthèse, présenté dans la troisième partie, est la recherche d'un compromis entre la complexité du graphe construit et les performances obtenues.

Les données d'entrée de l'algorithme sont la demande, présentée précédemment, et le modèle du système contrôlé. Ce modèle reste à définir au travers des informations requises par

l'algorithme de synthèse. Ainsi, le modèle du système contrôlé doit être une représentation : des liens entre les grandeurs de commande et les évolutions du système contrôlé, des informations nécessaires afin de répondre à la demande, et enfin des informations requises pour satisfaire les contraintes de sécurité et d'écologie.

Dans ce cas, le modèle du système contrôlé ne doit pas être une traduction en terme de grandeurs de commande de l'ensemble des demandes possibles et de l'ensemble des contraintes de sécurité et d'écologie. En effet, il serait alors équivalent à une base de recettes de toutes les lois de commande envisageables ; réduisant ainsi l'algorithme à un rôle de sélection. De par le rôle de l'algorithme présenté ci-dessus, nous rejetons alors ce point de vue.

4 Conclusion

Dans le cadre de ce chapitre, nous avons tout d'abord exposé avec précision le périmètre de la problématique que nous traitons dans la suite de ce mémoire. Ce périmètre, décrit en terme d'objectifs d'une loi de commande et de données initiales du problème de conception de loi de commande, nous a conduit à entrer dans le cœur du système de pilotage de chaînes fonctionnelles. Il a ainsi mis en exergue toute la difficulté de générer une loi de commande à un tel niveau. Cette difficulté est tout d'abord liée au mécanisme extrait de la Figure 3.1 page 52 que nous pourrions baptiser "*effet miroir*". Ce dernier montre qu'un module de coordination à pour rôle d'agir sur le flux de produit, mais que pour arriver à cette fin, il ne peut agir que sur les grandeurs de commande des chaînes de fonctionnelles. Il en découle donc une modélisation complexe, devant intégrer divers comportements en interactions permanente. Au delà de cette difficulté de modélisation, ce périmètre nous a montré l'importance, mais aussi la difficulté, de caractériser précisément la demande. Celle-ci doit faire clairement apparaître les états initiaux et finaux du ou des produits qui doivent être transformés, mais également les critères qui permettront à un algorithme de synthèse d'optimiser la loi de commande générée. Enfin, une des dernières difficultés montré dans ce chapitre, naît des différents contextes dans lesquels le mécanisme de conception de loi de commande devra travailler. En effet, en fonction du contexte de configuration ou de reconfiguration avéré, le mécanisme de conception sera plus ou moins contraint dans le temps pour générer une solution. Ainsi, toute l'originalité de l'approche proposée du point de vue de l'algorithme de synthèse, présenté dans la troisième partie, est la recherche d'un compromis entre la complexité du graphe construit et les performances de la solution obtenue.

Fort de cette problématique, ce chapitre a exposé la démarche générale que nous avons retenu pour la traiter. Elle consiste d'une part à proposer une modélisation adéquate du système contrôlé et d'autre part développer un mécanisme général de recherche de chemin dans le modèle proposé afin de satisfaire la demande. Cette démarche structure la suite de ce document.

Deuxième partie

Modèle du Système Contrôlé

Chapitre 4

Caractéristiques du Modèle

1 Introduction

L'approche que nous proposons ici se singularise en abordant de manière complète la problématique de la conception de lois de commande. Ainsi, au delà du mécanisme de conception que nous présenterons dans la partie III de ce mémoire, nous nous proposons tout au long de cette partie de développer non seulement le formalisme de modélisation de systèmes contrôlés que nous préconisons d'utiliser dans les niveaux coordination de chaînes fonctionnelles, mais également de décrire les prémisses d'une méthodologie de modélisation que nous proposons à l'ingénieur méthode.

Ce quatrième chapitre est donc naturellement dédié à la présentation des principales caractéristiques du modèle que nous allons progressivement construire dans les chapitres 5 et 6.

En premier lieu, nous décrirons, avec précision, les sept types d'informations qui doivent être intégrés au modèle. Après quoi, la deuxième section s'attachera à argumenter et à présenter la structuration de ces informations. La troisième section donnera brièvement les propriétés du modèle à partir desquelles le mécanisme de conception de lois de commande pourra voir le jour. Enfin, la dernière section de ce chapitre s'attachera à donner des pistes qui nous conduiront dans le chapitre 5 à spécifier un formalisme de modélisation adapté à nos besoins.

2 Les informations à modéliser

Les objectifs d'une loi de commande, présentés au chapitre précédent, sont résumés dans la Figure 3.5 page 57. Ainsi, afin de concevoir une loi de commande avec ces objectifs, il va être nécessaire de connaître les évolutions qui peuvent être imposées par les grandeurs de commande. En sus de ces informations sur les évolutions, des informations complémentaires sont requises afin de s'assurer d'une part de répondre correctement à la demande et d'autre part de satisfaire les contraintes de sécurité et d'écologie.

Ainsi, nous allons d'abord caractériser l'ensemble des informations représentant le lien entre les grandeurs de commande et les évolutions qu'elles peuvent imposer. Puis, nous étudierons les informations nécessaires afin de s'assurer de répondre correctement à la demande, pour terminer avec celles nécessaires afin de satisfaire les contraintes de sécurité et d'écologie.

2.1 Informations nécessaires afin d'imposer des évolutions

Afin de décider des services auxquels faire appel, il est nécessaire de connaître les effets de ces services sur l'état des chaînes fonctionnelles et des produits. La connaissance de ces effets passe par la modélisation du lien entre les requêtes (et les comptes rendus) avec les évolutions des chaînes fonctionnelles et du flux de produits.

Au delà de ces informations, les liens entre les évolutions de l'environnement, ses effets sur les produits et les informations qu'il fournit doivent être modélisés. La Figure 4.1 résume l'ensemble des informations requises afin de connaître les évolutions qui peuvent être imposées par les grandeurs de commande.

Comme le démontre la section suivante, les informations sur les chaînes fonctionnelles et les produits ne sont pas suffisantes afin de s'assurer de répondre correctement à une demande.

FIG. 4.1 – Modélisation de la relation entre les grandeurs de commande et les évolutions du système contrôlé.

2.2 Informations nécessaires afin de répondre à une demande

S'assurer de répondre correctement à la demande requiert de pouvoir comparer la demande avec le résultat des décisions prises. Tout d'abord, les évolutions présentées au paragraphe précédent suffisent à s'assurer de bien atteindre l'état demandé des chaînes fonctionnelles et des produits. En revanche, les informations actuellement prises en compte dans le modèle ne sont pas suffisantes afin de respecter les performances fixées par la demande.

En effet ceci suppose d'évaluer avec les mêmes critères que ceux de la demande les performances d'un comportement résultant d'un ensemble d'évolutions des chaînes fonctionnelles et des produits. Ainsi suivant le ou les critères considérés, il sera parfois nécessaire d'ajouter au modèle des informations permettant de quantifier les performances.

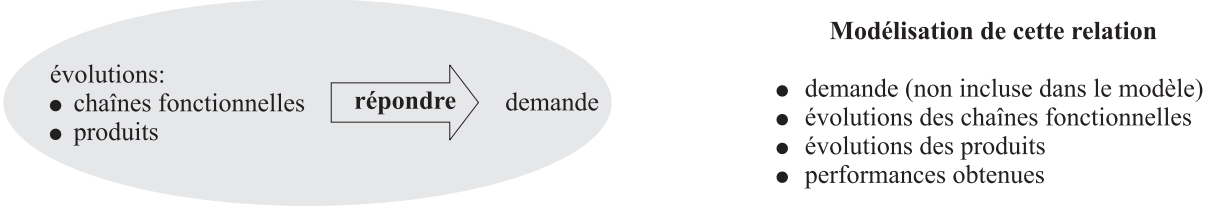


FIG. 4.2 – Modélisation de la relation entre les évolutions du système contrôlé et la demande.

Des critères comme la minimisation du nombre de services auxquels il est fait appel ne requièrent pas d'informations supplémentaires. En revanche, si nous voulons minimiser le temps de cycle d'une loi de commande, il sera nécessaire d'ajouter une durée aux évolutions. Bien que les informations requises pour quantifier un critère soient directement liées au critère et par conséquent à la demande, nous pouvons dire d'une manière générale que les informations relatives aux caractéristiques des évolutions (durée, coût financier, consommation énergétique, etc) devront pouvoir être modélisées.

Les informations à modéliser afin de s'assurer de répondre à la demande sont récapitulées dans la figure 4.2. Pour finir de caractériser l'ensemble des informations à modéliser, la section suivante se focalise sur celles nécessaires afin de satisfaire les contraintes.

2.3 Informations nécessaires afin de satisfaire les contraintes

Disposant des informations pour imposer des évolutions qui puissent répondre à une demande, il reste à satisfaire les contraintes de sécurité et d'écologie. Ces contraintes présentées au § 2.2.3 page 55 sont toutes relatives aux interactions entre les chaînes fonctionnelles contrôlées, les produits et l'environnement.

Ainsi, afin de s'assurer de ne pas atteindre un état interdit, la connaissance des évolutions des chaînes fonctionnelles, des produits et de l'environnement est suffisante. Toutes ces informations sont reprises dans la Figure 4.3.

La section suivante récapitule l'ensemble des informations nécessaires à la conception de lois de commande.

FIG. 4.3 – Modélisation de la relation entre les évolutions du systèmes contrôlé et les contraintes de sécurité et d'écologie.

2.4 Récapitulatif des informations à modéliser

La connaissance des évolutions via les grandeurs de commande, le respect de la demande et la satisfaction des contraintes requièrent toutes les informations présentées précédemment ; elles sont résumées ici :

- les grandeurs de commande (services offerts par les chaînes fonctionnelles contrôlées et informations fournies par l'environnement) ;
- les évolutions des chaînes fonctionnelles ;
- les évolutions des produits ;
- les évolutions de l'environnement ;
- les liens de cause à effet entre les grandeurs de commande et les évolutions précédentes ;
- les contraintes de sécurité et d'écologie ;
- les caractéristiques des évolutions (temporelles, financières, énergétiques, etc).

Suite à la caractérisation des informations à modéliser, nous proposons, dans la section suivante, de les structurer afin d'aboutir à un modèle du système contrôlé.

3 Structuration des informations

Parmi les sept catégories d'informations qui doivent être modélisées, seuls les liens de cause à effet entre les grandeurs de commande et les évolutions n'ont pas été étudiés jusqu'à maintenant. Ainsi, cette section expose les relations de causalité existantes sur lesquelles s'appuie ensuite la structuration des données du modèle.

Afin de caractériser l'ensemble de ces liens, nous allons adopter une double démarche : ascendante et descendante. La démarche ascendante consiste à déterminer les origines d'une évolution de l'état d'un produit (physique ou spatial). La seconde démarche, descendante, est l'analyse des effets d'une part de la réalisation d'un service, et d'autre part d'une action de l'environnement.

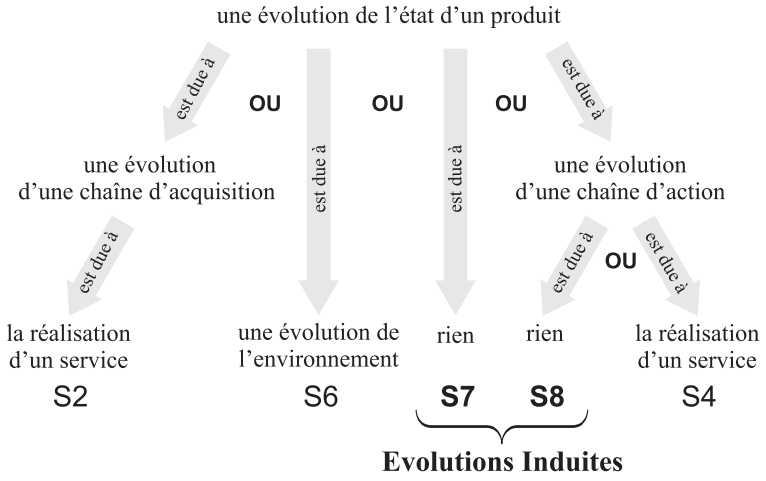


FIG. 4.4 – Origine d'une évolution de l'état d'un produit.

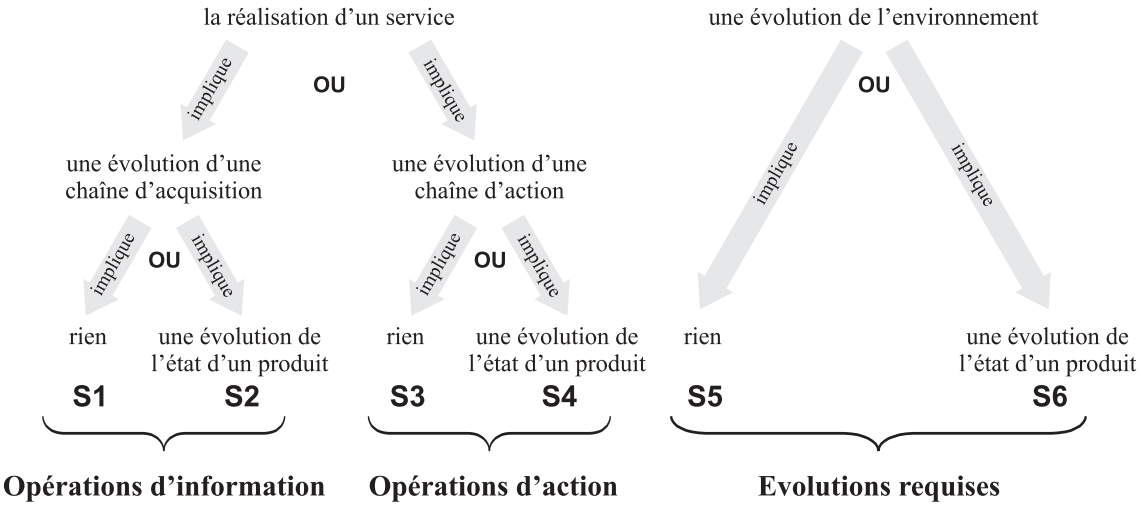


FIG. 4.5 – Effets possibles résultants de la réalisation d'un service ou d'une évolution de l'environnement.

Comme nous pouvons le voir sur les figures 4.5 et 4.4, cette double démarche d'analyse des relations de causalité caractérise huit situations, de S1 à S8, à modéliser. La structuration du modèle choisie est la même que celle classiquement utilisée en ordonnancement ou en planification automatique. Ainsi, ces huit situations seront couvertes par quatre catégories d'opérations : opérations d'action, opérations d'information, évolutions induites et évolutions requises. Les critères à la base de ce regroupement (8 situations, 4 éléments) sont d'une part la capacité du module coordination à déclencher une opération, et d'autre part à la maîtriser.

Ainsi, une opération d'action couvre les situations (S3 et S4), déclenchées et maîtrisées, qui sont basées sur une chaîne d'action. Les deux autres situations (S1 et S2), contrôlées et maîtrisées, basées sur une chaîne d'acquisition sont couvertes par les opérations d'information. Une évolution induite (S7 et S8) correspond à une situation non déclenchée mais maîtrisée puisque le module de coordination pilote la chaîne fonctionnelle à l'origine d'une telle évolution. Et enfin, une évolution requise (S5 et S6) est une situation non déclenchée et non maîtrisée, puisque la chaîne fonctionnelle à l'origine d'une telle évolution n'est pas pilotée par le module de coordination. Les quatre catégories d'opérations composant le modèle sont maintenant détaillées.

3.1 Opération d'action

Une opération d'action est basée sur la réalisation d'un service provoquant une évolution d'une chaîne d'action. En fonction de l'état du flux de produit lors du déclenchement de l'opération, il y aura simultanément ou non une évolution du flux de produits. Le déclenchement d'une opération se fait par l'envoi d'une requête à laquelle correspond un compte rendu de fin renvoyé par la chaîne fonctionnelle offrant le service.

3.2 Opération d'information

Une opération d'information fait appel à un service offert par une chaîne fonctionnelle composée uniquement d'une chaîne d'acquisition. La réalisation du service provoque, du point de vue du niveau coordination, une évolution du flux de produits par une connaissance plus précise de son état. Basé sur un service offert par une chaîne fonctionnelle, le déclenchement d'une opération d'information se fait par une requête à laquelle correspond toujours un compte rendu de fin du service.

3.3 Évolution induite

Une évolution induite est une évolution de l'état d'un ou plusieurs produits depuis un état, dit instable, des produits. Cet état instable est atteint suite à l'exécution d'opérations d'action ou d'évolutions requises. N'étant pas basée sur la réalisation d'un service offert par une chaîne fonctionnelle, une évolution induite n'est pas déclenchée par une requête d'appel à un service.

3.4 Évolution requise

Une évolution requise est une évolution d'une chaîne fonctionnelle de l'environnement, avec ou sans effet sur des produits, sans laquelle il peut être impossible de répondre à la demande.

Les évolutions requises modélisent les informations indispensables qu'un module coordination doit connaître de son environnement (da Silveira, 2003). En effet, de par les interactions entre le système contrôlé et son environnement dans les espaces partagés (cf. Figure 3.3), la

réalisation de services offerts par les chaînes fonctionnelles contrôlées est contrainte par l'état de l'environnement et des produits sur lesquels il agit.

La réalisation d'une évolution requise dépend de l'environnement et de sa propre stratégie. Le niveau coordination considéré ne pilotant pas son environnement, il ne peut pas déclencher une évolution requise. Seules les informations qui lui seront fournies par l'environnement permettront dans certains cas de connaître le début et la fin d'une évolution requise. C'est la situation, par exemple, d'attente de l'arrivée d'un produit dans un stock d'entrée.

Le choix de l'outil de modélisation qui est conditionné par les informations à modéliser et leur structuration dépend aussi en partie des propriétés du modèle présentées dans la section suivante.

4 Propriétés du modèle

Le modèle défini par l'ensemble des informations structurées en quatre catégories d'opérations possède les propriétés suivantes : parallélisme et concurrence entre les opérations, synchronisation, et enfin déterminisme et indéterminisme. Si les trois premières sont celles d'une loi de commande, l'indéterminisme est en revanche spécifique au modèle. Nous traiterons dans cette section le déterminisme et l'indéterminisme qui sont liés.

Le déterminisme concerne les effets sur l'état des chaînes fonctionnelles et des produits des opérations d'action, des évolutions induites et requises. En effet, en l'absence d'aléa, les évolutions des chaînes fonctionnelles et du flux de produits résultantes d'une des trois catégories d'opérations citées précédemment sont déterministes au sens où, depuis un état, l'opération conduit dans un seul et unique état. Du point de vue des chaînes fonctionnelles, ce déterminisme traduit également la relation de cause à effet entre les requêtes émises par le système de commande et les évolutions résultantes des chaînes fonctionnelles en l'absence d'aléa.

L'indéterminisme résulte quant à lui d'une part des effets des opérations d'information, et d'autre part des combinaisons possibles entre les opérations afin d'atteindre un état donné.

Basée sur un service offert par une chaîne d'acquisition, une opération d'information renseigne le module de coordination de l'état du flux de produits. Le recours à une opération d'information s'avère nécessaire quand il existe une incertitude sur l'état d'entrée du flux de produits. Aussi, l'état atteint à la fin de l'opération d'information sera fonction des données accompagnant le compte rendu de fin de l'opération. Il existe ainsi un indéterminisme sur l'état atteint suite à l'exécution d'une opération d'information.

En sus de cet indéterminisme, les opérations à exécuter et leur enchaînement afin d'atteindre un état souhaité est indéterministe de par la flexibilité physique du système de production. Sans cet indéterminisme, il n'y aurait plus de problème d'optimisation puisque la solution serait unique.

Les informations à modéliser, leur structuration et les propriétés du modèle étant maintenant définies, il nous reste à choisir un outil de représentation. Ainsi, la section suivante vise à établir l'outil le mieux adapté.

5 Outil de Modélisation

En premier lieu, l'outil de modélisation doit permettre la représentation des informations avec la structuration proposée et les propriétés qui en découlent. Mais la seule considération de ces quatre aspects n'est pas suffisante afin de déterminer l'outil de modélisation le mieux adapté. Ainsi en plus de ces aspects, nous considérerons, par ordre d'importance, les trois critères suivants :

1. guider au maximum l'expert chargé de réaliser le modèle (aspect graphique ou non, structuration du modèle, etc) ;
2. extraire facilement les différentes données dont l'algorithme de synthèse aura besoin ;
3. faciliter la validation du modèle.

La difficulté majeure liée à la modélisation est l'obtention de l'ensemble des informations que seul l'expert est en mesure d'apporter, d'où l'importance de ce critère. Ainsi, l'outil de modélisation doit être une aide afin de collecter, de la façon la plus intuitive et naturelle possible pour l'expert, l'ensemble des informations. En résumé, l'outil doit faciliter l'assimilation par l'expert des capacités de la partie opérative. De plus, l'outil se doit de ne pas introduire de difficultés supplémentaires liées à son utilisation et à son formalisme.

Pour ces raisons, nous avons choisi une modélisation par opération. Cet outil de modélisation, comme nous le verrons au cours de cette partie, offre l'avantage de ne pas nécessiter une vision globale du système contrôlé, mais impose à l'expert de se focaliser sur une chaîne fonctionnelle et son environnement.

Tel que nous le verrons dans la quatrième partie consacrée à une étude de cas, le recours à un outil de type automate ou réseaux de Petri engendre, dans ce cas, une complexité telle que tout l'intérêt de l'aspect graphique de ces outils est perdu. En effet, l'étude que nous avons menée sur l'utilisation de l'outil réseaux de Petri (Deschamps, 2004) et qui ne sera pas reprise ici pour des raisons de concision a conduit à ces conclusions.

Néanmoins, le choix de l'outil que nous avons fait n'interdit pas à des fins de validation du modèle une représentation basée sur un autre outil tel que les réseaux de Petri. Il s'agit alors de transformer le modèle comme nous l'avons proposé dans (Deschamps et al., 2004).

6 Conclusion

Dans ce chapitre, nous nous sommes attachés à présenter les caractéristiques fondamentales que le modèle doit posséder. La démarche de recherche de ces caractéristiques a été largement orientée par le point de vue *effet miroir* (cf. Figure 3.1 page 52) que nous avons des modules de coordination, des chaînes fonctionnelles et de leur rôle sur le flux de produit. Aussi, nous avons proposé un balayage systématique de toutes les informations à capitaliser dans le modèle, en partant des grandeurs de commande offertes aux modules de coordination jusqu'aux informations quantitatives liées aux évolutions des chaînes fonctionnelles, des produits et de l'environnement, qui permettra à terme d'évaluer les performances de la solution proposée. Au delà de la mise en exergue des sept types d'informations à modéliser, nous avons également proposé une structuration du modèle. Celle-ci, argumentée sur la base du principe de causes à effets existant entre les grandeurs de commande et les évolutions s'articule autour de quatre

catégories d'opérations : les opérations d'action, les opérations d'information, les évolutions induites et enfin les évolutions requises. Après avoir discuté des propriétés intrinsèques au modèle, notamment sur le plan de son indéterminisme natif, nous avons terminé ce chapitre par une spécification des besoins du formalisme requis pour représenter la structure complexe de données proposée.

Les caractéristiques fondamentales du modèle étant désormais avancées, nous nous proposons de présenter, dans le détail, le formalisme que nous avons développé.

Chapitre 5

Le modèle du Système Contrôlé

1 Introduction

Ce chapitre se veut être véritablement le cœur de cette partie. Il va s'attacher à décrire, avec le souci permanent de fouiller, d'argumenter, ou encore de préciser toutes les notions qui sont à la base du formalisme amené et donc de la modélisation des systèmes contrôlés que nous proposons. De fait, il en résulte une rédaction qui pourra parfois sembler dense et énumérative, mais il s'agit d'une étape incontournable qui est à la base de la méthode de capitalisation des connaissances que nous proposons.

Compte tenu de la complexité liée à la présentation d'un formalisme, la première section de ce chapitre propose un cas d'étude sur lequel nous allons nous appuyer pour illustrer l'ensemble des notions proposées. Ainsi, la deuxième section nous amènera progressivement à poser le concept d'opération d'action dérivé de celui utilisé en Planification Automatique. La section trois décrira alors avec précision le modèle générique d'une opération d'action pour développer, dans la section quatre, la notion de comportement (Henry et al., 2004b). Sur la base du modèle comportemental d'une telle opération, les sections 5, 6 et 7 exposent respectivement les modèles des opérations d'information, des évolutions induites et requises. La section 8 termine ce chapitre par une présentation des interactions entre ces quatre catégories d'opérations.

2 Cas d'étude

Cette section présente un exemple d'application basé sur le système d'approvisionnement d'un robot en pièces identifiées. L'application est un sous-système de la plate forme de recherche SAPHIR du Laboratoire d'Automatique de Grenoble (France). La plate-forme SAPHIR est dédiée à l'assemblage d'arbres à cames. Le système d'approvisionnement, présenté dans la Figure 5.1 se compose de sept chaînes fonctionnelles : un détecteur de présence en A, un magasin rotatif, un poste d'identification, un convoyeur à bande et trois vérins. Le magasin rotatif avec huit emplacements est utilisé pour recevoir jusqu'à six types différents de pièces. Un opérateur humain approvisionne aléatoirement le magasin rotatif tant du point de vue du nombre des pièces déposées que de leur type. Un vérin 1 transfère les pièces du magasin rotatif au poste d'identification qui est basé sur un système de pesée. Une fois un produit identifié, un vérin 2 le déplace sur un convoyeur central. Pour conduire les pièces vers le robot d'assemblage des arbres à cames, le convoyeur central sert de stock tampon grâce à un vérin de blocage, le vérin 3.

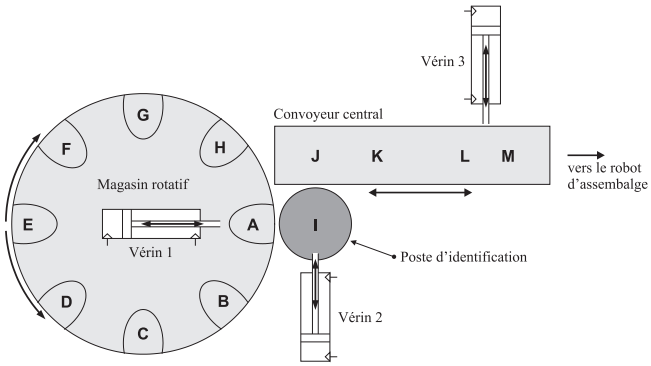


FIG. 5.1 – Système d'approvisionnement du robot en pièces identifiées.

Notons que dans ce synoptique les points sont fixes. Cela signifie que la rotation du magasin a pour effet de déplacer un produit qui est en A entre A et B, puis en B, et ainsi de suite, de même pour les points J, K, L et M du convoyeur central.

Le système présenté est formé de cinq chaînes fonctionnelles avec des actionneurs : le magasin rotatif avec son moteur électrique, le vérin 1, le vérin 2, le convoyeur central et le vérin 3. Le poste d'identification et le capteur de présence en A constituent la sixième et la septième chaînes fonctionnelles qui sont basées sur une chaîne d'acquisition (par exemple pour l'identification : jauge de contrainte, traitement du signal puis comparaison à une base de données). Deux modules de coordination pilotent ces sept chaînes fonctionnelles. Le premier module coordonne le magasin rotatif, la détection de présence de pièce en A, le vérin 1, le poste d'identification et le vérin 2. Les deux autres chaînes fonctionnelles, le convoyeur central et le vérin 3, sont pilotées par le deuxième module de coordination (cf. Figure 10.3 page 174).

Cette application sert maintenant d'illustration à la présentation du modèle d'une opération d'action.

3 Vers un modèle d'une opération d'action

Pour aboutir à un modèle générique d'une opération d'action, un dénominateur commun aux données requises est proposé. A cet égard, le modèle d'une opération utilisé en planification automatique servira de base à notre réflexion sur le modèle d'une opération d'action. La proximité avec notre problématique des travaux existants en planification automatique présentée au § 2 page 37 et l'outil de représentation du modèle choisi au § 5 page 69 ont dicté ce choix. En effet, c'est la richesse du modèle d'une opération par la description de son effet sur le système contrôlé qui permet de décider des opérations à exécuter.

Cette section vise alors, à partir du modèle d'une opération utilisé en planification automatique et appliqué au système d'approvisionnement, à définir la structure et les champs données d'une opération d'action qui répondent d'une part à la problématique de conception d'une loi de commande, et d'autre part au fort besoin de guider l'expert dans sa démarche de modélisation. Après avoir présenté le modèle d'une opération en planification automatique, nous étudions ses limites pour la conception de lois de commande et proposons son extension pour répondre à nos besoins.

3.1 Concept d'opérations en planification automatique

En sus du modèle classique et afin de tenir compte des parallélismes, le modèle d'une opération est étendu conformément à la proposition de (Sandewall et Rönnquist, 1986) sur laquelle sont basés les travaux pour la génération de loi de commande (Klein, 1999), (Castillo, 2000) et (Aylett, 2001). Une opération se compose de : l'effet final sur le système contrôlé, une pré-contrainte, une contrainte et une post-contrainte à satisfaire respectivement avant, pendant et après l'action (cf. Figure 5.2). Pour faciliter leur lecture, les modèles des opérations sont donnés sous la forme de tableaux. Pour illustrer nos propos, les opérations liées à la sortie du vérin 1 du système d'approvisionnement servent de base à l'étude. Il s'agit de la sortie du vérin 1 sans effet sur le flux de produits et de l'opération de sortie du vérin avec un déplacement d'une pièce (P) de A vers I.

La Figure 5.2 présente l'opération de sortie du vérin 1 sans effet sur le flux de produits. Depuis un état du système contrôlé et de son environnement pour lequel la pré-contrainte est satisfaite, l'action *"Sortir Vérin 1 sans effet sur le flux de produits"* peut être lancée. L'effet de l'action est limité à une évolution du vérin 1 de l'état rentré à l'état sorti. La contrainte est utilisée pour vérifier le parallélisme possible de cette opération avec d'autres. Par exemple, une rotation du magasin qui amènerait une pièce entre A et B risque d'entraîner une collision de cette pièce avec le vérin 1. Aussi, pendant cette opération, soit le plateau est arrêté soit il n'y a pas de pièce en B et il peut alors tourner d'un huitième de tour dans le sens trigonométrique.

L'action "Sortir Vérin 1 avec effet sur le flux de produits" de la Figure 5.3 déplace une pièce (P) de la position A sur le magasin vers la position I sur le poste d'identification. La pré-contrainte est ici différente de l'action précédente au sens où elle porte sur l'état arrêté et indexé du magasin plutôt que sur l'absence de pièce entre A et B, et entre A et H. En effet, avec initialement une pièce en A, il ne peut pas y avoir de pièce entre A et B, et entre H et A de par la géométrie du magasin. De plus, il existe une interaction entre le magasin et la pièce quand celle-ci est en A ou entre A et I.

Afin de proposer un modèle d'une opération d'action, ces deux opérations, basées sur le modèle utilisé en planification automatique, sont maintenant étudiées vis-à-vis des besoins afin de concevoir une loi de commande.

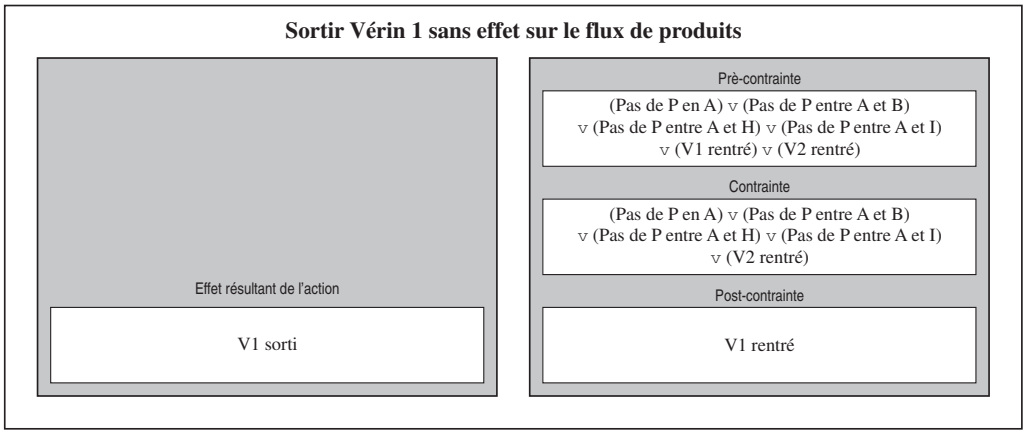


FIG. 5.2 – Exemple d'une opération d'action sans effet sur le flux de produits.

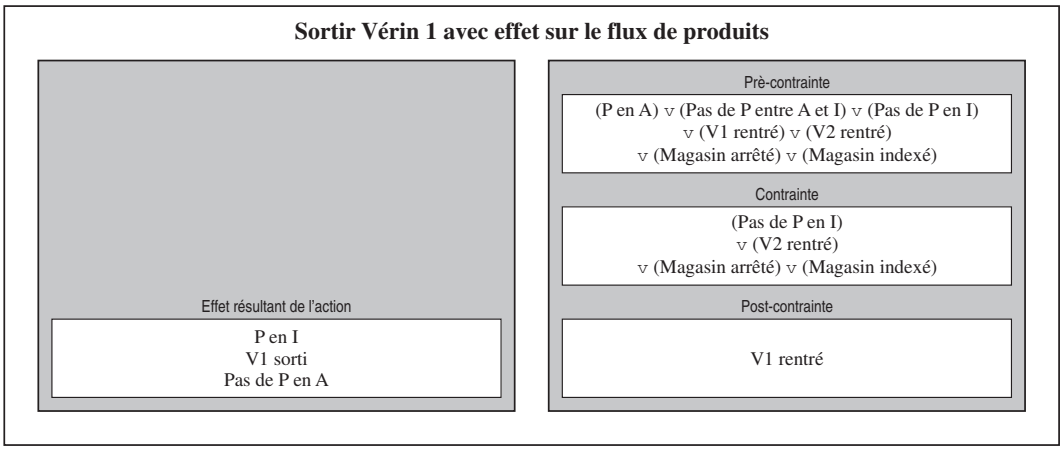


FIG. 5.3 – Exemple d'une opération d'action avec effet sur le flux de produits.

3.2 Limites du modèle d'une opération

Initialement, comme nous l'avons vu au chapitre 2, § 2, les travaux en planification automatique ont essentiellement visés des domaines comme la robotique. Ils se sont progressivement étendus à d'autres domaines d'application. Mais ce n'est que très récemment qu'ils se sont intéressés aux systèmes de production et plus particulièrement à la conception de lois de commande. Les travaux peu nombreux sur ce domaine d'application ((Klein, 1999), et (Castillo, 2000), (Aylett, 2001)) ont, du point de vue de la modélisation, consisté uniquement à étendre le modèle d'une opération par l'ajout d'une contrainte à respecter durant son exécution afin de vérifier certains parallélismes. Les travaux se sont en fait essentiellement focalisés sur les algorithmes de synthèse d'un comportement des chaînes fonctionnelles sans apporter une véritable étude quant au besoin spécifique de modélisation dans le contexte des systèmes de production. L'ensemble des incohérences et des difficultés liées au modèle d'une opération est énuméré ci-dessous et illustré avec les deux opérations présentées dans la section précédente.

1. Le modèle proposé d'une opération ne modélise pas l'état transitoire durant son exécution. Il est alors impossible de vérifier une pré-contrainte ou une contrainte quand elle porte sur un état transitoire. Par exemple, l'opération de rotation d'un huitième de tour du magasin ne modélise pas l'état transitoire du flux de produits résultant de son exécution. Or cet état peut conduire à la présence d'une pièce entre A et B. Aussi, la pré-contrainte et la contrainte des actions sortir vérin 1 (cf. Figure 5.2 et 5.3 portant sur l'absence de pièce entre A et B) apparaîtront toujours comme satisfaites. Finalement, les contraintes de sécurité et d'écologie risquent d'être violées sans que le modèle ne permette de le constater.
2. Dans la post-contrainte, l'état vérin 1 rentré comme le propose (Castillo, 2000) n'a pas à être imposé. Cette post-contrainte sert en effet uniquement à toujours imposer l'existence de l'opération de rentrée du vérin après l'opération de sortie. Or, il n'existe aucune raison d'imposer de toujours devoir rentrer le vérin après l'avoir sorti dans une même loi de commande ceci notamment lors d'une reprise.
3. Il n'est pas fait de différence entre ce qui relève des contraintes de sécurité et/ou d'écologie et ce qui relève des conditions liées à l'obtention des évolutions. La pré-contrainte "pièce en A" n'est pas liée à une contrainte de sécurité et d'écologie. En effet, en l'absence de pièce en A, il ne s'agira plus de l'opération sortir vérin 1 avec effet sur le flux de produits mais

de l'opération sortir vérin 1 sans effet sur le flux de produits. Il n'y a donc pas violation d'une contrainte de sécurité et/ou d'écologie mais uniquement une condition non satisfaite. Cette différence d'origine des propositions atomiques, composant la proposition logique d'une pré-contrainte, est une source de confusion et d'erreurs pour l'expert en charge de la modélisation.

4. Du point de vue du modèle global composé de l'ensemble des opérations, il y a redondance des données modélisant l'évolution des chaînes fonctionnelles. La sortie du vérin 1 est en effet modélisée dans les deux opérations de sortie du vérin des figures 5.2 et 5.3.
5. Le nombre d'opérations basées sur un même service peut devenir très important. Pour le service sortir vérin 1, il y a trois opérations basées sur ce service, les deux présentées précédemment plus celle où l'effet sur le flux de produits déplace une pièce initialement entre A et I vers I. Trois opérations semblent être un nombre raisonnable d'opérations à modéliser pour un service. En revanche pour le service de rotation d'un huitième de tour du magasin, 256 opérations sont à modéliser (cf. démonstration § 5.1 page 78) ce qui est un obstacle pour l'expert en charge de la modélisation.
6. Il n'y a pas de champ de données pour quantifier les critères décisionnels à des fins d'optimisation. En effet, il s'agit avant tout, en planification automatique, de trouver une séquence de façon à satisfaire une demande qui spécifie uniquement un état à atteindre.
7. Les grandeurs de commande ne sont pas intégrées au modèle d'une opération car la structure du système de commande et plus particulièrement du niveau local avec les chaînes fonctionnelles n'est pas prise en compte. En effet, une loi de commande spécifie les services demandés et à qui les demander (une des chaînes fonctionnelles contrôlées).

Dans la section suivante, la proposition d'une solution est faite pour chacun des sept points énoncés ci-dessus.

3.3 Propositions

Des propositions d'ajout ou de suppression de champ de données apportent une solution à chacun des sept points énoncés.

1. En modélisant l'effet transitoire des opérations, il est alors possible de s'assurer de satisfaire les contraintes.
2. Le champ post-contrainte est supprimé. En effet, il n'existe aucune raison d'imposer une contrainte après l'opération puisque l'effet de cette dernière sur le système contrôlé est terminé et qu'elle ne peut donc plus conduire dans un état violant des contraintes de sécurité et d'écologie. Ceci est bien sûr valable à condition de voir la post-contrainte uniquement avec une vision sécurité et écologie et de disposer d'un champ particulier dédié à la description de l'effet de l'opération sur l'état du système contrôlé.
3. Le rôle du champ pré-contrainte se limite à contraindre l'état du système contrôlé tel que le lancement de l'opération satisfasse les contraintes de sécurité et d'écologie. Au delà de la pré-contrainte, une condition sur les chaînes fonctionnelles et le flux de produits doit être initialement satisfaite pour que les effets décrits aient lieu.
4. L'évolution de la chaîne fonctionnelle obtenue par l'appel à un service est le facteur commun à toutes les opérations basées sur un même service. Afin de supprimer la redondance

des données, toutes les opérations sont regroupées en factorisant l'évolution de la chaîne fonctionnelle.

5. La factorisation proposée au point précédent fait correspondre une et une seule opération à un service. Elle évite ainsi un trop grand nombre d'opérations pour un même service, comme l'exemple précédent de l'indexation du plateau.
6. Un champ de données est créé pour les caractéristiques (temporelles, financières, énergétiques) de l'opération.
7. Les données nécessaires à la connaissance du service à appeler et de la chaîne fonctionnelle offrant ce service sont ajoutées dans un champ de données appelé grandeurs de commande.

Sur la base de ces sept propositions, l'opération de la Figure 5.4 représente les trois opérations associées au service *Sortir Vérin 1*. Elle contient notamment les deux opérations dont les modèles ont été donnés dans les figures 5.2 et 5.3.

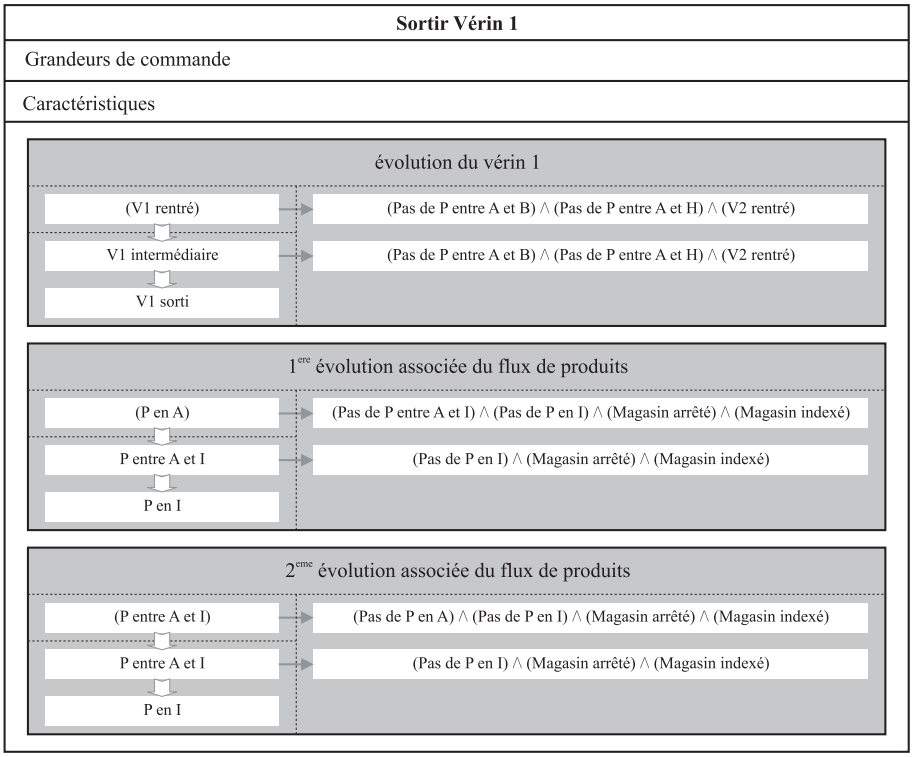


FIG. 5.4 – Modèle de l'opération d'action Sortir Vérin 1 avec ou sans effet sur le flux de produits.

4 Modèle générique d'une opération d'action

Suite à l'étude menée dans la section précédente, les champs de données d'une opération d'action, présentés dans la Figure 5.5, sont : l'identificateur de l'opération, les grandeurs de commande pour le niveau coordination, les caractéristiques de l'opération afin d'évaluer les performances des lois de commande conçues, et la dynamique du système contrôlé. Ce champ de données est explicité dans la suite de cette section.

La dynamique du système contrôlé est modélisée au sein d'une opération d'action i, notée OA_i, par deux types d'évolutions :

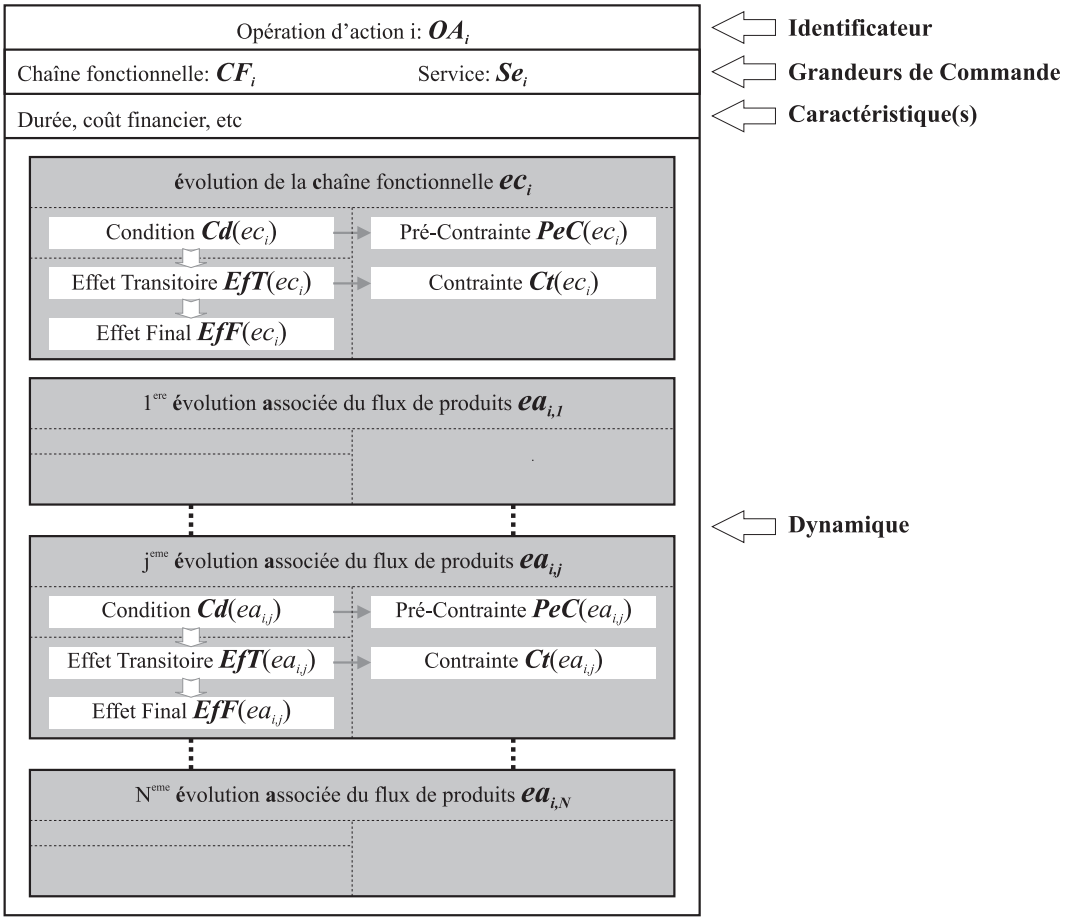


FIG. 5.5 – Champs de données et notations du modèle générique d'une opération d'action.

L'évolution de la chaîne fonctionnelle, notée ec_i, dont l'exécution de l'opération implique toujours l'existence. Elle se caractérise par cinq champs de données :

- *une condition*, notée $Cd(ec_i)$, sur l'état initial de la chaîne fonctionnelle. Si cette condition n'est pas satisfaite, le service à la base de l'opération ne peut pas être demandé. L'écriture de la condition est basée sur la logique de proposition (ou logique d'ordre 0).
- *l'effet transitoire*, noté $EfT(ec_i)$, sur l'état de la chaîne fonctionnelle durant l'opération. Il affecte une valeur particulière à une ou plusieurs variables d'état de la chaîne fonctionnelle sur laquelle l'opération d'action est basée.
- *l'effet final*, noté $EfF(ec_i)$, sur l'état de la chaîne fonctionnelle suite à l'opération. Il affecte une valeur particulière à une ou plusieurs variables d'état de la chaîne fonctionnelle sur laquelle l'opération d'action est basée. $EfF(ec_i)$ avec $EfT(ec_i)$ forment l'effet sur la chaîne fonctionnelle de ec_i.
- *la pré-contrainte*, notée $PeC(ec_i)$, à satisfaire sur l'état du flux de produits et des chaînes fonctionnelles avant le début de l'opération si la condition $Cd(ec_i)$ est vraie. L'écriture de la pré-contrainte est également basée sur la logique de proposition.
- *la contrainte*, notée $Ct(ec_i)$, à satisfaire sur l'état du flux de produits et des chaînes fonctionnelles durant l'opération. Elle est également spécifiée par une proposition logique.

Les évolutions associées du flux de produit, notée $ea_{i,j}$ avec $j \in [1, N]$ pour une opération d'action avec N évolutions associées. Elle se caractérise par cinq champs de données :

- *une condition*, notée $Cd(ea_{i,j})$, sur l'état du flux de produits et des chaînes fonctionnelles depuis lequel l'exécution de l'opération implique l'effet de $ea_{i,j}$. Si cette condition n'est pas satisfaite, l'exécution de l'opération d'action est tout de même possible mais sans l'effet de $ea_{i,j}$.
- *l'effet transitoire*, noté $EfT(ea_{i,j})$, sur l'état du flux de produits durant l'opération. Il affecte une valeur particulière à une ou plusieurs variables d'état du flux de produits.
- *l'effet final*, noté $EfF(ea_{i,j})$, sur l'état du flux de produits suite à l'opération. $EfF(ea_{i,j})$ avec $EfT(ea_{i,j})$ forment l'effet sur le flux de produits de $ea_{i,j}$.
- *la pré-contrainte*, notée $PeC(ea_{i,j})$, à satisfaire sur l'état du flux de produits et des chaînes fonctionnelles avant le début de l'opération si la condition $Cd(ea_{i,j})$ est vraie.
- *la contrainte*, notée $Ct(ea_{i,j})$, à satisfaire sur l'état du flux de produits et des chaînes fonctionnelles durant l'opération.

L'évolution de la chaîne fonctionnelle et les évolutions associées du flux de produits représentent généralement plusieurs comportements de l'opération d'action en fonction de son contexte d'exécution. Ainsi pour lever toute ambiguïté d'interprétation et pour servir de base à la validation du modèle présentée au chapitre 8, les comportements d'une opération d'action sont maintenant formalisés.

5 Les comportements d'une opération d'action

Comme nous l'avons montré précédemment, une opération d'action modélise en fonction de l'état des chaînes fonctionnelles et du flux de produits des évolutions différentes du système contrôlé (chaînes fonctionnelles contrôlées et flux de produits sur lequel elles peuvent agir). Ainsi, nous parlerons d'un comportement de l'opération en fonction de son contexte d'exécution. Un comportement d'une opération d'action est alors obtenu uniquement pour un ensemble d'états initiaux particuliers. La définition de ces états requiert au préalable la définition d'une notation des comportements d'une opération.

5.1 Notation d'un comportement

Cette section introduit pour un comportement d'une opération d'action une notation simple et explicite qui contient les évolutions associées du flux de produits obtenues.

Le comportement d'une opération d'action OA_i est caractérisé par un sous-ensemble d'évolutions associées obtenues, noté $EA_{i,k}$, de l'ensemble des évolutions associées, noté $EA_i = \{ea_{i,j}\}_{j\in[0,N]}$. Il est défini par :

$$EA_{i,k} = \{ea_{i,j}/j \in J_k \wedge k \in [0, 2^N - 1]\}, EA_{i,k} \subset EA_i$$

Pour avoir une notation à la fois concise et explicite quant aux évolutions associées obtenues, l'indice k, qui est l'identificateur d'un comportement d'une opération, spécifie l'ensemble J_k des indices j des évolutions du flux de produits caractérisant le comportement. En considérant la valeur de l'entier k en binaire, J_k est l'ensemble des puissances des poids qui valent un. Ainsi, pour un comportement k, une évolution associée $(ea_{i,j})$ est obtenue, si son indice j appartient à

l'ensemble J_k défini par :

$$J_k = \{x/(k = \sum_{y=0}^{N-1} c_y 2^y / c_y \in [0,1] \wedge y \in \mathbb{N}) \wedge (c_y = 1 \Rightarrow x = y+1)\}$$

Pour un sous-ensemble $EA_{i,k}$, le comportement d'une opération d'action se compose des évolutions associées du flux de produits appartenant à $EA_{i,k}$ et toujours de l'évolution de la chaîne fonctionnelle qui est la base de l'opération. Ainsi, le comportement d'une opération d'action, noté $COA_{i,k}$, est défini par :

$$COA_{i,k} = \{ec_i, EA_{i,k}\}$$

L'ensemble des comportements d'une opération d'action OA_i, noté $COA_i = \{COA_{i,k}\}$, contient au maximum toutes les combinaisons avec de 0 à N évolutions associées du flux de produits. Pour n évolutions associées, il existe C_N^n combinaisons. Le nombre maximum de comportements d'une opération est alors la somme des combinaisons pour $n \in [0, N]$.

$$\text{Card}(COA_i) = \text{Card}(EA_{i,k}) = \sum_{n=0}^{N} C_N^n = 2^N$$

Ainsi pour un magasin rotatif pouvant contenir de 0 à 8 produits, le nombre de comportements d'une opération d'indexation du magasin (cf. FigureC.2) est égale à $2^8 = 256$ comportements possibles de l'opération d'action. Cette opération qui représente 256 comportements comporte une évolution de la chaîne fonctionnelle et 8 évolutions associées du flux de produits. Ceci est à comparer au 256 opérations obtenues avec la modélisation utilisée en planification automatique.

Appliquons maintenant le système de notation proposé à un exemple dont le nombre de comportements est plus réduit ; l'opération d'action *"Sortir Vérin 1"* présentée dans la Figure 5.4. Cette opération d'action a deux évolutions associées du flux de produits : $ea_{SV1,1}$, $ea_{SV1,2}$. L'une déplace un produit de A vers I, et l'autre transfère un produit qui est entre A et I, en I. Ainsi, l'ensemble des évolutions associées du flux de produits de l'opération (OA_{SV1}) comporte deux éléments :

$$EA_{SV1} = \{ea_{SV1,1}, ea_{SV1,2}\}$$

Il existe quatre combinaisons possibles de ces deux évolutions associées du flux de produits, données par les ensembles J_k suivants :

$J_0 = \{\emptyset\}$ *pour* $k = 0$, $J_1 = \{1\}$ *pour* $k = 2^0$, $J_2 = \{2\}$ *pour* $k = 2^1$, $J_3 = \{1,2\}$ *pour* $k = 2^0 + 2^1$

Finalement, les quatre comportements de l'opération d'action (OA_{SV1}) sont :

- $COA_{SV1,0} = \{ec_{SV1}\}$, aucune évolution du flux de produits n'est associée à l'évolution de la chaîne fonctionnelle,
- $COA_{SV1,1} = \{ec_{SV1}, ea_{SV1,1}\}$, simultanément à l'évolution de la chaîne fonctionnelle un produit est déplacé de A vers I,
- $COA_{SV1,2} = \{ec_{SV1}, ea_{SV1,2}\}$, simultanément à l'évolution de la chaîne fonctionnelle un produit entre A et I est déplacé en I,
- $COA_{SV1,3} = \{ec_{SV1}, ea_{SV1,1}, ea_{SV1,2}\}$, ce comportement aurait pour effet de déplacer simultanément une pièce de A vers I, et une pièce entre A et I vers I. Cette deuxième pièce chuterait du poste d'identification suite à la collision avec l'autre pièce. Au vu des pré-contraintes de l'opération (cf. Figure 5.4), il n'existe pas d'état initial depuis lequel l'exécution de l'opération a ce comportement.

Comme nous venons de le voir pour le quatrième comportement de l'opération de sortie du vérin 1, tous les comportements d'une opération d'action ne peuvent pas être obtenus. Ainsi, la section suivante établit l'ensemble des états initiaux depuis lesquels un comportement d'une opération est possible.

5.2 Un comportement d'une opération d'action

Afin de ne pas violer les contraintes de sécurité et d'écologie, une opération d'action avec un comportement particulier n'est pas autorisée depuis n'importe quel état du système contrôlé et de son environnement. Afin d'établir l'ensemble des états initiaux, un système de transition d'états D, appelé "le domaine" en planification automatique, est associé aux évolutions du système contrôlé. Le système de transitions d'états D, est un triplet $D = (Q_S, \Sigma, \delta)$ où :

- Q_S est l'ensemble d'états représentables du système contrôlé. Il est défini par $Q_S = \prod_{x \in VE} V_x$ avec V_x l'ensemble des valeurs de la variable d'état x, et VE l'ensemble des variables d'états choisies pour modéliser le système contrôlé. Un état $q \in Q_S$ est défini par $q = \{(x = c) | x \in VE\}$, où $c \in V_x$;
- Σ est l'ensemble des grandeurs de commande ;
- $\delta : Q_S \times \Sigma \rightarrow Q_S$ est la fonction de transitions d'états définie de $Q_S \times \Sigma \rightarrow Q_S$ qui associe un état d'arrivée à un état de départ et à une grandeur de commande.

Un comportement d'une d'opération d'action est possible uniquement depuis un sous-ensemble de Q_S donné par la définition 1.

Définition 1 *L'ensemble des états initiaux du comportement $COA_{i,k}$, noté $Q_{It(COA_{i,k})}$, est l'ensemble des états depuis lesquels l'occurrence de l'événement de début de l'opération d'action OA_i est autorisé avec le comportement $COA_{i,k}$. Formellement, cet ensemble est représenté par :*

$$Q_{It(COA_{i,k})} = \{q \in Q_S / Cd(ec_i) \wedge PeC(ec_i) \bigwedge_{j \in J_k} [Cd(ea_{i,j}) \wedge PeC(ea_{i,j})] \bigwedge_{j \in \overline{J_k}} \neg Cd(ea_{i,j}) = Vraie\}$$

Dans un état depuis lequel l'exécution d'une opération d'action OA_i est autorisée et implique un comportement $COA_{i,k}$, les conditions et les pré-contraintes de l'évolution de la chaîne fonctionnelle et des évolutions associées du flux de produits obtenues sont vraies. En revanche, les conditions des évolutions associées du flux de produits qui ne sont pas obtenues sont fausses. Les pré-contraintes des évolutions associées non obtenues ne sont pas à satisfaire. Par exemple, s'il n'y a pas de pièce en H dans le magasin rotatif, l'indexation du magasin rotatif dans le sens horaire ne nécessite pas que le vérin 1 soit en position rentrée.

Depuis un état appartenant à cet ensemble d'états initiaux, la définition 2 caractérise l'évolution globale du système contrôlé résultante des effets de l'opération d'action suite à l'occurrence de l'événement de début de l'opération.

Définition 2 *Depuis un état q appartenant à l'ensemble des états initiaux du comportement $COA_{i,k}$, $q \in Q_{It(COA_{i,k})}$, et après l'occurrence de l'événement de début de l'opération d'action $OA_{i,k}$, noté $d(OA_{i,k})$, le système contrôlé atteint un état q'_1.*

La valeur des variables d'état dans l'état q'_1 est celle dans l'état q modifié par les effets transitoires $EfT(ec_i)$ et $EfT(ea_{i,j})$ pour $j \in J_k$.

La fonction de transition partielle est alors définie par $q'_1 = \delta(d(OA_{i,k}), q)$.

$$q \xrightarrow{d(OA_{i,k})} q'_1$$

Avec un point de vue système de commande par la prise en compte des grandeurs de commande, l'événement de début de l'opération est l'envoi de la requête à la chaîne fonctionnelle. Cet événement est noté $Rq(Se_i)$. La fonction de transition s'écrit alors $q'_1 = \delta(Rq(Se_i), q)$.

$$q \xrightarrow{Rq(Se_i)} q'_1$$

L'état q'_1 n'est pas nécessairement l'état depuis lequel l'événement de fin de l'opération aura lieu. En effet comme précisé au chapitre 4, le modèle doit représenter les parallélismes autorisés entre les opérations. Cela présuppose alors que durant une opération d'action, l'état du système contrôlé peut évoluer du fait d'autres opérations (d'action ou d'information) ou d'évolutions (induites ou requises). Par conséquent, les évolutions du système contrôlé et de son environnement depuis l'état q'_1 ne doivent pas le conduire dans un état tel que l'exécution en cours de l'opération d'action OA_i viole les contraintes de sécurité et d'écologie. Les états intermédiaires représentent ces états vers lesquels d'autres opérations (d'action ou d'information) ou évolutions (induites ou requises) peuvent faire évoluer le système contrôlé.

Définition 3 *L'ensemble des états intermédiaires du comportement $COA_{i,k}$, noté $Q_{Id(COA_{i,k})}$, est l'ensemble des états dans lesquels le système contrôlé et son environnement doivent être suite à l'occurrence de l'événement de début de l'opération d'action OA_i avec le comportement $COA_{i,k}$ et avant l'occurrence de l'événement de fin de cette opération. Formellement, cet ensemble est représenté par :*

$$Q_{Id(COA_{i,k})} = \{q \in Q_S / EfT(ec_i) \wedge Ct(ec_i) \bigwedge_{j \in J_k} [EfT(ea_{i,j}) \wedge Ct(ea_{i,j})] = Vraie\}$$

L'exécution de l'opération ne violera pas de contraintes de sécurité et d'écologie tant que l'état du système contrôlé et de son environnement restent dans l'ensemble des états intermédiaires. Depuis un état de cet ensemble, l'occurrence de l'événement de fin de l'opération aura pour effet sur le système contrôlé celui défini par la définition 4.

Définition 4 *Depuis un état q'_2 appartenant à l'ensemble des états intermédiaires du comportement $COA_{i,k}$, $q'_2 \in Q_{Id(COA_{i,k})}$, et après l'occurrence de l'événement de fin de l'opération d'action $OA_{i,k}$, noté $f(OA_{i,k})$, le système contrôlé atteint un état q''.*

La valeur des variables d'états dans l'état q'' est celle dans l'état q'_2 modifiée par les effets finaux $EfF(ec_i)$ et $EfF(ea_{i,j})$ pour $j \in J_k$.

La fonction de transition partielle est définie par $q'' = \delta(f(OA_{i,k}), q'_2)$.

$$q'_2 \xrightarrow{f(OA_{i,k})} q''$$

En intégrant les grandeurs de commande, l'événement de fin de l'opération d'action est la réception d'un compte rendu de fin. Cet événement est noté $Cr(Se_i)$. La fonction de transition s'écrit alors $q'' = \delta(Cr(Se_i), q'_2)$.

$$q'_2 \xrightarrow{Cr(Se_i)} q''$$

La Figure 5.6 représente les ensembles d'états initiaux et intermédiaires avec les évolutions du système contrôlé résultantes de l'occurrence des événements de début et de fin de l'opération d'action. Les évolutions entre q'_1 et q'_2, en pointillés sur la Figure, résultent d'opérations (d'action ou d'information) ou d'évolutions (induites ou requises) autres que l'opération d'action OA_i.

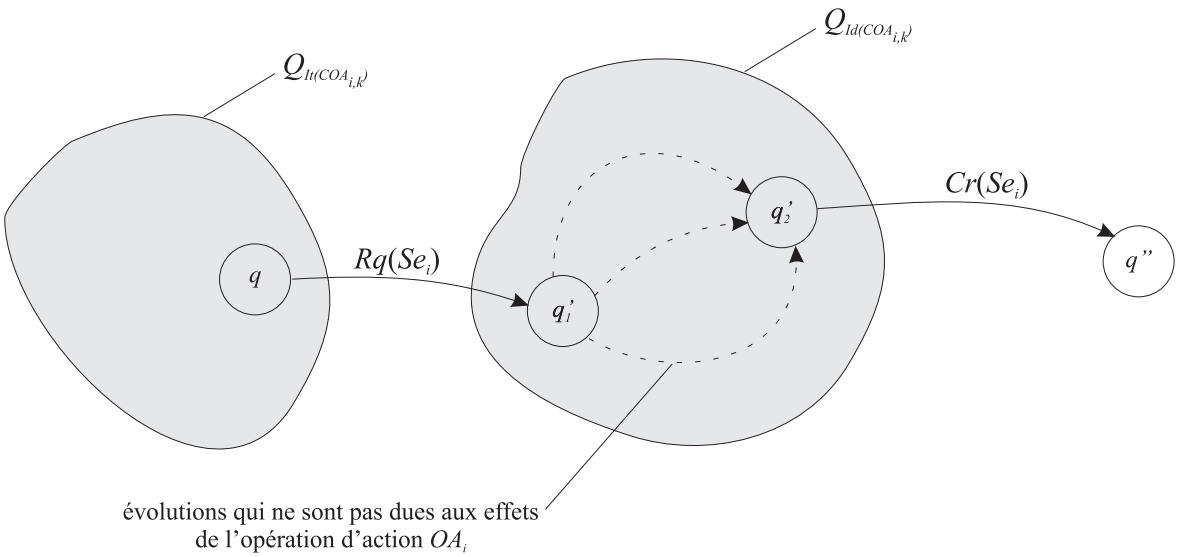


FIG. 5.6 – Comportement d'une opération d'action avec un point de vue grandeurs de commande.

Suite à la présentation des opérations d'action, nous allons présenter les opérations d'information.

6 Les opérations d'information

Une opération d'information modélise l'effet de la réalisation d'un service offert par une chaîne d'acquisition. Cet effet concerne d'une part la chaîne d'acquisition qui devient indisponible durant l'opération, et d'autre part la connaissance de l'état du flux de produit au niveau coordination. En effet, les données acquises lors de l'exécution du service, offert par une chaîne d'acquisition, accompagnent le compte rendu de fin. Ces données renseignent alors le niveau coordination de l'état du flux de produits sur lequel une incertitude existait ; celle-ci étant la conséquence des informations partielles fournies par l'environnement au module de coordination sur l'état des produits entrant dans le système contrôlé.

Comme une opération d'action, une opération d'information modélise le lien de causalité entre la réalisation d'un service, une évolution d'une chaîne fonctionnelle, et enfin une évolution du flux de produits. Ainsi, les champs de données du modèle d'une opération d'information sont identiques à ceux d'une opération d'action, à savoir, l'identificateur de l'opération d'information, les grandeurs de commande, les caractéristiques et enfin la dynamique du système contrôlé. Pour cette raison, nous ne représenterons pas ici son modèle illustré par la Figure A.1 en annexe A.

Comme nous l'avons rappelé dans le paragraphe précédent, une opération d'information se caractérise essentiellement par les données qui peuvent accompagner le compte rendu de fin du service. Ainsi l'état final, suite à la réception du compte rendu de fin, sera fonction de ces données. Il existe par conséquent plusieurs états finaux dont le modèle doit tenir compte.

En se basant toujours sur la même structure de modèle que celui d'une opération d'action (cf. Figure 5.5), le champ de données *Dynamique* d'une opération d'information intègre l'existence de plusieurs effets finaux en jouant sur l'écriture de l'effet final de l'évolution associée du flux de produits, noté ea_i. Dans cette optique, depuis un même état initial satisfaisant la condition

et la pré-contrainte de l'évolution associée du flux de produits, le champ de données *effet final* spécifie les effets possibles.

Une opération d'information possédera généralement une seule évolution associée du flux de produits. En effet, une chaîne d'acquisition sur laquelle se basent les opérations d'information est destinée presque toujours à acquérir la valeur d'une variable d'état qui est la même quel que soit le produit, par exemple le poids ou la présence. Aussi, l'effet sur le flux de produits est limité à l'évolution de cette variable d'état.

Néanmoins, si la chaîne d'acquisition est plus complexe et est capable en fonction du type de produit et/ou de sa position d'acquérir des données de natures différentes, le modèle d'une opération d'information possédera alors comme une opération d'action plusieurs évolutions associées du flux de produits.

Afin de simplifier la présentation des comportements d'une opération d'information, nous limiterons à un le nombre d'évolutions associées du flux de produits. Dans ce cas, une opération d'information possède deux comportements envisageables. Le premier comportement est caractérisé uniquement par l'évolution de la chaîne fonctionnelle. Le second comportement est lui caractérisé par l'évolution de la chaîne d'acquisition et l'évolution associée du flux de produits. Plusieurs états finaux pouvant être atteints, la définition de ce comportement est particulière à une opération d'information. Pour cette raison, nous allons maintenant uniquement présenter ce deuxième comportement.

La fin de cette section vise ainsi à définir le comportement, noté $COI_{i,k}$, d'une opération d'information caractérisé par une évolution de la chaîne d'acquisition et une évolution associée du flux de produits. Seul le champ de données *effet final* d'une opération d'information diffère par rapport au modèle d'une opération d'action. Ainsi, n'étant pas défini en fonction du champ de données *effet final*, l'ensemble des états initiaux $Q_{It(COI_{i,k})}$, la fonction de transition partielle $\delta(d(OI_{i,k}), q)$ liée à l'événement de début de l'opération d'information, et l'ensemble des états intermédiaires $Q_{Id(COI_{i,k})}$ ont les mêmes définitions que celles données pour une opération d'action (cf. définition 1, 2, et 3). Aussi, nous nous focaliserons dans cette section uniquement sur la fonction de transition partielle définissant l'évolution liée à la fin de l'opération d'information.

La fonction de transition partielle liée à la fin d'une opération d'information est définie depuis un état q'_2 qui appartient à l'ensemble des états intermédiaires $Q_{Id(COI_{i,k})}$. L'état atteint suite à la fin de l'opération étant fonction des données reçues avec le compte rendu de fin, la fonction de transition partielle caractérise l'ensemble des états finaux atteignables (cf. définition 5).

Définition 5 *Depuis un état q'_2 appartenant à l'ensemble des états intermédiaires du comportement $COI_{i,k}$, $q'_2 \in Q_{Id(COI_{i,k})}$, et après l'occurrence de l'événement de fin de l'opération d'information OI_i, noté $f(OI_{i,k})$, le système contrôlé atteint un des états finaux atteignables $\{q''_1, q''_2, q''_3, ...\}$.*

La valeur des variables d'état dans un état final q''_j est celle dans l'état q'_2 modifiées par le j^{ieme} effet de $EfF(ec_i)$ et $EfF(ea_i)$.

La fonction de transition partielle est définie par $\delta(f(OI_{i,k}), q'_2) = \{q''_1, q''_2, q''_3, ...\}$

La Figure 5.7 illustre cette définition avec pour événements, non plus le début et la fin de

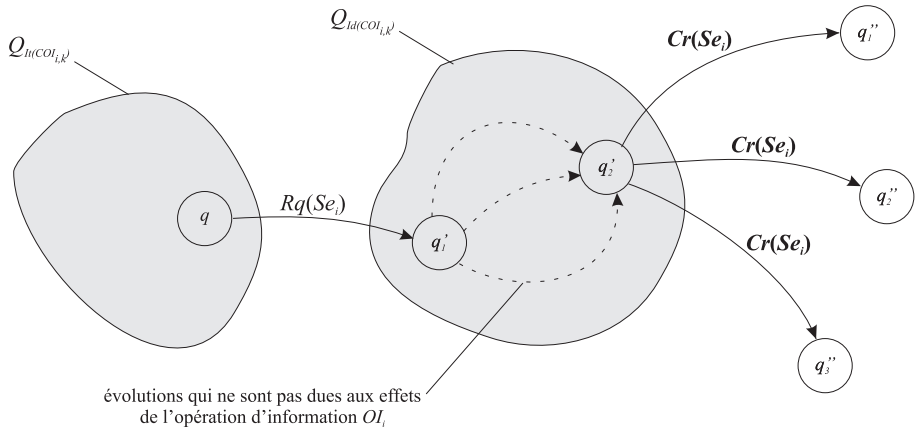


FIG. 5.7 – Comportement d'une opération d'information avec un point de vue grandeurs de commande.

l'opération d'information, mais les grandeurs de commande, à savoir, la requête d'appel du service Se_i et le compte rendu de fin du service Se_i.

Suite à l'étude du modèle des opérations d'action et d'information, nous proposons maintenant de nous focaliser sur les évolutions induites et requises.

7 Les évolutions induites

En préambule à l'étude des évolutions induites, nous allons d'abord repréciser ce qui les caractérise. La Figure 5.8 illustre l'enchaînement d'une opération d'action avec une évolution induite. Elle représente de plus les grandeurs de commande du niveau coordination. Au début et à la fin de l'opération d'action, il correspond respectivement une requête et un compte rendu de fin. L'état atteint à la fin de cette opération d'action est qualifié d'état instable puisque le flux de produits continue à évoluer. Cette évolution du flux de produits qui n'est pas simultanée à la réalisation d'un service est modélisée par une évolution induite. Cette instabilité de l'état du flux de produits peut être due soit à l'action sur le flux de produits d'une chaîne d'action dans un état particulier, comme par exemple le convoyeur central quand il est en marche, soit à une réaction chimique suite par exemple au mélange de deux produits. Une évolution induite ne peut pas représenter par hypothèse, du point de vue du niveau de coordination, une évolution d'une chaîne fonctionnelle dont seule la réalisation d'un service peut modifier l'état.

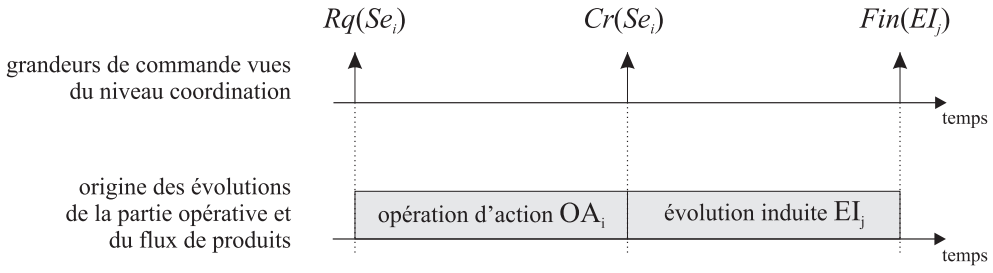


FIG. 5.8 – Perception d'une opération et d'une évolution induite au niveau coordination.

Suite à la présentation de ce qui caractérise une évolution induite par rapport à une opération, nous proposons maintenant de traduire ces différences en terme de modélisation.

Une évolution induite se différencie d'une opération d'une part par ses grandeurs de commande puisque son déclenchement ne correspond pas à l'envoi d'une requête, et d'autre part par l'absence d'une évolution d'une chaîne fonctionnelle perçue au niveau coordination. Ainsi, le modèle d'une évolution induite vis-à-vis de celui d'une opération ne comportera pas d'évolution de la chaîne fonctionnelle et les grandeurs de commande seront différentes :

La grandeur de commande est limitée à l'information relative à la fin de l'évolution induite, qui est une information capitale au niveau coordination afin par exemple de déclencher une opération. Par principe pour l'optimisation, une évolution induite aura toujours une durée $D(EI_i)$ comme pour les opérations d'action. L'événement de fin correspondra alors à la fin d'une temporisation d'une durée $D(EI_i)$ déclenchée lors du passage par un état instable $q \in Q_{It(CEI_{i,k})}$. La durée $D(EI_i)$ est un modèle temporel de l'évolution induite. Ainsi, toute information fournie par un capteur sur l'état du flux de produits atteint suite à une évolution induite est redondante avec le modèle temporel. De ce fait, des capteurs fournissant une telle information sont dédiés à la surveillance, et ils ne doivent donc pas apparaître dans notre modèle.

La dynamique se compose d'une à plusieurs évolutions simultanées du flux de produits. Des évolutions sont simultanées d'une part si elles résultent de l'action d'un même actionneur et d'autre part si la géométrie de l'actionneur impose cette simultanéité.

- Par exemple, pour la modélisation des actions basées sur le magasin rotatif du système d'approvisionnement de la Figure 5.1 page 72, il est possible de modéliser trois opérations (arrêter, démarrer moteur dans le sens horaire ou trigonométrique) et deux évolutions induites associées chacune à un sens de rotation. Une évolution induite aura huit évolutions du flux de produits (de A vers B, de B vers C,..., H vers A), et inversement pour les évolutions du flux de produits de la seconde évolution induite. Cette modélisation de huit évolutions du flux de produits pour une même évolution induite est rendue possible par la géométrie du magasin rotatif. En effet, elle impose que la position de tous les produits, présents dans le magasin, évolue simultanément lors de la rotation du magasin.
- En revanche pour le convoyeur central, les évolutions de J à K, de K à L et de L à M ne

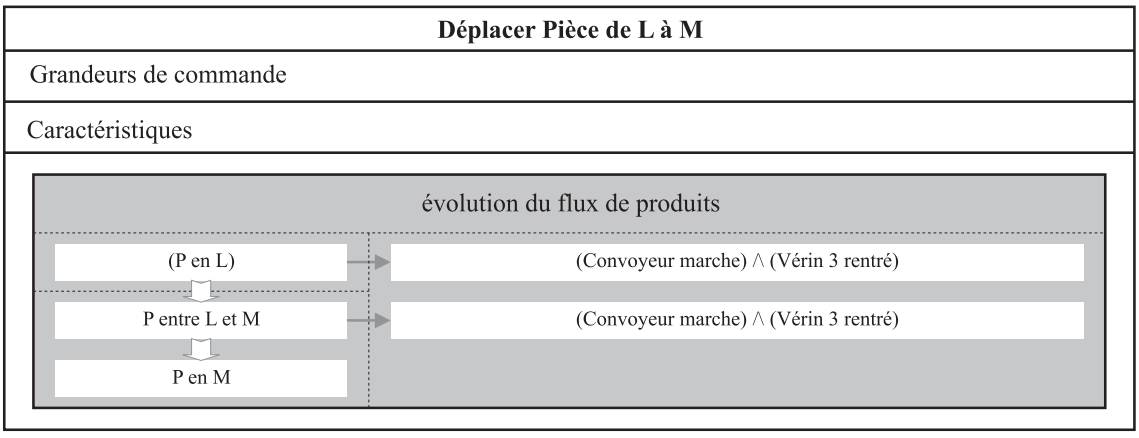


FIG. 5.9 – Évolution induite "Déplacer pièce de L à M" du point de vue du deuxième module de coordination du système d'approvisionnement de la plate-forme SAPHIR.

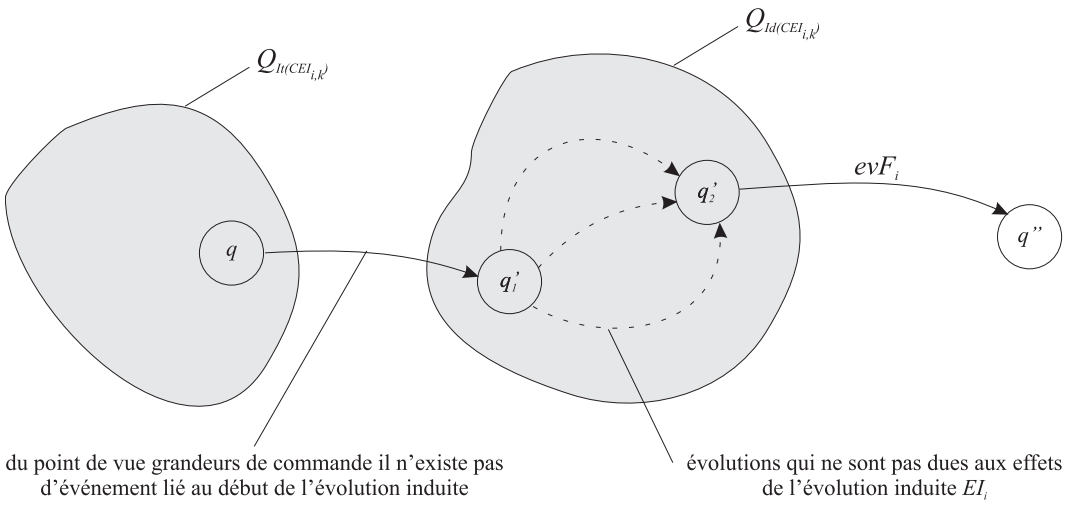


FIG. 5.10 – Comportement d'une évolution induite avec un point de vue grandeurs de commande.

peuvent pas être modélisées dans la même évolution induite. Toutes les trois résultent bien de l'action du convoyeur central, mais la géométrie du convoyeur n'impose pas la simultanéité des évolutions contrairement au magasin de pièces ; ce dernier offre en effet des emplacements distincts de pièces imposant un arrêt de rotation pour le chargement. La Figure 5.9 présente l'évolution induite modélisant l'évolution d'un produit de L à M.

Le modèle d'une évolution induite et la définition de son comportement qui sont semblables à ceux d'une opération d'action ne sont pas repris dans cette section. Cependant, le lecteur pourra se rapporter, dans l'annexe A, à la Figure A.2 qui représente le modèle d'une évolution induite, et au § 2 définissant son comportement. Toutefois, la Figure 5.10 illustre, avec un point de vue grandeurs de commande, les effets d'une évolution induite sur l'état du système contrôlé.

La dernière catégorie d'action, les évolutions requises, est présentée dans la section suivante.

8 Les évolutions requises

L'étude proposée dans cette section vise à définir le modèle d'une évolution requise. Les espaces partagés (cf. Figure 5.11) sont à l'origine de la modélisation des évolutions requises. Elle représente d'une part les arrivées et les départs des produits du système contrôlé, et d'autre part les interactions entre le système contrôlé et son environnement qui peuvent conduire à violer les contraintes de sécurité et d'écologie.

Suite à cette clarification, le modèle d'une évolution requise est proposé dans la section suivante.

8.1 Les données à modéliser

Les évolutions requises modélisent une évolution d'une chaîne fonctionnelle de l'environnement avec ou sans effet sur le flux de produits. Le niveau coordination considéré ne pilotant pas les chaînes fonctionnelles de son environnement, il n'y a pas de différence pour lui entre ce qui est, du point de vue de son environnement, une opération d'action ou une évolution induite. Ainsi,

pour couvrir ces situations et mis à part les grandeurs de commande, le modèle d'une évolution requise est identique à celui d'une opération d'action.

Les grandeurs de commande sont en effet différentes de celles d'une opération puisque le module de coordination ne pilote pas les chaînes fonctionnelles de son environnement. En revanche, le module de coordination considéré a besoin d'informations fournies par l'environnement afin de décider des opérations à exécuter, ne serait-ce que d'une part pour des raisons de sécurité afin d'éviter une collision, et d'autre part être informé de l'arrivée d'un produit à traiter.

Ainsi, le champ de données *"Grandeurs de commande"* peut être partiellement renseigné voir vide. Les trois cas de figures suivants sont alors envisageables :

- Aucune information n'est fournie par l'environnement, ni sur le début de l'évolution requise ni sur sa fin. Le champ *Grandeurs de commande* est vide. Pour le système d'approvisionnement de la plate-forme SAPHIR, c'est le cas des évolutions de l'opérateur humain qui dépose les pièces dans le magasin. L'opérateur humain n'a pas ici de moyens pour donner des informations sur le début ou la fin de ses évolutions.
- L'information fournie renseigne seulement sur le début ou la fin de l'évolution requise. C'est notamment le cas pour l'évolution du convoyeur central qui évacue une pièce qui est en J. En effet, seule l'information indiquant que la pièce est en K est fournie par l'environnement du premier module de coordination.
- Des informations indiquent le début et la fin d'une évolution requise. Il s'agit par exemple d'une évolution requise qui positionne un produit dans un espace partagé. Il peut être nécessaire pour des raisons de sécurité et d'écologie de connaître le début de l'évolution qui introduit un produit dans un espace partagé. Et il peut aussi être important de disposer d'informations sur la fin de l'évolution afin d'exécuter ensuite des opérations sur le produit positionné.

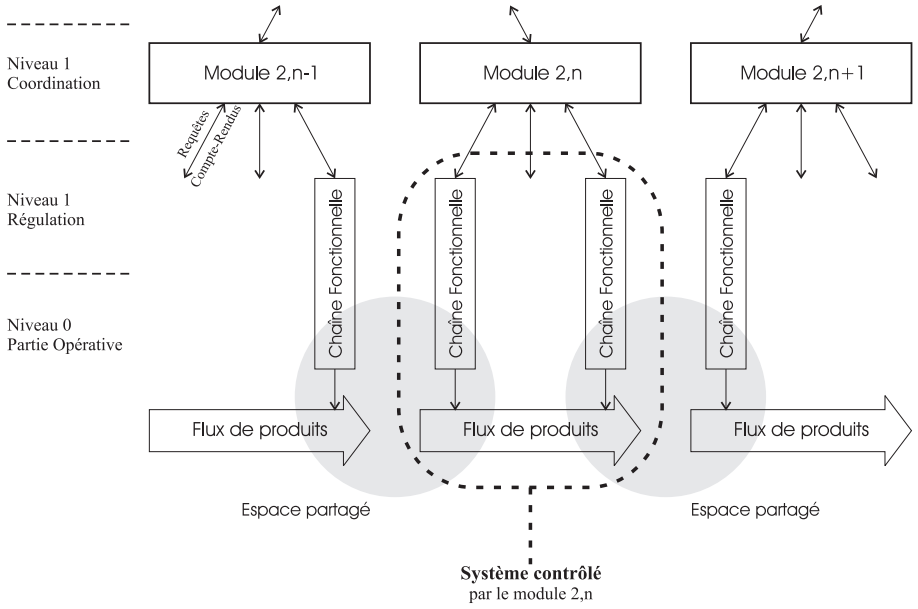


FIG. 5.11 – Redondance partielle et espaces partagés.

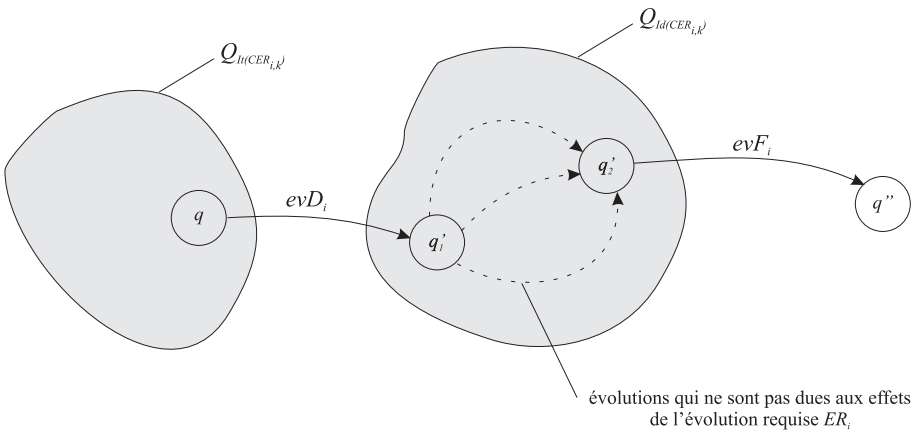


FIG. 5.12 – Comportement d'une évolution requise avec un point de vue grandeurs de commande.

La Figure 5.12 représente les ensembles d'états initiaux et intermédiaires ainsi que les évolutions du système contrôlé et de l'environnement dues à une évolution requise. Elle illustre le cas où le module coordination dispose des informations nécessaires sur les évolutions de son environnement lui permettant de connaître l'occurrence des événements de début et de fin de l'évolution requise. Les deux autres cas présentés précédemment au § 8.1 ne sont pas donnés ici mais ils sont une adaptation de la Figure 5.12 en supprimant les événements de début et/ou de fin suivant le cas considéré.

La prise en compte des nouvelles données du champ *"Grandeurs de commande"* conduit au modèle d'une évolution requise, présenté en annexe A dans la Figure A.3. En effet, plutôt que de présenter le modèle d'une évolution requise similaire à celui d'une opération d'action, nous avons choisi dans la section suivante de préciser les informations modélisées par une telle évolution.

8.2 Provenance des informations modélisées

Une évolution qui est requise du point de vue d'un module de coordination x est pour un module de coordination y une opération ou une évolution induite. Ce module de coordination y est celui qui pilote la chaîne fonctionnelle à l'origine de l'évolution requise. Comparativement, en terme d'informations, une évolution requise sera généralement moins riche que son équivalente sous la forme d'une opération ou d'une évolution induite. Nous remarquerons que ceci introduit forcément un problème de redondance partielle tel que traité dans (da Silveira, 2003). Afin de minimiser le travail de l'expert dans sa phase de modélisation générale de plusieurs modules de coordination, il sera conseillé de procéder en deux étapes successives : premièrement, et pour chaque module de coordination, réaliser la modélisation complète des opérations d'action et des évolutions induites, puis dans un second temps, modéliser les évolutions requises par reprise des modèles d'opération déjà réalisés pour les modules de coordination environnants.

Contrairement à une opération ou une évolution induite, tous les effets, aussi bien sur une chaîne fonctionnelle que sur le flux de produits, d'une évolution requise n'apparaissent pas forcément dans la dynamique de l'évolution. C'est la conséquence directe du rôle d'une évolution requise qui fournit les informations minimales qu'un module de coordination doit connaître d'une évolution de son environnement ; d'une part pour satisfaire les contraintes de sécurité et d'écologie, et d'autre part sur l'arrivée ou le départ de produits. Par conséquent, les variables

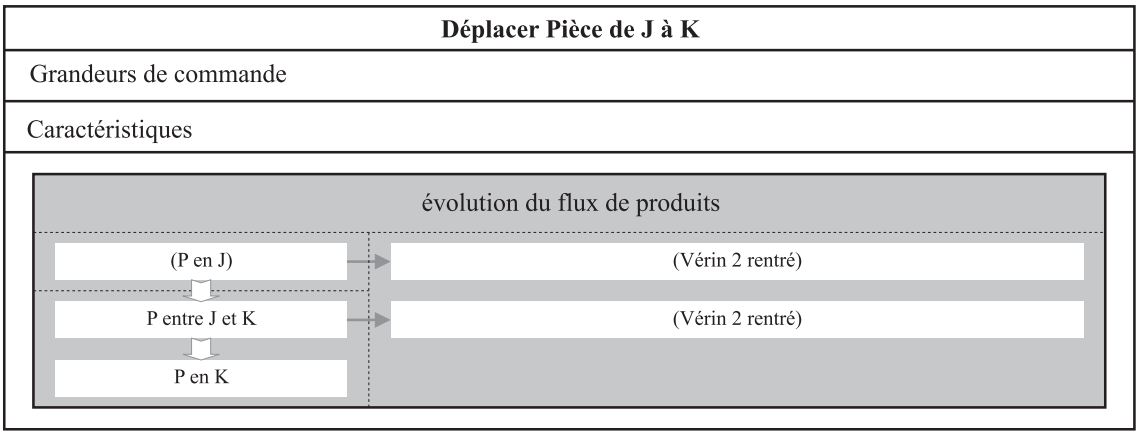


FIG. 5.13 – Évolution requise "Déplacer pièce de J à K" du point de vue du premier module de coordination du système d'approvisionnement de la plate forme SAPHIR.

d'états utilisées dans une évolution induite sont limitées à celles qui apparaissent dans les effets, conditions, pré-contraintes et contraintes des opérations (d'action ou d'information) et des évolutions induites. Par exemple, aucune variable d'état du convoyeur central n'apparaît dans le modèle de l'évolution requise de la Figure 5.13 vue du premier module de coordination. La raison en est l'absence d'interaction entre le vérin 2 et le convoyeur central.

Suite à l'étude des quatre catégories d'opérations qui forment le modèle du système contrôlé destiné à la synthèse de lois de commande, la section suivante donne une vue globale de ce modèle, et plus particulièrement des parallélismes et des concurrences modélisées.

9 Vision globale du modèle

Afin de terminer la présentation du modèle, nous proposons de revenir un instant sur la différence essentielle qui caractérise les quatre catégories d'opérations du point de vue du niveau de coordination. Ensuite, nous présenterons les parallélismes et les concurrences entre opérations que le modèle met en évidence.

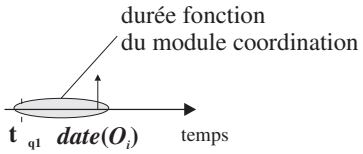
Du point de vue du niveau coordination, la caractéristique qui différencie les quatre catégories d'opérations est leur déclenchement. La Figure 5.14 résume pour chacune des catégories d'opérations le délai d'occurrence de l'événement de début de l'action depuis une date $t_{\rightarrow q1}$, $t_{\rightarrow q2}$ et $t_{\rightarrow q3}$ d'arrivée respectivement dans les états $q1$, $q2$ et $q3$ du système contrôlé et de son environnement depuis lequel les événements respectivement de début d'une opération $d(O_i)$ (d'action ou d'information), de début d'une évolution induite $d(EI_j)$ et de début d'une évolution requise $d(ER_k)$ sont autorisés.

Suite à ce rappel sur le déclenchement des différentes catégories d'opérations, la section suivante présente les parallélismes et les concurrences mis en évidence par le modèle du système contrôlé.

Opération d'action ou d'information

$\mathbf{t}_{q1}$ date d'arrivée dans un état q_1 autorisant O_i

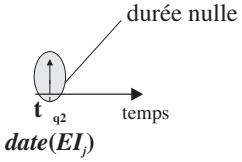
***date*(O_i)**: date de début de O_i



Evolution induite

$\mathbf{t}_{q2}$ date d'arrivée dans un état q_2 autorisant EI_j

***date*(EI_j)**: date de début de EI_j



Evolution requise

$\mathbf{t}_{q3}$ date d'arrivée dans un état q_3 autorisant ER_k

***date*(ER_k)**: date de début de ER_k

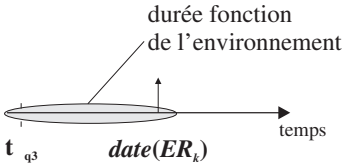


FIG. 5.14 – Comparatif du délai de déclenchement d'une opération (d'action ou d'information), d'une évolution induite et d'une évolution requise.

9.1 Parallélismes autorisés entre opérations

D'après les propriétés données au § 4 page 68, le modèle du système contrôlé et de son environnement se veut être représentatif des parallélismes autorisés entre les opérations. Cependant la structuration proposée des informations avec quatre catégories d'opérations ne fait pas apparaître explicitement les parallélismes. Ainsi, nous proposons dans cette section, à partir d'un ensemble de propriétés, de montrer comment les parallélismes sont inclus dans le modèle.

D'abord nous considérons qu'il n'y pas simultanéité des événements de début des opérations dont le parallélisme est recherché. Ensuite, nous considérerons la simultanéité des événements de début des opérations qui correspond au niveau coordination à l'émission simultanée de deux requêtes. Enfin, en dehors de toute considération sur la simultanéité des événements de début, une propriété générique sera proposée.

9.1.1 Parallélisme sans simultanéité

Les deux propriétés suivantes définissent les conditions nécessaires et suffisantes pour que deux opérations soit exécutées en parallèle sans simultanéité de leur événements de début ou de fin. La première propriété s'intéresse à vérifier qu'une opération peut être lancée en parallèle d'une autre qui est déjà en cours d'exécution. Puis, quand les deux opérations sont en exécution, la seconde propriété vérifie que la fin d'une des deux opérations ne conduit pas dans un état qui viole les contraintes de la seconde, toujours en exécution.

Commençons d'abord par la propriété 1 relative au lancement d'une opération en parallèle d'une autre déjà en exécution. N'étant pas fait ici de différence entre les quatre catégories d'opérations, le comportement d'une opération est simplement noté c.

Propriété 1 *Considérons le k^{ieme} comportement d'une opération O_i, noté $c_{i,k}$, et $c_{p,m}$, le m^{ieme} comportement d'une action O_p.*

Depuis un état q appartenant d'une part à l'ensemble des états intermédiaires du comportement $c_{i,k}$, $q \in Q_{Id(c_{i,k})}$, et d'autre part à l'ensemble des états initiaux du comportement $c_{p,m}$, $q \in Q_{It(c_{p,m})}$, l'occurrence de l'événement de début de l'opération O_p, $d(O_p)$, avec le comportement $c_{p,m}$ est autorisée si et seulement si :

l'état q'_1 atteint suite au lancement de l'opération O_p , $q'_1 = \delta(d(O_p), q)$, appartient toujours à l'ensemble des états intermédiaires du comportement $c_{i,k}$:

$$q'_1 \in Q_{Id(c_{i,k})}$$

Quand cette propriété est vérifiée, les deux opérations sont alors en exécution. Il est alors de surcroît nécessaire de s'assurer que la fin d'une des deux opérations ne violera pas les contraintes de l'autre, comme le propose la propriété 2.

Propriété 2 *Considérons toujours les deux comportements $c_{i,k}$ et $c_{p,m}$.*

Depuis un état q'_2 appartenant à l'ensemble des états intermédiaires d'une part du comportement $c_{i,k}$, $q'_2 \in Q_{Id(c_{i,k})}$, et d'autre part du comportement $c_{p,m}$, $q'_2 \in Q_{Id(c_{p,m})}$, l'occurrence de l'événement de fin de l'opération O_p, f_{O_p}, est autorisée si et seulement si :

l'état q'' atteint suite à la fin de l'opération O_p , $q'' = \delta(f_{O_p}, q'_2)$, appartient toujours à l'ensemble des états intermédiaires de $c_{i,k}$:

$$q'' \in Q_{Id(c_{i,k})}$$

Nous remarquerons que les deux propriétés 1 et 2 ne sont pas basées sur une représentation de l'espace d'états complet. Il suffit pour vérifier ces propriétés de disposer de l'état courant du système contrôlé et du modèle des deux opérations.

Si les propriétés précédentes sont intéressantes afin de déterminer les parallélismes autorisés entre deux opérations, elles ne couvrent pas le cas ou le niveau coordination lance simultanément deux opérations. La section suivante traite de ce problème.

9.1.2 Parallélisme avec simultanéité

D'une manière générale, le système de commande d'un module de coordination peut lancer simultanément deux opérations afin d'imposer un parallélisme d'exécution dans un objectif de minimisation du temps de cycle. Dans ce cas, les événements de début des deux opérations sont simultanés et la seule appartenance aux deux ensembles d'états initiaux des deux opérations ne suffit pas pour satisfaire les contraintes de sécurité et d'écologie.

C'est notamment le cas pour le système d'approvisionnement de la plate-forme SAPHIR, en particulier lors du lancement simultané de la sortie des vérins 1 et 2 qui sont orthogonaux. Considérons l'état q pour lequel les deux vérins sont rentrés et l'absence de produits. L'état q appartient d'une part à l'ensemble des états initiaux du comportement $COA_{SV1,1}$ de l'opération d'action *Sortir Vérin 1*, et d'autre part à l'ensemble des états initiaux du comportement $COA_{SV2,1}$ de l'opération d'action *Sortir Vérin 2*. Pourtant, le lancement simultané des deux opérations d'action viole les contraintes.

Aussi, il s'avère nécessaire de vérifier si le parallélisme entre ces deux opérations est autorisé quand elles sont lancées simultanément. Mais avant de proposer une propriété relative à leur lancement simultané, nous définissons d'abord la notion de lancement simultané.

Définition 6 *Considérons un comportement $c_{i,k}$ d'une opération O_i, et un comportement $c_{p,m}$ d'une opération O_p.*

Depuis un état q appartenant d'une part à l'ensemble des états initiaux du comportement $c_{i,k}$, $q \in Q_{It(c_{i,k})}$, et d'autre part à l'ensemble des états initiaux du comportement $c_{p,m}$, $q \in Q_{It(c_{p,m})}$, le système contrôlé atteint un état q'_1 suite à l'occurrence simultanée de l'événement de début de l'opération O_i, $d(O_i)$, et de l'événement de début de l'opération O_p, $d(O_p)$.

La valeur des variables d'état de l'état q'_1 est celle dans l'état q modifiée par les effets cumulés des effets transitoires de l'opération O_i, $EfT(ec_i)$, $EfT(ea_{i,j})$ $\forall j \in J_k$, et des effets transitoires de l'opération O_p, $EfT(ec_p)$, et $EfT(ea_{p,m})$ $\forall j \in J_m$.

$$q \xrightarrow{d(O_i).d(O_p)} q'_1$$

Le lancement simultané de deux opérations conduisant dans un état q'_1 tel que spécifié par la définition précédente est autorisé si et seulement si la propriété 3, présentée ci-dessous, est vérifiée.

Propriété 3 *Depuis un état q appartenant d'une part à l'ensemble des états initiaux du comportement $c_{i,k}$, $q \in Q_{It(c_{i,k})}$, et d'autre part à l'ensemble des états initiaux du comportement $c_{p,m}$, $q \in Q_{It(c_{p,m})}$, les opérations O_i et O_p peuvent être lancées simultanément si et seulement si :*

l'état q'_1 atteint suite au lancement simultané des deux opérations, $q'_1 = \delta(d(O_i).d(O_p), q)$, appartient d'une part à l'ensemble des états intermédiaires du comportement $c_{i,k}$, et d'autre part à l'ensemble des états intermédiaires du comportement $c_{p,m}$:

$$q'_1 \in Q_{Id(c_{i,k})} \cap Q_{Id(c_{p,m})}$$

Pour vérifier la propriété précédente sur le parallélisme, l'espace d'états du système contrôlé et de son environnement n'est pas requis. Néanmoins, il est nécessaire d'une part de disposer de l'état courant q, et d'autre part de calculer l'état q'_1 atteint suite au lancement simultané des deux opérations.

De plus, cette propriété est spécifique au lancement simultané de deux opérations. Quand le lancement n'est pas simultané, les propriétés à vérifier sont celles présentées dans la section précédente (les propriétés 1 et 2).

Ainsi pour des raisons de simplicité, la section suivante propose une propriété générique pour déterminer si le parallélisme entre deux opérations est autorisé.

9.1.3 Propriété générique afin de vérifier le parallélisme

Afin de disposer d'une propriété simple et générique, seule la vérification de la propriété 4, ci-dessous, est nécessaire afin de s'assurer du parallélisme entre deux opérations.

Propriété 4 *Considérons une séquence de deux opérations : O_i avec le comportement $c_{i,k}$, puis O_p avec le comportement $c_{p,m}$; et de plus un état q du système contrôlé et de son environnement depuis lequel la séquence peut être exécutée.*

Les deux opérations pourront être exécutées en parallèle avec ou sans simultanéité de leur lancement si :

- *le comportement $c_{i,k}$ n'a pas d'effet sur les variables d'états (ve) utilisées dans le comportement $c_{p,m}$ pour spécifier $Cd(ec_p)$, $Cd(ea_{p,j})$, $PeC(ec_p)$, $PeC(ea_{p,j})$, $Ct(ec_p)$, et $Ct(ea_{p,j})$ $\forall j \in J_m$; et*
- *le comportement $c_{p,m}$ n'a pas d'effet sur les variables d'états (ve) utilisées dans le comportement $c_{i,k}$ pour spécifier $Cd(ec_i)$, $Cd(ea_{i,j})$, $PeC(ec_i)$, $PeC(ea_{i,j})$, $Ct(ec_i)$, et $Ct(ea_{i,j})$ $\forall j \in J_k$.*

Avec cette propriété 4 et pour une séquence d'actions, seul le modèle des opérations est nécessaire afin de déterminer si le parallélisme d'exécution est autorisé.

Suite à cette étude sur le parallélisme d'exécution, la section suivante se focalise sur la concurrence entre opérations.

9.2 Analyse des concurrences entre opérations

Dans le cadre de cette section, nous allons nous intéresser non seulement à la compréhension des situations de concurrence entre opérations (tout parallélisme d'exécution interdit depuis un état donné) mais également à leur pertinence. En effet, sans compréhension fine de ces situations, il n'est envisageable, à terme, ni vérification du modèle obtenu, ni son utilisation, ne serait-ce qu'à des fins de synthèse de lois de commande.

Pour un état considéré, et deux types d'opérations concurrentes prises parmi les quatres mises en exergue (opération d'action, opération d'information, évolution induite et évolution requise), six situations doivent être étudiées, en confondant les opérations d'action et d'information basées sur l'appel à un service :

Deux opérations concurrentes. Le module de coordination aura à choisir l'opération à exécuter en envoyant la requête correspondante en fonction du ou des critères fixés.

Deux évolutions induites concurrentes. Les deux évolutions induites résultent de l'instabilité de l'état q. Le délai d'occurrence de leur événement de début étant nul, elles auront lieu en même temps. Aussi, cette situation ne doit pas exister au sein du modèle car le démarrage simultané de deux opérations concurrentes non maîtrisées viole les contraintes de sécurité et d'écologie. Afin de ne pas être confronté à une telle situation, le modèle du système contrôlé devra vérifier cette propriété.

Deux évolutions requises concurrentes. Cette situation est envisageable car les modules de coordination contrôlant les chaînes fonctionnelles à l'origine de ces deux évolutions doivent coordonner leurs actions quand il existe des interactions.

Une opération d'action et une évolution induite concurrentes. Par hypothèse, nous considérons que le lancement au plus tôt de l'opération d'action est possible. Si l'opération d'action est lancée au plus tôt, l'évolution induite n'a pas lieu. Par exemple, une évolution induite de déplacement d'une pièce par un convoyeur à bande, et une opération de sortie d'un vérin éjectant le produit du convoyeur sont concurrentes quand le produit se trouve devant le vérin. L'opération de sortie du vérin est autorisée, et son exécution supprime l'évolution induite.

Une opération et une évolution requise concurrentes. Par principe, le niveau coordination a une connaissance minimale de son environnement afin de satisfaire les contraintes de sécurité et d'écologie. De même, ce principe s'applique à l'environnement vis-à-vis du niveau de coordination. De plus, les occurrences des événements de début des deux opérations qui ne sont pas déclenchées par le même niveau de coordination ne peuvent pas être simultanées. Ainsi, nous considérons que le lancement au plus tôt de l'opération est possible. Afin de choisir l'exécution de l'évolution requise, le niveau coordination devra attendre la réalisation de cette évolution.

Une évolution induite et une évolution requise concurrentes. Comme pour la situation correspondante au couple {opération, évolution induite}, seul le lancement au plus tôt de l'évolution requise permet son exécution. Le lancement au plus tôt d'une évolution requise ne peut pas être garanti car elle est déclenchée par l'environnement. Ainsi dans cette situation de concurrence, le module de coordination considérera uniquement l'évolution induite.

10 Conclusion

Dans ce chapitre, nous nous sommes employés à présenter le formalisme que nous proposons. Organisés autour du découpage en quatre catégories d'opérations qui caractérise la vision que doit avoir un niveau de coordination du système contrôlé, nous avons, pour chacune de ces opérations détaillé et argumenté la structure du modèle. Pour une opération d'action, le modèle se compose de quatre champs : le premier caractérise l'identificateur de l'opération, le second spécifie les grandeurs de commande contrôlables et observables par un module de coordination, le troisième permet d'énumérer les caractéristiques quantitatives de l'opération (durée, coût, ...), le quatrième permet de modéliser la dynamique comportementale de l'opération au travers de l'évolution unique de la chaîne fonctionnelle et des évolutions associées du flux de produits. Essentiellement basées sur le modèle d'une opération d'action, les opérations d'information, les évolutions induites et requises sont ensuite présentées. Les caractéristiques découlant de l'interaction de chacun de ces modèles d'opérations ont fait également l'objet d'une étude, afin de discuter des parallélismes et des concurrences résultants du modèle global. De la compréhension de ces parallélismes et concurrences, découle en partie la structuration algorithmique du mécanisme de conception de loi de commande.

Le formalisme de modélisation ayant été désormais présenté, le chapitre suivant se propose de donner un ensemble de propriétés destinées à la validation du modèle global obtenu.

Chapitre 6

Validation du Modèle et Guide de Modélisation

1 Introduction

Les fondements mêmes de toute modélisation reposent sur la capacité de l'être humain à passer d'une vision informelle de la réalité à sa représentation formelle. S'il semble très difficile de vérifier l'adéquation du modèle obtenu par rapport au monde réel, il n'en demeure pas moins que certaines voies peuvent être explorées pour contribuer à minimiser la distance qui sépare le formel de l'informel.

Dans cet objectif, ce chapitre se propose justement d'explorer deux de ces voies. La première vise à définir un ensemble de propriétés génériques que doit vérifier le modèle obtenu. La seconde, davantage amont, préconise la mise en place d'une méthodologie de modélisation permettant de guider l'expert dans sa phase d'acquisition de la connaissance du monde réel.

2 Propriétés pour la validation du modèle

La validation du modèle par la vérification d'un ensemble de propriétés qui reposent sur les notions telles que les évolutions interdites/autorisées et les états interdits/dangereux/autorisés requiert de définir au préalable ces notions. Ainsi, nous commencerons par définir la notion d'évolutions interdites et autorisées pour ensuite présenter les états interdits, dangereux et autorisés. Enfin, les propriétés du modèle destinées à sa validation seront exposées.

2.1 Les évolutions autorisées et interdites

L'ensemble des contraintes de sécurité et d'écologie à satisfaire conduit à classer les évolutions du système contrôlé en deux catégories :

- **les évolutions interdites**, qui conduisent à un état dans lequel les contraintes de sécurité et d'écologie sont violées. C'est par exemple le cas pour un état dans lequel deux vérins en L sont sortis, ou bien encore celui du dépassement de la capacité d'un stock conduisant à la détérioration des produits.
- **Les évolutions autorisées** sont les évolutions non interdites. L'ensemble de ces évolutions définit tout ce qu'il est possible de faire du point de vue du niveau coordination. Le recours

uniquement à ces évolutions, afin de répondre à une demande, assure de satisfaire les contraintes de sécurité et d'écologie.

Au delà d'une probabilité plus grande de converger vers une solution satisfaisant la demande, l'accès à toutes les évolutions autorisées est un gage d'obtention de meilleures performances. Sur ce dernier point, il est crucial de bien tenir compte de la problématique générale de conception de lois de commande dans tous les contextes (configuration et reconfiguration) et ne pas se limiter à une vision de fonctionnement normal.

Par exemple pour des états qui sont accessibles uniquement suite à une défaillance, les évolutions autorisées depuis ces états sont très importantes dans un contexte de reprise. Prenons le cas de deux vérins en L qui sont en position intermédiaire. Cet état n'est pas accessible par une évolution autorisée car il y a un risque de collision entre les deux vérins. Mais, suite à une défaillance par exemple d'un capteur de fin de course du vérin, il est possible d'atteindre un tel état. Depuis cet état et sans collision des vérins les ayant détériorés, il existe une évolution permettant de rentrer un des deux vérins. De cette manière, il est encore envisageable de répondre à la demande en cours sans nécessiter une intervention immédiate du service de maintenance.

La section suivante définit les notions d'états interdits, dangereux et autorisés.

2.2 Les états interdits, dangereux et autorisés

Les notions d'états interdits et autorisés sont proches de celles utilisées dans le cadre de la théorie de Ramadge et Wonham et plus particulièrement pour le problème d'états interdits (Achour et Rezg, 2004) ; à ceci près que ces états se rapportent à ceux des chaînes fonctionnelles. En revanche, la notion d'états dangereux est, dans notre cas, différente de celle des états faiblement interdits.

Nous introduisons d'abord les notions d'états interdits, dangereux et autorisés pour ensuite montrer qu'ils ont un sens uniquement vis à vis d'une chaîne fonctionnelle.

États interdits

Définition 7 *Il s'agit des états dans lesquels les contraintes de sécurité et d'écologie sont violées.*

De tels états ne sont pas réellement et physiquement atteignables sans détériorer le système contrôlé et/ou son environnement conduisant alors à changer la nature du système et de ses capacités. C'est par exemple le cas où les deux vérins (1 et 2) sont en position sortie. Cet état n'est pas atteignable par le système réel sans modifier ses caractéristiques physiques : tige des vérins tordues avec ou sans détérioration des joints d'étanchéité, etc. Le modèle du système contrôlé et de son environnement à des fins de conception de lois de commande n'a pas vocation de représenter les défaillances, leurs causes et leurs conséquences dans le contexte d'une approche CERBERE. Les détériorations subies par le système contrôlé et son environnement n'étant pas modélisées et par conséquent les services encore offerts par les chaînes fonctionnelles étant inconnus dans un état interdit, les évolutions autorisées depuis un état interdit ne peuvent pas être modélisées.

États dangereux

Définition 8 *Ce sont des états dans lesquels il y a un risque de violer les contraintes de sécurité et d'écologie.*

C'est le cas par exemple où les deux vérins 1 et 2 qui sont orthogonaux sont en position intermédiaire. La précision du modèle ne permet pas dans cet état de savoir si les deux vérins sont entrés en collision ou non. Aussi, il est impossible de savoir si les contraintes de sécurité et d'écologie sont satisfaites. Il faut alors interdire d'atteindre un état dangereux pour lequel la probabilité de violer les contraintes de sécurité et d'écologie est non nulle. Cependant, suite à l'occurrence d'une défaillance, un état dangereux peut être atteint sans détériorer le système contrôlé et son environnement (informations données par les fonctions *diagnostiquer* et *suivre* cf. Chapitre 1). Depuis un tel état, il peut exister des évolutions respectant les contraintes de sécurité et d'écologie.

États autorisés

Définition 9 *Ils représentent les états dans lesquels les contraintes de sécurité et d'écologie sont satisfaites.*

Hormis l'état initial du système contrôlé et de son environnement, depuis lequel une loi de commande est applicable, tous les autres états par lesquels passera le système contrôlé et son environnement sont des états autorisés en l'absence de défaillance. En effet, il n'existe pas d'évolution dans le modèle conduisant à un état interdit ou dangereux. Dans ce cas, seules des évolutions vers des états autorisés sont possibles.

Les notions d'états interdits, dangereux, et autorisés sont relatives à l'état d'un sous-système qui se compose d'une chaîne fonctionnelle et des éléments avec lesquels elle est en interaction. Nous démontrons cette proposition uniquement pour les états interdits. Mais un raisonnement identique démontre également que les notions d'états dangereux et autorisés ont un sens uniquement vis-à-vis d'une chaîne fonctionnelle et de son environnement.

En effet, les chaînes fonctionnelles et les produits dont l'état n'apparaît pas dans les conditions et les pré-contraintes d'une opération peuvent violer les contraintes de sécurité, comme par exemple deux chaînes fonctionnelles en collision. Dans ce cas, si la notion d'état interdit s'applique à un état du système contrôlé, il y a contradiction entre la définition d'un état interdit et les définitions des états initiaux. En effet, pour un même état, la définition des états initiaux autorise une évolution alors que la définition des états interdits proscrit toute évolution.

En revanche, en appliquant la notion d'état interdit uniquement à un état d'un sous-système composé d'une chaîne fonctionnelle et des éléments avec lesquels elle est en interaction, les définitions des états initiaux d'un comportement (d'une opération d'action ou d'information, d'une évolution requise ou induite) et des états interdits sont conformes.

Considérons le module qui coordonne les quatre chaînes fonctionnelles suivantes : le magasin rotatif, les deux vérins (1 et 2) et le poste de pesée. De surcroît, supposons que deux des chaînes fonctionnelles sont dans un état violant les contraintes de sécurité et d'écologie (collision d'une pièce dans le magasin rotatif avec le vérin 1). Alors ce n'est pas pour autant que nous pouvons parler d'états interdits du système contrôlé par le module de coordination. En effet, des opérations des deux autres chaînes fonctionnelles peuvent tout à fait être exécutées depuis un tel état. L'opération d'identification reste en effet disponible car aucune condition, ni pré-contrainte, ni contrainte ne contient des variables d'états du vérin 1, du magasin rotatif et du flux de produits présent dans le magasin. Cet état est un état interdit du sous-système constitué de la chaîne fonctionnelle associée au magasin rotatif et des éléments avec lesquels elle

est en interaction : le vérin 1 et le flux de produits pour toutes les positions dans le magasin et la position entre le magasin et le poste d'identification.

Les états interdits se définissent pour un sous-système qui se compose d'une chaîne fonctionnelle, notée ChF_i, et des éléments avec lesquels elle est en interaction. Les états Q_S du système contrôlé et de son environnement dans lesquels le sous-système ci-dessus est dans un état interdit sont notés $Q_I(ChF_i)$.

Afin d'illustrer cette notion d'états interdits définie pour une chaîne fonctionnelle et de son environnement, considérons l'exemple d'un système constitué de quatre vérins présenté dans la Figure 6.1. L'état de chacun des vérins est défini par sa position qui peut prendre la valeur sortie (S) et rentrée (R). L'ensemble des états Q_S du système composé des quatre vérins sans la prise en compte du flux de produits comporte 16 états représentés (cf. Figure 6.1). Ainsi, pour cet exemple, quatre états interdits liés au vérin 1, six états interdits liés au vérin 2, six états interdits liés au vérin 3, et enfin quatre états interdits liés au vérin 4 sont dénombrables.

Les ensembles d'états interdits, dangereux et autorisés, notés respectivement Q_I, Q_D et Q_A, sont utilisées maintenant afin d'exprimer les propriétés pour la validation du modèle du système contrôlé.

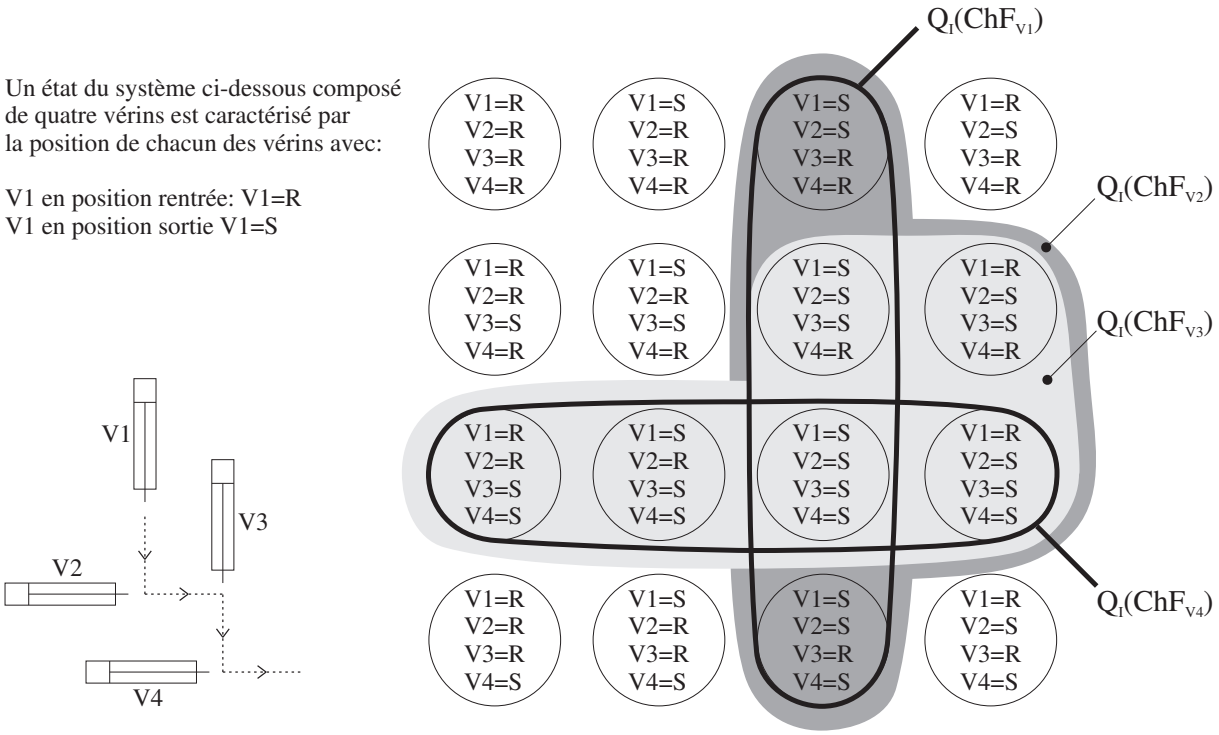


FIG. 6.1 – Intersection des ensembles d'états interdits de différentes chaînes fonctionnelles.

2.3 Propriétés servant à valider le modèle

Basée sur les définitions du comportement d'une opération d'action (opération d'information, évolution induite et requise) (Henry et al., 2005) et des états interdits (dangereux et autorisés), cette section propose un ensemble de propriétés dont l'objectif est de lever toute ambiguïté de compréhension du modèle proposé mais également d'offrir un moyen de valider le modèle.

De par la structure générique du modèle des quatre catégories d'opérations, les propriétés de cette structure sont communes aux opérations et aux évolutions. Leur présentation est faite sous trois aspects : cohérence des champs de données, déterminisme des opérations, et cohérence vis-à-vis des états interdits et dangereux. Cette section se termine par une propriété spécifique aux évolutions requises.

2.3.1 Cohérence des champs de données

La propriété ci-dessous offre la possibilité de vérifier la cohérence d'une part entre les champs *condition* et *pré-contrainte*, et d'autre part entre les champs *effet transitoire* et *contrainte*. Elles sont basées sur la proposition logique définissant l'ensemble des états initiaux et celle de l'ensemble des états intermédiaires.

Propriété 5 $\forall ea_{i,j} \in EA_i$, *alors* $\exists c_{i,k}$ *tel que* $j \in J_k$, $Q_{It(c_{i,k})} \neq \{\emptyset\}$ *et* $Q_{Id(c_{i,k})} \neq \{\emptyset\}$.

Cette propriété exprime le fait qu'il doit exister pour une évolution associée du flux de produits au moins un comportement de l'opération, pour lequel elle est obtenue, dont l'ensemble des états initiaux et l'ensemble des états intermédiaires sont non vides.

Si cette propriété n'est pas satisfaite, alors l'effet de cette évolution associée n'est jamais réalisable. A moins que cette évolution soit inutile et donc à supprimer, il existe une incohérence parmi les éléments suivants : les conditions ($Cd(ec_i)$, $Cd(ea_{i,j})$), les pré-contraintes ($PeC(ec_i)$, $PeC(ea_{i,j})$), les effets transitoires ($EfT(ec_i)$, $EfT(ea_{i,j})$) et les contraintes ($Cd(ec_i)$, $Cd(ea_{i,j})$).

La vérification de cette propriété ne nécessite pas de disposer des états interdits, dangereux et autorisés. Elle requiert seulement les propositions logiques des différentes données de la dynamique d'une opération d'action. Après cette propriété sur la cohérence des données, la section suivante propose des propriétés relatives au déterminisme d'une opération d'action.

2.3.2 Déterminisme des opérations d'action

Comme annoncé précédemment, cette section s'intéresse à deux propriétés d'une opération d'action visant à s'assurer de son déterminisme. Pour une même opération d'action, la première se focalise sur les conditions de deux évolutions associées du flux de produits. Ensuite toujours pour une même opération d'action, la seconde propriété s'intéresse à l'effet sur une même variable d'état de deux évolutions associées du flux de produits.

Propriété 6 $\forall m \in [0, N]$ *et* $\forall n \in [0, N]$ *avec* $m \neq n$, *alors* $Cd(ea_{i,n}) \neq Cd(ea_{i,m})$.

En d'autres termes, les conditions de deux évolutions associées d'une opération doivent être différentes. Pour prouver cette propriété, considérons deux évolutions associées $ea_{i,1}$ et $ea_{i,2}$, parmi toutes, qui ont la même condition : $Cd(ea_{i,1})$ et $Cd(ea_{i,2})$. L'ensemble des états initiaux

d'un comportement de l'opération avec $ea_{i,1}$ mais sans $ea_{i,2}$ est défini par, voir définition 1 page 80 :

$$Q_{It(c_{i,k})} = \{q \in Q_S / Cd(ec_i) \wedge PeC(ec_i) \wedge Cd(ea_{i,1}) \wedge PeC(ea_{i,1}) \wedge \neg Cd(ea_{i,2}) \bigwedge_{j \in J_k - \{1\}} [Cd(ea_{i,j}) \wedge PeC(ea_{i,j})] \bigwedge_{j \in \overline{J_k} - \{2\}} \neg Cd(ea_{i,j}) = \text{VRAIE}\}$$

La proposition $Cd(ea_{i,1}) \wedge \neg Cd(ea_{i,2})$ est toujours fausse $\forall q \in Q_S$ car $Cd(ea_{i,1})$ et $Cd(ea_{i,2})$ sont identiques. Aussi, l'ensemble $Q_{It(CO_{i,k})}$ est vide et prouve le déterminisme d'une opération pour laquelle soit les deux évolutions associées avec la même condition sont obtenues simultanément, ou alors ni l'une ni l'autre est obtenue. Dans ce cas, elles doivent être fusionnées afin de diminuer la taille du modèle.

Propriété 7 *Soit $ea_{i,m}$ et $ea_{i,n}$ avec un effet sur une même variable d'état, alors $Q_{It(c_{i,k})} = \{\emptyset\}$ si m et $n \in J_k$ et $m \neq n$.*

Si deux évolutions associées du flux de produits ont un effet sur une même variable d'état, il ne doit pas exister de comportement de l'opération, dont l'ensemble des états initiaux est non vide, pour lequel les deux évolutions associées sont obtenues simultanément. Dans le cas contraire, l'effet d'une opération est alors indéterministe car la variable d'état peut prendre la valeur affectée par l'une ou l'autre des évolutions associées. Cette propriété assure par conséquent le déterminisme souhaité d'une opération d'action.

Les dernières propriétés communes aux opérations et aux évolutions sont présentées dans la section suivante.

2.3.3 Cohérence vis-à-vis des états interdits et dangereux

Comme nous le montrerons au § 3.1, les états interdits et dangereux peuvent être spécifiés pour chacune des chaînes fonctionnelles en parallèle de la modélisation du système contrôlé et de son environnement. Les propositions spécifiant ces états et celles spécifiant les états initiaux, intermédiaires et finaux étant différentes, le modèle se doit alors d'être cohérent vis-à-vis de ces états interdits et dangereux, et vice versa. Ainsi, les propriétés ci-dessous visent à s'assurer de cette cohérence. Rappelons bien qu'il s'agit ici uniquement de la possibilité de vérifier la cohérence du modèle et en aucun cas d'affirmer que le modèle est "juste" ou "correct". En effet, ce dernier étant le fruit du travail d'un expert réalisant le passage d'un monde réel non formel à un modèle formel, seule une vérification de la cohérence des données est possible.

Propriété 8 *$\forall q'_1$ tel que $q'_1 = \delta(d(c_{i,k}), q)$, alors $q'_1 \in Q_{Id(c_{i,k})}$*

L'état q'_1 atteint depuis l'état q suite à l'occurrence de l'événement de début de l'opération O_i appartient à l'ensemble des états intermédiaires du comportement de l'opération obtenu pour l'état q. En effet, s'il n'y a pas d'autre opération ou évolution durant l'opération O_i, le système contrôlé et son environnement resteront dans l'état q'_1 jusqu'à l'occurrence de l'événement de fin de l'opération. D'après la définition 3, le système contrôlé et son environnement doivent rester

pendant toute l'opération dans l'ensemble des états intermédiaires de son comportement. Par conséquent, q'_1 doit appartenir à cet ensemble.

Si cette propriété n'est pas satisfaite pour un comportement $c_{i,k}$, un ou plusieurs des éléments suivants sont mal spécifiés parmi ceux-ci : les conditions ($Cd(ec_i)$, $Cd(ea_{i,j})$ $\forall j \in J_k$), les effets transitoires ($EfT(ec_i)$, $EfT(ea_{i,j})$ $\forall j \in J_k$), les pré-contraintes ($PeC(ec_i)$, $PeC(ea_{i,j})$ $\forall j \in J_k$), et les contraintes ($Ct(ec_i)$, $Ct(ea_{i,j})$ $\forall j \in J_k$).

Si l'ensemble des champs de données de la dynamique d'une opération d'action est correctement spécifié, alors un état initial est un état autorisé ou dangereux, un état intermédiaire est un état autorisé, ou dangereux s'il est identique à l'état initial, et un état final ne peut être qu'un état autorisé. Ceci est exprimé par les trois propriétés suivantes.

Propriété 9 *$\forall q \in Q_{It(c_{i,k})}$ avec $c_{i,k}$ un comportement d'une opération O_i basée sur une chaîne fonctionnelle (ChF_j), alors $q \in Q_A(ChF_j) \cup Q_D(ChF_j)$*

Propriété 10 *$\forall q'_1$ tel que $q'_1 = \delta(d_{c_{i,k}}, q)$ avec $q \neq q'_1$, alors $q'_1 \in Q_A(ChF_j)$*

Propriété 11 *$\forall q"$ tel que $q" = \delta(f_{c_{i,k}}, q'_1)$, alors $q" \in Q_A(ChF_j)$*

Toutes les propriétés communes aux opérations (d'action et d'information) et évolutions (induites et requises) sont maintenant définies. Il nous reste à présenter une propriété spécifique aux évolutions requises.

2.3.4 Propriété spécifique aux évolutions requises

D'après sa définition, une évolution requise fournit la connaissance minimale sur le comportement de l'environnement qui permet de vérifier les pré-contraintes et contraintes des opérations (d'action et d'information) et des évolutions induites. Ainsi, toute variable d'état utilisée dans une évolution requise doit apparaître dans au moins une opération d'action ou d'information, ou une évolution induite.

Propriété 12 *$\forall ve$, une variable d'état utilisée dans le champ dynamique d'une évolution requise composé de $EfT(ee_i)$, $EfF(ee_i)$, $EfT(es_{i,j})$ ou $EfF(es_{i,j})$, $\exists OA_i$ ou OE_i ou EI_i dans laquelle ve est utilisée.*

La vérification de cette propriété comme celle des propriétés précédentes ne demande pas de compétences particulières à l'expert comparativement à la vérification de lois de commande, comme le démontre la section suivante.

2.4 Difficulté de vérification des propriétés

Par comparaison avec la vérification et la validation (Frey et Litz, 2000) de lois de commande par model-checking (mac Millan, 1993) ou par theorem-proving (Newborn et Newborn, 2001), la vérification des propriétés du modèle du système contrôlé et de son environnement ne requiert pas l'écriture de propriétés complexes dans un langage particulier comme la logique temporelle (Emerson, 1990). En effet, le travail de l'expert sera limité à la modélisation des opérations et des évolutions à laquelle s'ajoute la spécification des états interdits et dangereux. A partir de ces données, les propriétés précédentes qui sont indépendantes du système modélisé

peuvent alors être automatiquement vérifiées sans autre intervention de l'expert. Ceci est à comparer avec la difficulté de vérification des propriétés d'une loi de commande conçue par un expert.

En se basant sur un modèle du système contrôlé satisfaisant les propriétés précédentes, considéré alors comme correct, il est impossible avec ce modèle d'atteindre un état interdit ou dangereux. En effet, l'algorithme de synthèse de lois de commande garantira le respect des contraintes spécifiées par le modèle du système contrôlé, comme dans les domaines de l'ordonnancement (Chretienne et al., 1997) et de la planification automatique (Ghallab et al., 2004).

Les propriétés qui viennent d'être présentées visent uniquement à vérifier le modèle du point de vue syntaxique. Pour cette raison en sus des propriétés précédentes, la section suivante propose un guide de modélisation visant à garantir la cohérence des informations contenues dans le modèle.

3 Guide de modélisation

Les différentes informations apportées dans cette section sur la démarche de modélisation se veulent être un guide. Elles n'ont pas la prétention d'être une méthodologie dont la définition nécessiterait à elle seule un mémoire de thèse.

De manière générale, la modélisation d'un système contrôlé et de son environnement commence par se focaliser sur les opérations (d'action et d'information) en fonction des services offerts par les chaînes fonctionnelles contrôlées. L'expert modélise ensuite les évolutions induites en s'appuyant sur les états des chaînes fonctionnelles atteints par les opérations. Enfin, la modélisation se termine par celle des évolutions requises en décrivant les effets de l'environnement sur toutes les variables d'états utilisées pour spécifier les opérations et les évolutions induites.

Qu'il s'agisse d'une opération ou d'une évolution, la démarche de modélisation est la même. Ainsi, nous présentons d'abord dans la section suivante la démarche générale à adopter par l'expert pour modéliser une opération d'action. Puis, la dernière section se focalise sur l'écriture des pré-contraintes et des contraintes.

3.1 Démarche de modélisation d'une opération d'action

La modélisation d'une opération d'action débute par la spécification de l'évolution de la chaîne fonctionnelle puis des évolutions associées du flux de produits. La structure de ces deux évolutions (ec_i et $ea_{i,j}$) étant identique, nous présenterons alors uniquement la démarche de modélisation d'une évolution d'une chaîne fonctionnelle.

Pour spécifier l'évolution de la chaîne fonctionnelle, l'expert doit commencer par écrire la condition puis l'effet transitoire et enfin l'effet final. Il complète ensuite la pré-contrainte et la contrainte qui sont fonction de la condition, de l'effet transitoire et de l'effet final.

Pour spécifier la pré-contrainte et la contrainte, l'expert doit visualiser les états pour lesquels les contraintes de sécurité et d'écologie sont ou risquent d'être violées. Afin de vérifier les propriétés 9, 10 et 11 sur la cohérence entre les spécifications d'une part des états interdits et

dangereux, et d'autre part des pré-contraintes et des contraintes, il est demandé à l'expert de formaliser non seulement la pré-contrainte et la contrainte mais également les états interdits et dangereux pouvant être atteints suite à l'effet de cette évolution. Ceci offre d'une part l'avantage de pouvoir vérifier les propriétés citées ci-dessus, et d'autre part de forcer l'expert à formaliser les états interdits et dangereux dont il a, dans tous les cas, une représentation informelle afin d'écrire les pré-contraintes et les contraintes.

Par exemple, pour l'opération *Sortir Vérin 1*, le vérin 2 est la seule chaîne fonctionnelle avec laquelle une interaction est possible. Les états interdits, appartenant à, $Q_I(ChF_{V1})$ sont spécifiés par la proposition suivante :

Vérin 1 en position sortie $\wedge$ Vérin 2 en position sortie

Quant aux états dangereux, appartenant à $Q_D(ChF_{V1})$, ils vérifient les propositions suivantes :

Vérin 1 en position sortie $\wedge$ Vérin 2 en position intermédiaire

Vérin 1 en position intermédiaire $\wedge$ Vérin 2 en position sortie

Vérin 1 en position intermédiaire $\wedge$ Vérin 2 en position intermédiaire

Tous les états dangereux vis-à-vis du vérin 1 sont ceux du sous-système composé du vérin 1 et de son environnement (vérin 2, magasin rotatif, produits présents entre A et H, entre A et B, en A, entre A et I, en I, et entre I et J) qui vérifient une des trois propositions ci-dessus.

Les ensembles $Q_I(ChF_{V1})$ et $Q_D(ChF_{V1})$ seront totalement définis quand toutes les opérations d'action basées sur le vérin 1 auront été spécifiées.

Pour compléter la démarche proposée, la section suivante se focalise sur l'écriture des pré-contraintes et de contraintes.

3.2 Écriture des pré-contraintes et des contraintes

Avant de se focaliser sur l'écriture à proprement parler des pré-contraintes et des contraintes, attardons-nous un instant sur les raisons qui autorisent leur écriture sans avoir connaissance de la demande et par conséquent des spécifications des produits.

Il est possible d'écrire les pré-contraintes et les contraintes car :

- soit les caractéristiques dimensionnelles des produits sont connues car les systèmes de transitique sont spécifiques (par exemple pour la plate-forme SAPHIR, les pièces qui circulent sont limitées en hauteur et ont deux diamètres possibles). Il est alors possible d'écrire les contraintes à partir des caractéristiques connues.
- Soit les spécifications des pièces sont totalement inconnues ce qui requiert alors des systèmes de transitiques basés sur un support commun, appelés palettes, à tous les types de produits. Dans ce cas, les pré-contraintes et les contraintes porteront sur ces supports servant au transport et sur le volume maximum pouvant être occupé par les produits.

D'une manière générale, les références produits ne doivent pas être modélisées dans le modèle du système de contrôlé ("ce qu'il est possible de faire au sens large"), au risque de tendre vers une modélisation de type loi de commande ("ce qui doit être fait").

Suite à cette précision sur la capacité à écrire les pré-contraintes et les contraintes sans disposer des spécifications exactes des produits, nous proposons maintenant de nous focaliser sur leur écriture.

En effet, malgré les propriétés énoncées, l'écriture des pré-contraintes et des contraintes reste un problème et notamment par le risque de sur-contraindre le modèle vis-à-vis du système réel et ne pas modéliser ainsi toutes les évolutions autorisées. Ainsi, des règles d'écriture des pré-contraintes et des contraintes sont proposées. Seule l'écriture de la pré-contrainte de l'évolution de la chaîne fonctionnelle d'une opération d'action est détaillée. Le principe reste identique d'une part pour la contrainte de l'évolution de la chaîne fonctionnelle, et d'autre part pour les pré-contraintes et les contraintes des évolutions associées du flux de produits.

La pré-contrainte de l'évolution de la chaîne fonctionnelle est spécifiée afin d'éviter d'atteindre un état dangereux ou interdit lors de l'évolution de la chaîne fonctionnelle à partir d'un état initial dans un état intermédiaire. Physiquement, ceci correspond à éviter les situations suivantes :

- Les collisions avec les autres chaînes fonctionnelles, ou toute autre interaction avec une autre chaîne fonctionnelle par des effets thermiques, magnétiques, etc. Par observation du système contrôlé et de son environnement, l'expert chargé de la conception de la partie opérative doit pouvoir déterminer pour une chaîne fonctionnelle les interactions possibles avec les autres. Alors, afin d'éviter une interaction, l'état de la chaîne fonctionnelle, avec laquelle cette interaction est possible, va être contraint. Par exemple, quand le vérin 1 évolue de sa position rentrée vers sa position intermédiaire, une collision est possible avec le vérin 2. Pour cette raison, le vérin 2 doit être en position rentrée.
- Conduire le flux de produits dans un état anormal suite, par exemple, à la chute d'un produit ou à sa détérioration. Pour éviter de telles conséquences, l'évolution de la chaîne fonctionnelle ne doit pas être autorisée pour un état du flux de produits depuis lequel il risque d'évoluer vers un état anormal. Pour cela, une proposition contraint uniquement l'état du flux de produit. Elle ne doit pas contraindre l'état des autres chaînes fonctionnelles au risque de sur-contraindre le modèle.
  Par exemple en fonctionnement normal, une loi de commande peut demander une sortie du vérin 1 quand un produit se trouve sur le magasin entre A et B ou entre A et H. Le produit risque alors de chuter. La proposition évitant cette chute est alors (Pas de P entre A et B) $\wedge$ (Pas de P entre A et H). Pour éviter le même risque, une proposition cette fois-ci sur l'état du magasin serait (Magasin arrêté) $\wedge$ (Magasin indexé). Le modèle sera sur-contraint et certaines évolutions autorisées ne seront pas modélisées. En effet, sans produit dans le magasin, la rotation du magasin est interdite pendant la sortie du vérin. Or, la rotation du magasin et la sortie du vérin 1 est une évolution autorisée quand il n'y a pas de produit dans le magasin.

Après avoir balayé toutes les interactions du vérin 1 avec son environnement et en avoir déduit un ensemble de propositions logiques, la pré-contrainte de l'évolution du vérin 1 est la proposition résultante du ET logique entre chacune de ces propositions logiques. La méthode reste la même pour écrire une pré-contrainte d'une évolution associée du flux de produits. Pour les contraintes aussi, le principe est le même mais cette fois-ci pour le passage de l'état intermédiaire à l'état final.

Malgré toutes les propriétés et la démarche de modélisation proposée, il est impossible à ce jour de garantir que le modèle soit correct. Cependant, il est demandé à l'expert de se focaliser successivement sur des parties restreintes du système contrôlé et de son environnement. Ces parties sont d'autant plus restreintes que l'expert s'intéresse à un moment donné uniquement à l'évolution d'une chaîne fonctionnelle ou à une des évolutions associées du flux de produits. Ce sous-système, d'une manière générale, sera de faible complexité du point de vue du nombre d'entités le composant (effecteurs et produits). En effet, une chaîne fonctionnelle sera très rarement en interaction avec plus de deux voire trois autres chaînes fonctionnelles.

Les champs de données d'une opération d'action, d'une opération d'information, d'une évolution induite et d'une évolution requise ainsi que la démarche de modélisation proposée sont structurants pour l'expert en charge de la modélisation. Ajouté à cela la vérification des propriétés présentées au début de ce chapitre, le modèle obtenu a *"toute les chances"* d'être correct.

De surcroît, l'expert peut être aidé par des analyses de sûreté préalablement réalisées pour la conception de la partie opérative (Villemeur, 1988) comme par exemple des arbres de défaillances (IEC-1025, 1990). De plus, aujourd'hui, l'expert dispose généralement d'une maquette virtuelle de la partie opérative avec le flux de produits à partir de la laquelle les logiciels de Conception Assistée par Ordinateur (CAO) offrent la possibilité de détecter automatiquement les interactions.

4 Conclusion

Soucieux d'apporter une assistance à l'expert chargé de la modélisation, ce chapitre s'est attaché à présenter deux orientations différentes permettant de contribuer à sa validation. La première permet une évaluation *a posteriori* du modèle obtenu. Pour se faire, elle met à disposition un ensemble de propriétés dont la vérification ne requiert pas de compétences supplémentaires de la part de l'expert ; la vérification peut se faire de manière automatique. La seconde orientation s'insère davantage dans une démarche d'accompagnement de l'expert. Elle propose un cadre de modélisation permettant à l'expert de structurer son observation du monde réel.

A ce stade de l'étude, tous les éléments sont disponibles pour envisager de proposer notre démarche de synthèse de lois de commande optimales. C'est ce que nous nous proposons de réaliser dans la partie suivante.

Troisième partie

Algorithme de Synthèse de Lois de Commande

Chapitre 7

Principes Généraux de l'Algorithme de Synthèse

1 Introduction

Dans la partie précédente, nous avons proposé un modèle du système contrôlé basé sur un ensemble d'opérations d'action, d'opérations d'information, d'évolutions induites, et d'évolutions requises. Les propriétés de ce modèle et le guide de modélisation proposés visent à assurer la cohérence des informations modélisées. La conception d'une loi de commande consiste alors à exploiter correctement et intelligemment cet ensemble d'informations afin, lors de l'exécution de cette loi, d'une part de répondre à la demande, et d'autre part de satisfaire les contraintes de sécurité et d'écologie.

Cette troisième partie, tout au long de laquelle la conception automatique de lois de commande est développée, débute par un chapitre consacré à l'exposé du principe général de l'algorithme de synthèse. Ainsi, l'algorithme que nous allons progressivement construire dans les chapitres suivants, est basé sur une technique finalement assez classique de recherche de chemins dans un graphe.

Dans le cadre de ce chapitre, nous allons nous attacher à sensibiliser le lecteur sur les limites inhérentes à l'utilisation d'une telle technique, et en second lieu à exposer deux principes fondamentaux permettant de dépasser ces limites.

2 Synthèse de Lois de Commande

Préalablement à la présentation de l'algorithme que nous proposons pour la synthèse de lois de commande, rappelons précisément le lieu dans lequel il vise à être implanté. La Figure 7.1 illustre la coordination de plusieurs chaînes fonctionnelles par un module de coordination qui est justement le niveau visé pour l'implantation des lois de commande conçues.

Suite à cette précision, nous rappelons que dans les parties précédentes nous avons présenté :

l'objectif d'une loi de commande dans le chapitre 2 § 2.1. Cet objectif est, suite à l'exécution de la loi de commande, d'imposer des évolutions à la partie opérative et aux flux de produits afin d'une part de répondre à une demande et d'autre part de satisfaire les contraintes de sécurité et d'écologie.

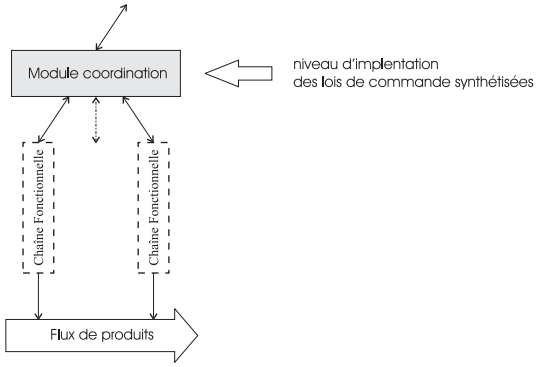


FIG. 7.1 – Niveau coordination auquel se place l'approche.

la demande à laquelle il faut répondre fixe, d'une part à la fois un état initial du système contrôlé et un ensemble d'états finaux, et d'autre part les performances attendues de l'évolution du système contrôlé de l'état initial à l'un des états finaux.

un modèle du système contrôlé et de son environnement, présenté dans la partie II, basé sur un ensemble d'opérations d'action, d'opérations d'information, d'évolutions induites et d'évolutions requises. Cette structuration du modèle a pour particularité de ne pas faire apparaître les états du système contrôlé et les transitions possibles entre ces états.

Ainsi, les données d'entrée de l'algorithme de synthèse sont d'une part le modèle du système contrôlé et de son environnement, et d'autre part la demande. A partir de ces entrées représentées sur la Figure 7.2, l'algorithme doit synthétiser une loi de commande en fonction de l'objectif défini ci-dessus.

Comme nous l'avons précisé en introduction, l'algorithme de synthèse va être basé sur une technique de recherche de chemins dans un graphe. Cependant, comme nous allons le constater dans la suite de ce chapitre, ces techniques ne sont pas directement utilisables à partir du modèle du système contrôlé proposé dans la partie II.

3 Limites d'utilisation des techniques de recherche de chemins dans un graphe

Avant d'étudier l'utilisation des techniques de recherche de chemins dans un graphe pour notre problème et leurs limites, nous présenterons d'abord les techniques existantes de recherche de chemins.

3.1 Techniques de recherche de chemins dans un graphe

La représentation des évolutions autorisées du système contrôlé est possible par un graphe dont un sommet représente un état du système contrôlé et un arc est l'image d'une évolution du système contrôlé entre deux états. Ainsi, le poids des arcs sera, suivant le critère à optimiser, une des caractéristiques disponibles des opérations/évolutions (durée, coût énergétique, coût financier, quantité de déchets).

Le graphe ainsi obtenu est un graphe orienté et valué dont le poids des arcs est positif ou nul à la vue des caractéristiques des opérations/évolutions.

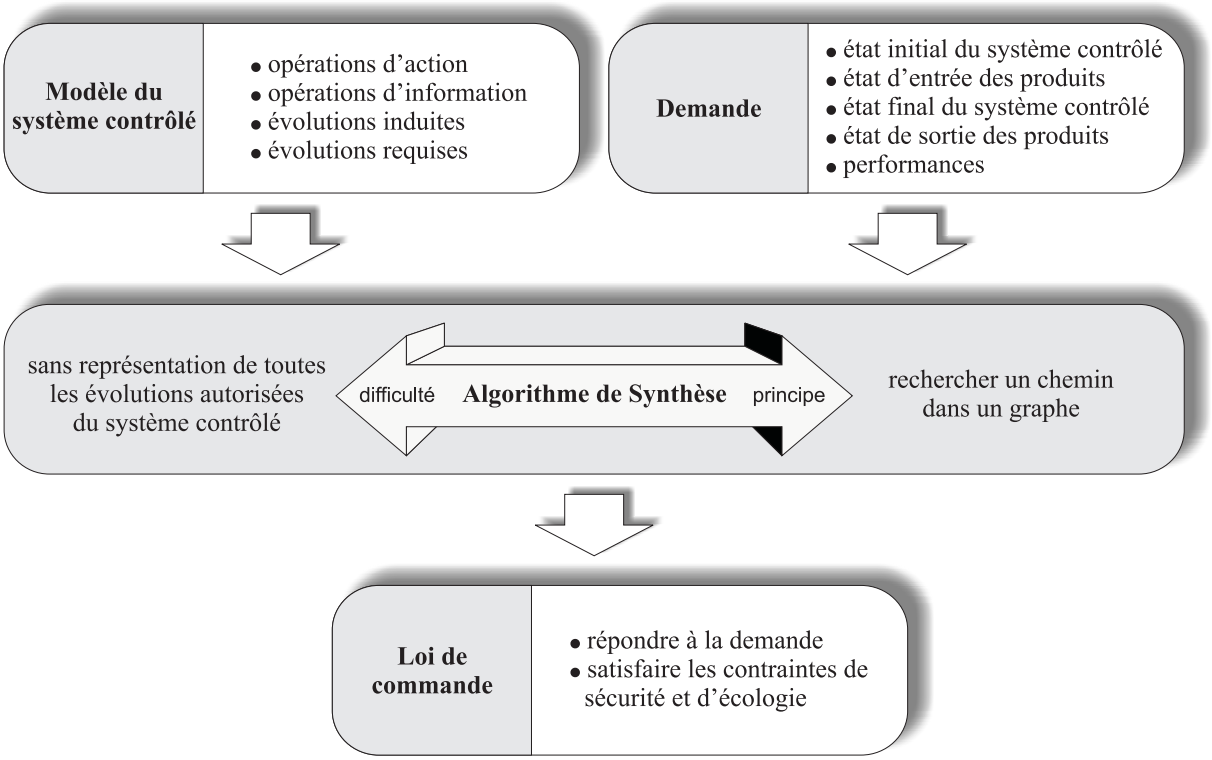


FIG. 7.2 – Données en entrée de l'algorithme de synthèse.

En supposant disposer de cette représentation du système contrôlé, une technique de recherche du plus court chemin est applicable pour déterminer les évolutions à imposer au système contrôlé afin de le faire passer d'un état initial à un état final.

Deux catégories d'algorithmes sont proposées dans la littérature : la recherche du plus court chemin entre une origine unique et tous les autres sommets du graphe (algorithme de Bellman-Ford, ou de Dijkstra), et la recherche du plus court chemin pour tous les couples de sommets (algorithme général, ou algorithme de Floyd-Wharshall). La recherche du plus court chemin qui nous intéresse est celle entre un sommet unique correspondant à l'état initial et un de l'ensemble des sommets représentant les états finaux. Le fait que le poids des arcs du graphe représentant les évolutions du système contrôlé est positif ou nul ne permet pas de choisir entre l'algorithme de Dijkstra ou celui de Bellman-Ford. Cependant, la complexité de chacun (Dijkstra de complexité $O(S)$, avec S le nombre de sommets, Bellman-Ford de complexité $O(S.A)$, avec A le nombre d'arcs) est en revanche un élément de choix important, notamment dans le cadre d'une reconfiguration suite à l'occurrence d'une défaillance. Notre choix s'oriente donc naturellement vers l'algorithme de Dijkstra (Dijkstra, 1959).

Afin d'utiliser cet algorithme, il est nécessaire de disposer du graphe représentant l'ensemble des évolutions du système contrôlé. Cependant, comme nous allons le voir par la suite, générer ce graphe dans le contexte particulier d'un procédé manufacturier complexe est illusoire.

3.2 Adéquation de l'algorithme de Dijkstra

Afin d'aboutir à un graphe correspondant aux évolutions autorisées du système contrôlé, nous devons d'abord générer le système de transition d'états correspondant. Pour cela, nous disposons du modèle du système contrôlé basé sur les quatre catégories d'opérations : opérations d'action et d'information, évolutions induites et requises. La transformation du modèle proposé dans la partie II vers une représentation à états consiste depuis un état à rechercher l'ensemble des opérations exécutables et les parallélismes autorisés entre ces opérations et celles déjà en cours. Ainsi, toutes les évolutions autorisées depuis cet état sont générées ainsi que l'état atteint pour chacune des évolutions autorisées. Ce principe est de nouveau appliqué depuis chacun des nouveaux états atteints. Quand il n'existe plus d'états depuis lesquels les évolutions autorisées n'ont pas été recherchées, le système de transition d'états représentant les évolutions du système contrôlé est obtenu.

Cependant, la complexité évaluée en terme de nombre d'états et de transitions de ce système de transition d'états risque d'être beaucoup trop grande pour envisager de le construire avec rapidité et efficacité. Pour les procédés auxquels nous nous intéressons, cette complexité résulte des trois caractéristiques suivantes :

- plusieurs produits évoluent simultanément ;
- les opérations et les évolutions à coordonner sont de natures différentes : de transformation ou de contrôle des produits, de déplacement des produits et enfin de préparation (Artigues, 1997).
- de nombreux parallélismes d'exécution existent dûs en partie à la présence de plusieurs produits et à la nature des opérations/évolutions, mais aussi entre des opérations/évolutions de même nature.

Face à cette trop grande complexité, une technique de recherche d'un chemin dans un graphe ne semble pas pouvoir être appliquée directement sur le graphe correspondant à l'ensemble des évolutions autorisées du système contrôlé. Ainsi, la section suivante développe un ensemble de propositions visant à réduire cette complexité tout en maintenant des performances.

3.3 Vers la recherche d'un compromis complexité/performances

Comme nous venons de le démontrer, il est illusoire de générer le système de transition d'états complet dans le cadre de la synthèse de lois de commande, en particulier en ligne. Cependant, nous remarquerons que l'hypothèse de travail sur le système de transition d'états complet du système contrôlé et de son environnement est une condition sine qua non à l'optimisation globale d'un critère. Rappelons néanmoins que la démarche de spécification d'une loi de commande est réalisée actuellement par un expert ne garantissant donc aucune performance (Rossi, 2003). Nous proposons donc de définir un compromis entre la performance de l'algorithme en terme de temps de calcul et les performances de la solution obtenue.

Le challenge est alors de conserver des performances acceptables tout en réduisant au maximum la complexité du ou des systèmes de transitions d'états à partir desquels nous recherchons un chemin optimal. Les performances acceptables de la solution sont au minimum celles offertes généralement par un expert. Ainsi à performances égales de la solution, le temps de conception des lois de commande sera quant à lui considérablement réduit.

Comme nous l'avons vu dans la section précédente, rechercher ce compromis revient à définir

les leviers permettant de réduire la complexité du système de transitions d'états. Au vu des origines de cette complexité présentées au § 3.2, la section suivante sera naturellement organisée autour de l'étude de chacun de ces trois leviers : le nombre de produits traités en parallèle, la nature des opérations/évolutions et enfin les parallélismes d'exécution.

4 Principes Adoptés

Le premier de ces principes se focalise sur la décomposition du problème de conception en fonction de la nature des opérations. Le second principe limite le nombre de produits présents dans une représentation d'états. Et enfin, le troisième et dernier principe porte sur la prise en compte différée des parallélismes d'exécution.

4.1 Décomposition en sous-problèmes

Le problème de conception d'une loi de commande est décomposé par le premier principe en se basant sur les trois types de comportements possibles d'une opération.

4.1.1 Trois types de comportements d'une opération

Lors de la modélisation du système contrôlé, présentée dans la partie 2, l'expert recherche les évolutions des chaînes fonctionnelles, de la position des produits et enfin de leur état. Ainsi, un état du système contrôlé et de son environnement est décrit par un ensemble de variables d'état VE qui est partitionné en trois sous-ensembles :

$$VE = \bigcup_{i=1}^{3} VE_i = VE_1 \cup VE_2 \cup VE_3$$

- VE_1, les variables d'état relatives à l'état physique des produits liées aux spécifications des produits (forme géométrique, dureté, état de surface, couleur, matériaux) ;
- VE_2, les variables d'état qui décrivent la position spatiale des produits dans le système de production ;
- VE_3, les variables d'état des chaînes fonctionnelles du point de vue du niveau coordination.

un comportement de **transformation**	un comportement de **transitique**	un comportement de **préparation**	
modifie	*ne peut pas modifier*	*ne peut pas modifier*	**l'état physique du flux de produits (VE_1)** (géométrie, état de surface, assemblage, etc)
peut modifier	*modifie*	*ne peut pas modifier*	**la position spatiale du flux de produits (VE_2)**
peut modifier	*peut modifier*	*modifie*	**l'état des chaînes fonctionnelles (VE_3)**

FIG. 7.3 – Relations entre la nature d'un comportement d'une opération et les trois sous-ensembles de variables d'état.

En fonction de sa capacité ou non à modifier les variables des trois sous-ensembles présentés ci-dessus, le comportement d'une opération est de nature différente : de transformation, de transitique, ou de préparation. La Figure 7.3 résume les relations entre la nature d'un comportement et ses effets sur les variables.

Remarque : l'analyse de ce tableau conduit naturellement à définir les différents comportements que peut avoir chaque catégorie d'opérations :

- une opération d'action ou d'information peut avoir un comportement de transformation, un comportement de transitique ou un comportement de préparation ;
- une évolution induite peut avoir un comportement de transformation ou un comportement de transitique ;
- une évolution requise peut avoir un comportement de transformation, un comportement de transitique ou un comportement de préparation.

Basée sur ces trois types de comportements différents, une décomposition de l'algorithme de synthèse de loi de commande est proposée dans la section suivante.

4.1.2 Décomposition

Le problème global de synthèse de loi de commande est décomposé en trois sous-problèmes liés à chacun des types de comportements.

Ainsi pour les systèmes de transitions d'états construits, un seul type de variable d'état évoluera ; celui de l'état physique des produits (VE_1), ou celui de la position des produits (VE_2), ou bien encore celui de l'état des chaînes fonctionnelles (VE_3). Ainsi, il sera successivement résolu trois problèmes :

1. un problème de transformation des produits,
2. un problème de transitique de façon à faire évoluer la position des produits,
3. un problème de préparation des chaînes fonctionnelles afin de les placer dans un état compatible avec l'exécution de futures opérations de transformation ou de transitique.

En effet d'après la Figure 7.3, l'ajout d'opérations de transitique à une séquence d'opérations de transformation n'affectera pas l'évolution de l'état physique des produits, de même que l'ajout d'opérations de préparation ne modifiera par la position et l'état des produits.

Par construction, les systèmes de transitions d'états construits ne contiendront aucun parallélisme d'exécution entre les opérations de transformation, les opérations de transitique, et les opérations de préparation. Ces parallélismes seront réintroduits ultérieurement lors de l'assemblage des solutions de chacun des trois sous-problèmes.

Le niveau des performances de la solution finale est conservé suite à cette décomposition sous deux hypothèses qui sont maintenant présentées.

La première hypothèse posée est l'impact mineur sur les performances du découplage des problèmes de transformation, de transitique et de préparation. Cela signifie que les comportements de transitique et de préparation des opérations n'ont pas d'impact sur l'optimalité de la solution du problème de transformation. De même, les comportements de préparation n'ont

pas d'impact sur la solution du problème de transitique. Mais ce principe de décomposition nécessite une seconde hypothèse détaillée ci-dessous.

Pour appliquer un tel principe, une hypothèse supplémentaire est à faire sur l'écriture des propositions définissant les conditions, les pré-contraintes et les contraintes. En effet, il est possible de découpler les évolutions des variables à condition de pouvoir découpler sur les trois sous-ensembles de variables d'état les propositions logiques spécifiant les conditions, les pré-contraintes et les contraintes. Aussi, toute proposition d'une condition, d'une contrainte ou d'une pré-contrainte peut être mise sous la forme : (proposition sur des variables d'état physique) ET (proposition sur la position des produits) ET (proposition sur l'état des chaînes fonctionnelles). Néanmoins, une proposition Q sur la position des produits peut être conditionnée par une proposition P sur l'état physique et la position des produits. La proposition globale sur la position du flux de produits s'écrit alors $P \Rightarrow Q$. De même, une proposition Q sur l'état des chaînes fonctionnelles peut être conditionnée par une proposition P sur l'état physique et la position des produits ainsi que sur l'état des chaînes fonctionnelles.

Suite au principe de décomposition présenté dans cette section, le paragraphe suivant propose de geler momentanément toute prise en compte de parallélismes résiduels afin de générer un graphe de complexité moindre.

4.2 Gel des parallélismes résiduels

Afin de réduire encore la complexité du graphe recherché, nous proposons d'appliquer deux principes supplémentaires.

Le premier de ces principes vise à restreindre la vision du système de transitions d'états à un seul produit permettant ainsi de limiter sa complexité mais vraisemblablement au détriment d'une future optimisation dans un contexte multi-produits. Cependant, comme nous le verrons au chapitre 9, lorsque la demande porte sur plusieurs produits, les séquences d'actions générées pour chacun des produits pourront ensuite être assemblées de manière à ré-introduire les parallélismes d'exécution non pris en compte jusqu'alors.

Ainsi, sous ce principe, si le système de transitions d'états était généré, il ne représenterait que l'évolution de l'état d'un seul produit. Aussi, seules les opérations de transformation et de transitique seraient affectées puisque les variables d'état des produits n'évoluent pas pour un problème de préparation.

A ce stade, des parallélismes résiduels seraient encore observables entre les opérations de préparations (mise en marche d'un convoyeur et rentrée du vérin 2 par exemple). Afin de limiter encore la complexité du graphe recherché, un dernier principe vise à différer la prise en compte de ces derniers parallélismes.

Basé sur ces trois principes (décomposition, focalisation sur un produits et gèle des parallélismes résiduels), il est alors possible, comme le présente la section suivante, de ne pas tenir compte des états intermédiaires et de limiter ainsi, de manière *"ultime"*, la complexité du système de transitions d'états à considérer.

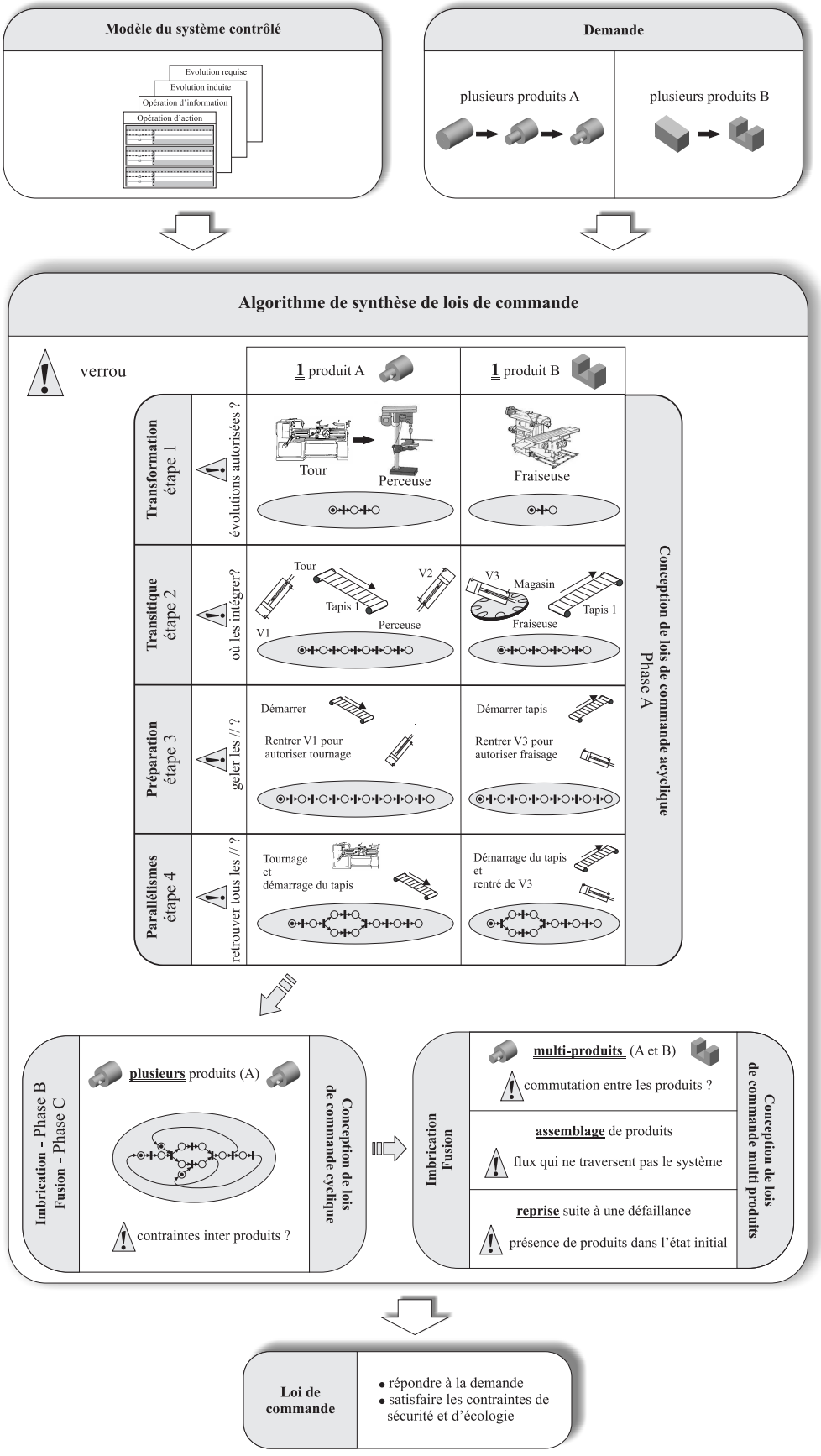


FIG. 7.4 – Demarche globale de l'algorithme de synthèse.

4.3 Conséquence sur la représentation d'états

L'application de ces principes conduit à des systèmes de transitions d'états sans aucun parallélisme d'exécution. Ainsi, et pour chacun des niveaux de décomposition (opération de transformation, de transitique et de préparation) depuis un état atteint suite au lancement d'une opération, le seul événement attendu correspond à la fin de cette même opération. Ainsi, la représentation des états pendant l'exécution d'une opération n'est d'aucune utilité du point de vue de la recherche du plus court chemin. Ainsi, les états intermédiaires représentant l'état du système contrôlé durant l'exécution d'une opération ne seront pas représentés dans les systèmes de transitions d'états. La complexité des espaces d'états sera alors réduite d'autant d'états et de transitions que d'opérations exécutées. Ainsi, les étiquettes seront réduites au nom de l'opération.

5 Conclusion

Afin de synthétiser une loi de commande, le principe général proposé dans ce chapitre découle d'une part de la modélisation du système contrôlé, et d'autre part de la demande limitée à la spécification d'un état final à atteindre. Deux stratégies s'opposent alors, la première consiste à construire le graphe complet, d'une complexité généralement non maîtrisable, mais permettant de trouver la solution optimale, la seconde, utilisée en planification automatique, se limite à construire un chemin d'un état initial à un état final sans aucune optimisation. L'originalité de l'approche réside dans la recherche d'un compromis de façon à maîtriser la complexité du graphe tout en optimisant, de façon locale, un ou plusieurs critères. Ce compromis est basé sur deux principes : la décomposition en trois sous-problèmes qui se focalisent chacun sur une classe d'opération, et le gel des parallélismes lors de la représentation des évolutions du système contrôlé. Les parallélismes d'exécution sont ensuite ré-introduits. Ce deuxième principe présuppose que toute loi de commande est exprimable comme une séquence d'opérations.

La simplicité apparente de la démarche de synthèse proposée ici est le fruit de la structuration du modèle proposé. Cependant, comme en attestent les deux chapitres suivants, au delà de cette apparente simplicité structurelle se cache un nombre très élevé de problèmes de faible granularité, certes, mais cependant capitaux pour assurer l'intégrité globale de l'algorithme que nous cherchons à développer. Afin de faciliter la lecture des deux chapitres suivants, qui rentrent au plus profond de l'algorithme de synthèse, nous mettons à disposition du lecteur une représentation graphique globale (cf. Figure 7.4) de la démarche de synthèse proposée. Cette Figure sera reprise, à tous les changements d'étapes qui sont détaillées dans les chapitres 8 et 9.

Avant de les présenter, prenons le temps de commenter cette Figure. Comme nous pouvons le remarquer, cette Figure reprend et détaille celle indicée 7.2 page 111. Sur la base d'une mise en situation exprimée par une demande de production de plusieurs produits de type A et plusieurs produits de type B, l'algorithme proposé est découpé en 3 phases plus une phase permettant de traiter les cas particuliers liés au contexte multi-produits.

La phase A est elle même décomposée en quatre étapes. Elles consistent, pour un seul produit de type X, à progressivement générer une loi de commande dite acyclique. Ensuite, les phase B et C permettent, pour un type X de produit de générer la loi de commande dite cyclique capable de gérer le flux de produits X. Pour n produits, les phases A, B et C sont répétées. Lorsque n est supérieur à un, alors, et selon le cas (multi-produits, assemblage, reprise), un processus d'imbrication et fusion est proposé entre les lois de commande cycliques de chaque produit.

Chapitre 8

Phase A : Synthèse d'une Loi de Commande Mono-Produit

1 Introduction

Comme nous avons pu le voir dans le chapitre précédent, il est illusoire de prétendre générer l'espace d'états atteignables pour un modèle du type manufacturier complexe. Pourtant, concevoir une loi de commande revient dans l'absolu à un problème de recherche de chemin dans cet espace d'états. Afin de réussir ce mariage délicat, nous avons proposé une démarche générale de décomposition du problème en trois problèmes de complexité moindre ; le revers de la médaille, une perte vraisemblable d'une partie des performances de la solution globale.

Sur cette base, ce chapitre se propose de développer la première phase de la méthode de synthèse proposée (cf. phase A Figure 7.4). Cette dernière, articulée autour des trois classes d'opérations retenues et du gel provisoire des parallélismes inhérents au modèle est discutée et argumentée dans les trois premières sections du chapitre. La dernière quant à elle, présente les mécanismes de réintégration des parallélismes *"gelés"*.

2 Principe Général

Le problème particulier, traité dans ce chapitre, est celui de la spécification d'une loi de commande qui doit répondre à une demande pour laquelle un seul produit entre dans un état donné et ressort du sous-système considéré dans un état physique attendu.

Conformément à la démarche présentée dans la Figure 7.4 du chapitre précédent, nous résumons ici le principe général de résolution du problème (Henry et al., 2004a) :

étape 1 : résolution de la problématique liée à la transformation d'un produit afin de le faire évoluer de son état d'entrée vers son état de sortie ; tous les deux spécifiés par la demande.

étape 2 : résolution de la problématique liée à la transitique afin de satisfaire d'une part la demande et d'autre part les conditions et les pré-contraintes des opérations de transformation sur la position du produit.

étape 3 : résolution de la problématique de préparations des chaînes fonctionnelles afin de satisfaire d'une part la demande et d'autre part les conditions et les pré-contraintes des actions de transformation et de transitique sur l'état des chaînes fonctionnelles.

étape 4 : optimisation du temps d'exécution par la ré-introduction des parallélismes gelés.

Remarque : nous ferons ici l'hypothèse de l'absence d'incertitude sur l'état d'entrée du produit qui conduirait à des états de sortie différents et finalement à un problème multi-produits traité au chapitre 9.

3 Phase A, étape 1 : Transformation du produit

3.1 Principe

La première étape vise à répondre à une partie seulement de la demande en se focalisant sur l'état physique du produit afin qu'il passe de son état d'entrée à son état de sortie avec les performances spécifiées. Ainsi, seules les opérations avec un comportement de transformation qui ont un effet sur les variables d'états liées à l'état physique du flux de produits sont utilisées dans cette étape.

Ainsi, un état q_1 du système contrôlé et de son environnement qui se limite à l'état physique du produit appartient à l'ensemble Q_1 défini par $Q_1 = \prod_{x \in VE_1} V_x$ avec V_x l'ensemble des valeurs de la variable d'état x. Un état $q_1 \in Q_1$ est défini par $q_1 = \{(x = c) | x \in VE_1\}$, où $c \in V_x$. L'espace d'états considéré dans cette première étape est celui décrivant les évolutions de l'état physique du produit. Cet espace d'états correspond au système de transition d'états D_1. Il est défini par un triplet $D_1 = (Q_1, C_1, \delta_1)$ avec Q_1 l'ensemble des états défini ci-dessus, C_1 l'ensemble des comportements de transformation issu de l'ensemble des opérations et δ_1 la fonction de transition de $Q_1 \times C_1$ dans Q_1.

Pour un état initial $q_{1,0}$, décrivant l'état physique d'entrée d'un produit, et un objectif Ob_1 défini par une ou plusieurs propositions sur l'état physique de sortie du produit, la première étape consiste alors à trouver un chemin optimal dans D_1 de l'état initial $q_{1,0} \in Q_1$ à un état objectif $q_1 \in Q_1$ tel que Ob_1 soit satisfait (cf. Figure 8.1).

Afin de construire uniquement la partie utile de l'espace d'états limité à la transformation du produit et réduire ainsi le temps requis nécessaire à le générer, seuls les états atteignables depuis l'état initial, $q_{1,0}$, pour un objectif donné, Ob_1, sont considérés. L'espace d'états est alors représenté par D_1 défini par un quintuplet $D_1 = (Q_{A1}, C_1, \delta_1, q_{1,0}, Ob_1)$ avec $Q_{A1} \subseteq Q_1$ l'ensemble restreint des états atteignables depuis $q_{1,0}$ pour l'objectif Ob_1.

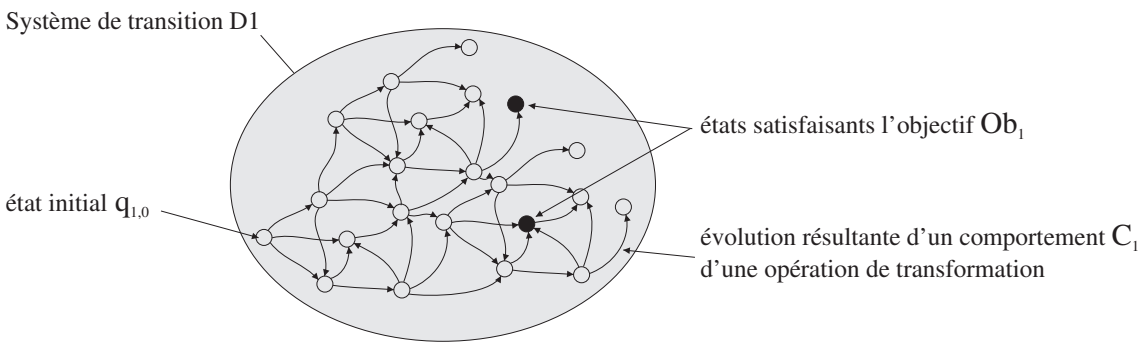


FIG. 8.1 – Espace d'états atteignables résultant uniquement de la transformation du produit.

3.2 Espace restreint d'états atteignables

La construction de l'espace restreint d'états atteignables par transformation du produit depuis un état initial $q_{1,0}$ pour un objectif Ob_1 est présentée progressivement en commençant par la fonction qui établit la liste des opérations avec un comportement de transformation exécutables depuis un état q_1. Puis basée sur cette fonction, la génération de l'espace d'états atteignables est ensuite détaillée. Afin de limiter la complexité de cet espace d'états, nous présentons une condition restrictive fonction de l'objectif.

3.2.1 Opérations avec un comportement de transformation

La construction de l'espace d'états atteignables limité à des évolutions de l'état physique d'un produit nécessite de déterminer pour un état toutes les opérations dont l'effet vient à modifier l'état physique du produit (cf. Figure 8.2). Ainsi, nous définissons une fonction $C_1(q_1)$ qui renvoie pour un état $q_1 \in Q_1$ l'ensemble des comportements d'opération qui sont susceptibles de transformer le produit ($C_1(q_1) = \{c_{1,i}\}$).

Depuis un état donné, le comportement d'une opération va transformer l'état physique d'un produit s'il comporte au moins une évolution associée du flux de produits dont l'effet transitoire ou final modifie l'état du produit. Mais, l'opération pourra être exécutée si toutes les pré-contraintes des évolutions associées du flux de produits dont la condition est satisfaite sont également satisfaites. Considérant uniquement l'état physique des produits lors de cette étape, les conditions et les pré-contraintes seront à satisfaire uniquement vis-à-vis des variables d'états décrivant l'état physique des produits. Ainsi, toute opération qui vérifie le test suivant pour un état $q_1 \in Q_1$ a un comportement dit de transformation depuis cet état :

Il existe au moins une évolution du flux de produits, composant l'opération ($ea_{i,j}$, $ep_{i,j}$ ou $es_{i,j}$), pour laquelle :
- la condition sur l'état physique du flux de produits est vraie ;
- la pré-contrainte sur l'état physique du flux de produits est satisfaite ;
- l'effet transitoire ou l'effet final modifie l'état physique du flux de produits.

Il n'existe pas d'évolution du flux de produits, composant l'opération ($ea_{i,j}$, $ep_{i,j}$ ou $es_{i,j}$), pour laquelle :
- la condition sur l'état physique du flux de produits est vraie ;
- la pré-contrainte sur l'état physique du flux de produits n'est pas satisfaite ;
- l'effet transitoire et l'effet final modifient l'état physique du flux de produits.

Pour l'évolution de la chaîne fonctionnelle de l'opération (ec_i ou ee_i), quand il y en a une :
- la pré-contrainte sur l'état physique du flux de produits est satisfaite.

Par application du test énoncé ci-dessus à chacune des opérations du modèle du système contrôlé, l'ensemble des opérations avec un comportement de transformation est alors déterminé.

A partir de la détermination de cet ensemble de comportements, la génération de l'espace d'états atteignables depuis un état initial est maintenant exposée.

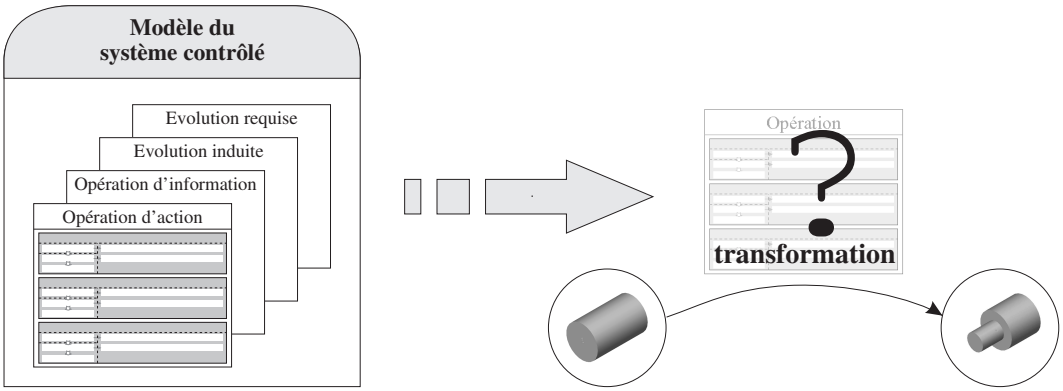


FIG. 8.2 – Extraction du modèle des opérations avec un comportement qui transforme le produit.

3.2.2 Espace d'états atteignables

Afin de construire uniquement l'espace d'états atteignables depuis un état initial $q_{1,0}$, l'algorithme utilise une technique de progression vers l'avant à partir cet état initial. La seconde technique envisageable est basée sur un mécanisme de régression depuis chacun des états satisfaisant l'objectif. Dans ce cas, l'espace d'états obtenu sera généralement plus grand que l'espace d'états atteignables depuis l'état initial.

L'algorithme de construction des états atteignables par des opérations de transformation depuis un état initial $q_{1,0}$ est un algorithme glouton (Lacomme et al., 2003) qui génère lui même de nouveaux états à visiter.

Ainsi, il va mettre à jour deux ensembles d'états :

- l'ensemble des états à visiter, noté Q_v ;
- l'ensemble des états traités, noté Q_{A1}.

Le principe de l'algorithme est alors le suivant :

- déterminer tous les comportements de transformation (cf. § 3.2.1) existants depuis un état de l'ensemble des états à visiter. Cet état est retiré des états à visiter et ajouté à l'ensemble des états traités ;
- calculer l'état atteint suite à l'application de chacun de ces comportements ;
- si l'état atteint est un nouvel état, il est alors ajouté à l'ensemble des états à visiter.

L'algorithme se termine bien évidemment lorsque l'ensemble des états à visiter, Q_v, est vide. L'ensemble des états atteignables est alors donné par l'ensemble des états traités. Le système de transitions d'états $D_1 = (Q_{A1}, C_1, \delta_1, q_{1,0})$ est ainsi construit.

La Figure 8.3 représente d'une part les deux ensembles d'états mis à jour, et d'autre part les comportements de transformation existants depuis l'état $q_{1,4}$ avec les états atteints suite à leur application. Seul l'état $q_{1,7} = \delta(q_{1,4}, c_{1,3})$ atteignable depuis $q_{1,4}$ est un nouvel état qui est par conséquent ajouté à l'ensemble Q_v des états à visiter.

Pour chacun des comportements, l'état $q"_{1,4}$ atteint suite à l'exécution de l'opération ayant ce comportement est calculé d'après les définitions 2 et 4. Comme énoncé au chapitre 7, l'état intermédiaire durant l'opération n'est pas représenté.

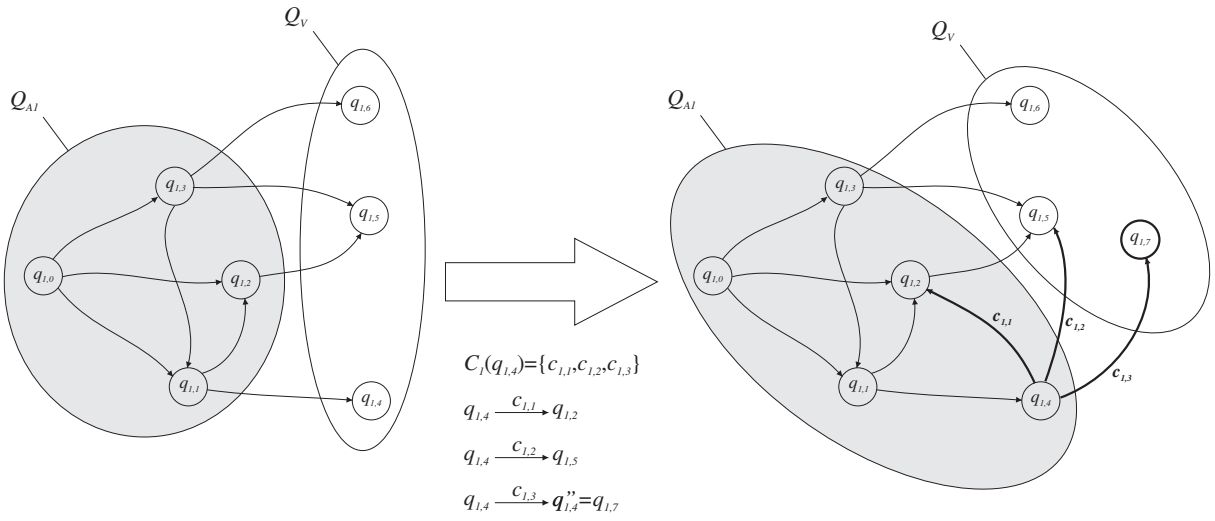


FIG. 8.3 – Construction des états atteignables par transformation du produit depuis un état initial $q_{1,0} \in Q_1$.

Fonction EtatsAtteignables($q_{1,0}$) : **système de transition d'états**
  $Q_v = \{q_{1,0}\}$
  $Q_{A1} = \{\emptyset\}$
  $\delta(q_1, c_1)$ est vide
  Tant que $(Q_v \neq \{\emptyset\})$ **faire**
    $C_{transformation} = C_1(q_1)$ tel $q_1 \in Q_v$
    $Q_v = Q_v \setminus \{q_1\}$
    $Q_{A1} = Q_{A1} \cup \{q_1\}$
    Tant que $(C_1(q_1) \neq \{\emptyset\})$ **faire**
      $q"_1 = \delta(q_1, c_1)$ tel que $c_1 \in C_1(q_1)$
      $C_{transformation} = C_{transformation} \setminus \{c_1\}$
      Si $(q"_1 \notin Q_{A1}$ et $q"_1 \notin Q_v)$ **Alors**
        $Q_v = Q_v \cup \{q"_1\}$
      Fin Si
    Fin Tq
  Fin Tq
  Retourner $D_1 = (Q_{A1}, C_1, \delta_1, q_{1,0})$;
Fin

ALG. 1: Algorithme de construction de l'espace d'états atteignables par transformation du produit depuis un état initial $q_{1,0} \in Q_1$.

Afin de ne pas construire des états et des transitions inutiles et ainsi réduire la taille du système de transition $D1$ et le temps pour le générer, une condition présentée dans la section suivante restreint les états à explorer depuis un état q_1 satisfaisant Ob_1.

3.2.3 Condition restrictive liée à l'objectif

L'algorithme 1 construit l'espace complet d'états atteignables depuis un état initial. Ainsi même après un état satisfaisant l'objectif, tous les états atteignables qui sont inutiles sont également construits. Il est cependant possible de limiter la complexité de l'espace d'états en ne construisant pas tous les états atteignables depuis un état objectif. Cette condition restrictive nécessite d'abord de différencier les états objectifs des autres, et ensuite de leur appliquer un traitement spécifique quand ils sont visités.

Tous les états qui satisfont l'objectif Ob_1 spécifié par une proposition logique sur l'état physique du produit sont des états appelés états objectifs. Afin de les différencier de ceux ne satisfaisant pas l'objectif, une liste d'états objectifs est crée.

Si un nouvel état atteint satisfait l'objectif, il sera alors ajouté à la liste des états objectifs en plus d'être ajouté à la liste des états à visiter. La construction de l'ensemble des états objectifs est donnée par l'algorithme 2.

Fonction EtatsAtteignables($q_{1,0}$,Ob_1) : **système de transition d'états**
  $Q_{1,Ob} = \{\emptyset\}$
  ...
  Si (q"$_1$ satisfait Ob_1) **Alors**
    $Q_{1,Ob} = Q_{1,Ob} \cup \{q"_1\}$
  Fin Si
  ...
  Retourner $D_1 = (Q_{A1}, C_1, \delta, q_{1,0}, Q_{1,Ob})$;
Fin

ALG. 2: Complément à l'algorithme 1 afin de créer l'ensemble des états objectifs.

Si le modèle était constitué uniquement d'opérations, il n'y aurait pas d'intérêt à explorer l'espace d'états atteignables depuis un état objectif. Mais le niveau coordination ne contrôlant pas les évolutions requises et induites, elles peuvent très bien être exécutées depuis un état objectif et conduire dans un état qui ne le satisfait plus. Par exemple, pour une transformation par cuisson, l'évolution induite représentant l'évolution de l'état du produit après la durée normale de cuisson devra être prise en compte. En effet en fonction de l'impact du dépassement de la durée de cuisson, cette évolution conduira soit dans un état satisfaisant toujours l'objectif (métal en fusion restant à température constante) ou au contraire dans un état ne satisfaisant plus l'objectif (cuisson de produits alimentaires).

Afin de tenir compte d'une telle situation, il est nécessaire de construire depuis un état objectif l'espace d'états atteignables par des évolutions induites ou des évolutions requises. Ceci implique d'abord de modifier la fonction $C_1(q_1)$ qui détermine les comportements de transformation existants depuis un état. Elle doit renvoyer pour un état objectif uniquement les comportements d'évolutions induites ou d'évolutions requises. Puis en fonction du respect de l'objectif, ou non, l'état atteint suite à un de ces comportements est traité différemment. Par exemple, pour une évolution induite, si l'état atteint ne satisfait plus l'objectif, il est nécessaire d'interdire son exé-

cution. Pour cela, la ou les conditions conduisant à l'obtention de l'évolution induite devront ne plus être satisfaites, par exemple par une opération d'arrêt d'un four. L'ajout de cette opération de préparation à l'étape 3, présentée plus loin dans ce chapitre, sera possible à condition d'avoir représenté depuis l'état objectif l'évolution induite qui conduit à ne plus satisfaire cet objectif. L'état, suite à cette évolution, n'étant jamais atteint par principe, il n'a pas à être visité ; il est donc directement ajouté à l'ensemble des états traités Q_{A1} afin de mémoriser l'existence de cette évolution induite.

A l'inverse, si l'état atteint depuis un état objectif suite à une évolution induite satisfait toujours l'objectif, il est ajouté à l'ensemble des états à visiter Q_v et à l'ensemble des états objectifs $Q_{1,Ob}$.

L'algorithme 3 propose l'intégration des traitements proposés ci-dessus à l'algorithme 2.

Fonction EtatsAtteignables($q_{1,0}$,Ob_1) : **système de transition d'états**
  ...
  Si (($q"_1 \notin Q_{A1}$ ET $q"_1 \notin Q_v$) OU ($q_1 \in Q_{1,Ob}$ ET $q"_1$ satisfait Ob_1)) **Alors**
    $Q_v = Q_v \cup \{q"_1\}$
  Sinon
    Si ($q_1 \in Q_{1,Ob}$ ET $q"_1$ ne satisfait pas Ob_1) **Alors**
      $Q_{A1} = Q_{A1} \cup \{q"_1\}$
    Fin Si
  Fin Si
  ...
  Retourner $D_1 = (Q_{A1}, C_1, \delta, q_{1,0}, Q_{1,Ob})$;
Fin

ALG. 3: Condition restrictive d'exploration des états atteignables depuis un état objectif.

A partir de la représentation par un système de transitions d'états, D_1, de l'espace restreint d'états atteignables, un chemin doit être trouvé de l'état initial vers un état satisfaisant l'objectif. Même si il est reconnu que le choix de l'algorithme de recherche d'un chemin dépend des caractéristiques intrinsèques du système contrôlé, notamment en terme d'existence de redondances fonctionnelles, nous avons volontairement proposé un algorithme capable de faire face à la diversité des systèmes contrôlés, au détriment parfois des performances mêmes de l'algorithme.

3.3 Chemin optimal

L'algorithme de recherche d'un chemin optimal afin d'atteindre l'objectif Ob_1 depuis un état initial $q_{1,0}$ est d'abord proposé avec un critère décisionnel unique. Puis, nous nous intéressons à la problématique multi-critères.

3.3.1 Problème mono-critère

La recherche d'un chemin optimal vis-à-vis d'un critère afin d'atteindre un état objectif depuis l'état initial est un problème classique de la théorie des graphes (Lacomme et al., 2003).

Parmi les nombreuses solutions existantes et moyennant une hypothèse sur le poids des arcs qui doit être positif ou nul, il est possible d'utiliser l'algorithme de Dijkstra (Dijkstra, 1959), de faible complexité, $O(S)$ avec S le nombre de sommets (cf. chapitre 7), qui fournit un chemin optimal pour un critère à partir d'un sommet source, à chacun des autres sommets du graphe. L'utilisation de l'algorithme de Dijkstra nécessite d'abord de construire à partir du système de transitions d'états D_1 un graphe G_1.

Les sommets du graphe correspondent aux états du système de transitions et les arcs aux transitions entre états. L'état initial $q_{1,0}$ devient le sommet $s_{1,0}$ et les sommets objectifs $s_{1,Ob}$ correspondent aux états objectifs $q_{1,Ob}$. Le poids des arcs est défini à partir de la caractéristique des opérations nécessaires à l'évaluation du critère d'optimisation. Quand la caractéristique considérée est la durée, le poids de l'arc est censé représenter le temps nécessaire pour passer d'un état à un autre. Mais, les opérations, d'action ou d'information, et les évolutions requises ne sont pas systématiquement lancées au plus tôt. Ainsi, le temps nécessaire pour passer d'un état à un autre suite à une opération ou une évolution requise est égal à la somme du délai de déclenchement et de la durée de l'action. La définition du poids de l'arc repose pour ces opérations sur deux hypothèses liées directement au délai de déclenchement présenté dans la Figure 5.14 page 90.

Pour une opération, l'événement de début est contrôlé par le module coordination. Par conséquent, le délai de déclenchement d'une opération est fixé par le module de coordination. N'ayant aucune raison d'attendre et d'autant plus quand le temps d'exécution de la loi de commande est à minimiser, toutes les opérations seront lancées au plus tôt. Ainsi, le poids d'un arc correspondant à une opération sera égal à la durée de cette opération.

En revanche, une évolution requise peut être lancée au plus tôt comme ne jamais l'être. Mais en se focalisant sur un seul produit, ces évolutions apparaîtront uniquement au début et à la fin de la séquence, correspondant respectivement à l'arrivée et à l'évacuation du produit. Les évolutions requises ne seront alors pas considérées lors de la recherche du plus court chemin. De manière à ne pas devoir modifier le graphe $G1$, le poids d'un arc correspondant à une évolution requise sera nul afin de ne pas avoir d'impact sur le poids des chemins.

A partir du graphe $G1$ et du résultat de l'algorithme de Dijkstra, il suffit ensuite de comparer le poids des chemins entre le sommet $s_{1,0}$ et les sommets $s_{1,Ob}$, et de retenir le chemin de poids minimum. La proposition ci-dessus soulève la question du choix du chemin quand le poids de plusieurs chemins est minimal. Cette question se pose d'une part entre les chemins menant à un même sommet objectif et d'autre part entre les chemins menant à différents sommets objectifs. En effet, pour l'algorithme de Dijkstra, une décision arbitraire est prise lorsque plusieurs chemins sont de poids minimum. Pour ne pas laisser le hasard décider par manque de critères, il est nécessaire de recourir à des critères décisionnels supplémentaires comme le propose la section suivante.

3.3.2 Problème multi-critères

Deux solutions sont envisageables afin de considérer plusieurs critères. La première est de proposer à un expert les chemins de poids minimum afin qu'il décide. La seconde est de prendre en compte explicitement d'autres critères et de développer un algorithme multi-critères. Cette problématique n'étant pas le cœur de l'approche proposée, nous donnons uniquement ici des indications sur les avantages et les inconvénients de ces deux solutions.

Un expert décide du chemin à sélectionner, afin depuis l'état initial, d'atteindre un état objectif. Le choix d'un chemin induit également le choix d'un sommet objectif. L'avantage est de s'appuyer sur les compétences d'un expert qui dispose bien souvent de critères très difficiles à évaluer numériquement. Mais en contre-partie, la durée nécessaire à l'intervention de l'expert risque de dégrader considérablement les performances de l'algorithme d'un point de vue temporel. Cette solution est par conséquent à privilégier uniquement en phase de configuration.

Un algorithme multi-critères qui peut par exemple considérer des critères classés par ordre d'importance. Le poids des arcs du premier graphe correspond à la caractéristique des opérations nécessaire à l'évaluation du premier critère. Suite à l'application de l'algorithme de Dijkstra pour ce premier graphe, il sera appliqué une seconde fois à un deuxième graphe composé de l'ensemble des chemins de poids minimal résultant de la première application de Dijkstra. Le poids des arcs de ce deuxième graphe correspond à la caractéristique des opérations nécessaire à l'évaluation du second critère, et ainsi de suite jusqu'à aboutir à un chemin unique.

Cette solution rend l'algorithme de recherche du chemin optimal plus complexe et nécessite pour certains critères d'enrichir le modèle. Néanmoins, sans enrichissement du modèle, il est possible de définir d'autres critères, comme par exemple de privilégier les états objectifs depuis lesquels il n'y a pas d'évolutions induites ou requises applicables, ou bien encore de minimiser le nombre d'actions à exécuter. Enfin, pour une même caractéristique des opérations, plusieurs critères sont envisageables. Notamment à partir de la durée des opérations, le temps qu'une chaîne fonctionnelle passe dans un état conduisant à une usure importante peut être minimisé, comme le temps où un dispositif d'aspiration par un système Venturi est activé. Tout ceci nécessite de pouvoir offrir à l'expert le moyen de définir ses propres critères en fonction du système de production et les mécanismes associés pour leur prise en compte par l'algorithme de synthèse.

Le chemin optimal de comportements de transformation, noté Ch_1, dans le système de transition d'états D_1 d'un état initial $q_{1,0}$ afin d'atteindre un objectif Ob_1 est obtenu par la correspondance entre les sommets du graphe G_1 et les états du système de transition d'états D_1. La séquence de comportements de transformation à imposer au système contrôlé est alors donnée par ce chemin (cf. Figure 8.4).

Lors de cette première étape limitée à la vision de l'état physique d'un produit, seules des opérations avec des comportements de transformation ont été définies. Ainsi, la seconde étape, présentée dans la section suivante, vise à considérer en plus de l'état physique du produit sa position dans le système de production.

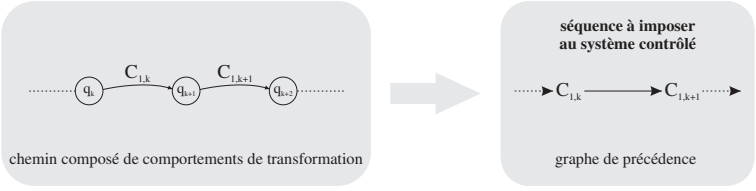


FIG. 8.4 – Traduction du chemin optimal en graphe de précédence représentant la séquence de comportements à imposer au système contrôlé.

4 Phase A, étape 2 : Déplacement du produit

La considération des évolutions de la position du produit a pour but d'une part de satisfaire la demande en tenant compte de l'arrivée et du départ du produit, et d'autre part de satisfaire les conditions et les pré-contraintes sur la position du produit afin de pouvoir imposer les comportements de transformation spécifiés par le chemin construit précédemment, Ch_1.

De par les similitudes importantes avec l'étape 1 présentée précédemment, la présentation de l'étape 2 dans cette section se focalise uniquement sur les points particuliers en donnant néanmoins les notations adoptées.

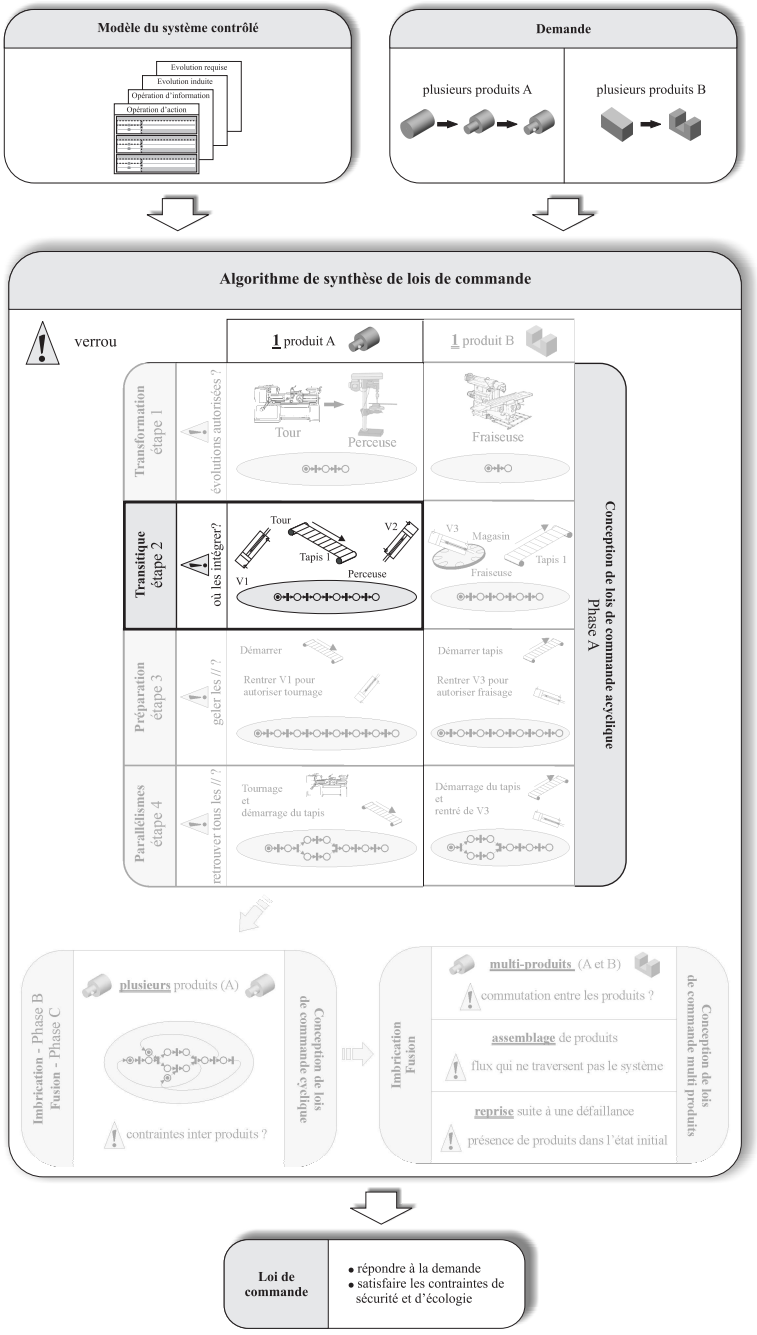


FIG. 8.5 – Étape 2 de conception d'une loi de commande acyclique.

4.1 Principe

A l'inverse de la première étape qui construit une seule séquence, l'étape 2 va générer autant de séquences, modifiant la position du produit, que nécessaire afin de satisfaire la demande et permettre la transformation du produit (cf. Figure 8.6). L'intégration de ces séquences au chemin, généré à l'étape précédente, conduit à la génération d'un chemin, dit de transformation et de transitique, noté Ch_{12}.

Le principe de construction d'un tel chemin fait apparaître une différence fondamentale dans la définition de l'état initial et de l'objectif. En effet à l'inverse de l'étape 1 pour laquelle l'état initial $q_{1,0}$ et l'objectif Ob_1 sont issus directement de la demande, l'état initial $q_{2,i,0}$ et l'objectif $Ob_{2,i}$ pour un problème de transitique ne sont pas donnés par la demande. Ainsi, la particularité de l'étape 2 est la nécessité de déterminer l'état initial $q_{2,i,0}$ et l'objectif $Ob_{2,i}$, pour définir ensuite les évolutions de la position du produit.

Au delà, de cette particularité sur la connaissance de l'état initial et de l'objectif, la recherche d'un chemin représentant les évolutions de la position du produit, noté $Ch_{2,i}$, diffère de la génération d'un chemin de transformation, Ch_1, sur les quatre points suivants :

- seuls les comportements de transitique des opérations sont considérés ;
- seule la position du produit évolue. Ainsi, un état q_2 d'un chemin $Ch_{2,i}$ appartient à l'ensemble Q_2 défini par $Q_2 = \prod_{x \in VE_1 \cup VE_2} V_x$ avec V_x l'ensemble des valeurs de la variable d'état x. Un état $q_2 \in Q_2$ est défini par $q_2 = \{(x = c) | x \in VE_1 \cup VE_2\}$, ou $c \in V_x$;
- le chemin de transformation et de transitique, Ch_{12}, est construit progressivement à chaque nouvelle séquence de transitique construite ;
- la complexité de l'espace d'états atteignables au sein duquel sont recherchées les évolutions de la position du produit est limitée par des conditions restrictives différentes de celles utilisées dans l'étape 1.

Comme nous l'avons précisé précédemment, la deuxième étape nécessite d'abord de déterminer l'état initial et l'objectif des chemins de transitiques à construire. Cette détermination étant basée sur le mécanisme de construction du chemin de transformation et de transitique, Ch_{12}, nous présentons d'abord ce mécanisme avant de nous focaliser sur la génération d'un chemin de transitique pour un état initial et un objectif donnés.

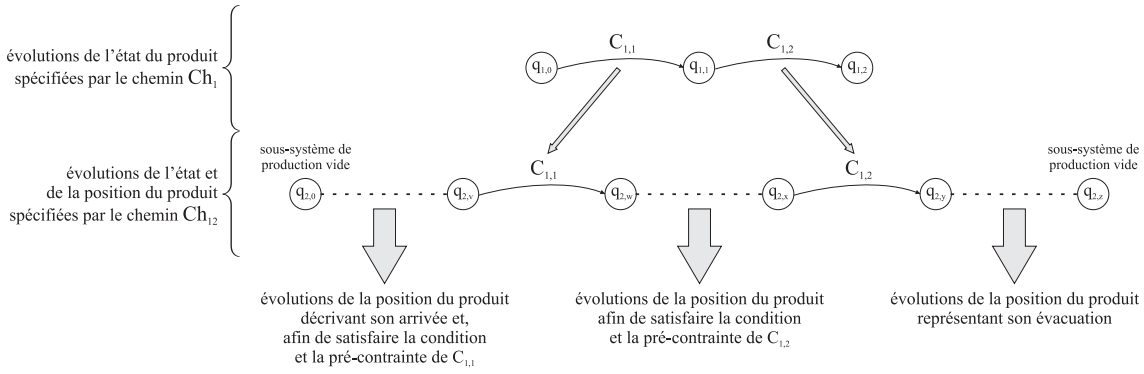


FIG. 8.6 – Principe de l'étape 2.

4.2 Séquence d'opérations de transformation et de transitique

Afin de satisfaire d'une part la partie de la demande imposant l'absence de produit dans l'état initial et l'état final, et d'autre part les conditions et les pré-contraintes sur la position du produit pour les transformations du produit spécifiées lors de l'étape 1, les évolutions de l'état et de la position du produit, représentées par un chemin Ch_{12} (cf. Figure 8.7) sont le résultat de l'intégration d'évolutions de la position du produit entre chaque évolution de son état.

La Figure 8.7 expose ce principe de construction progressive du chemin Ch_{12} basé sur l'algorithme 4 qui présente uniquement la partie centrale de l'algorithme.

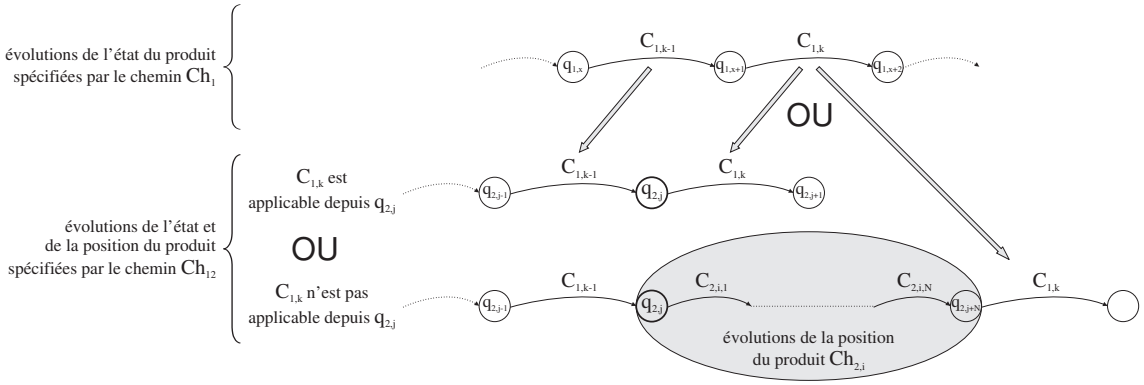


FIG. 8.7 – Ajout d'évolutions de la position du produit entre deux évolutions de son état physique.

Notons, avant de poursuivre l'étude de l'étape 2, que l'algorithme 4 est incomplet. En effet, il est nécessaire d'amener un produit depuis l'état initial $q_{2,1,0}$, défini par l'absence de produits dans le système de production, afin de positionner ce dernier pour la première transformation. La particularité de ces déplacements est l'état initial $q_{2,1,0}$ qui correspond à l'état initial $q_{2,0}$ imposé par la demande. De même, il s'avère nécessaire d'évacuer le produit suite à la dernière transformation. La particularité de l'objectif visé par ces déplacements est d'être imposé par la demande.

L'algorithme 2 page 124qui spécifie les évolutions de l'état et de la position du produit fait appel à la fonction *CheminOptimal* qui spécifie les évolutions de la position du produit requises entre deux transformations de son état. Ainsi, la suite de la présentation de l'étape 2 se focalise sur la recherche d'un chemin optimal spécifiant les évolutions de la position du produit. La recherche d'un chemin optimal $Ch_{2,i}$ est identique à la recherche du plus court chemin effectuée à l'étape 1 au §3.3 page 125. Pour cette raison, nous détaillerons uniquement la construction de l'espace d'états sur lequel est basé la recherche du plus court chemin.

4.3 Espace restreint d'états atteignables

A partir du modèle du système contrôlé et de son environnement, pour un état initial $q_{2,i,0}$ et un objectif $Ob_{2,i}$ défini par une ou plusieurs propositions sur la position du produit, la recherche d'un chemin optimal de l'état initial $q_{2,i,0} \in Q_2$ à un état objectif $q_2 \in Q_2$ tel que $Ob_{2,i}$ soit satisfait nécessite de construire l'espace restreint d'états atteignables $D_{2,i} = (Q_{A2,i}, C_2, \delta_{2,i}, q_{2,i,0}, Ob_{2,i})$ avec $Q_{A2,i} \subseteq Q_2$ l'ensemble restreint des états atteignables depuis $q_{2,i,0}$ pour l'objectif $Ob_{2,i}$.

Le principe de construction de l'espace restreint des états atteignables repose sur quatre points :

- déterminer l'ensemble des comportements existants qui affectent la position du produit depuis un état q_2. Cet ensemble sera le résultat de la fonction $C_2(q_2)$;
- construire l'espace des états atteignables par ces comportements depuis l'état initial $q_{2,i,0}$;
- limiter la complexité de l'espace d'états par une condition restrictive sur le nombre de produits ;
- limiter la complexité de l'espace d'états par une condition restrictive fonction de l'objectif.

Fonction Construction($q_{2,0}$,Ob_2,Ch_1) : Ch_{12}
  ...
  K=nb de comportements de transformation spécifiés par Ch_1
  k=2
  Tant que $(k \leq K)$ **faire**
    $q_{2,j}$=dernier état actuel de Ch_{12}
    Si (les conditions et les pré-contraintes de l'opération $O_{1,k}$ sont satisfaites pour $q_{2,j}$) **Alors**
      $q_{2,j+1} = \delta_2(q_{2,j}, O_{1,k})$
      $Ch_{12} = Ch_{12} \cup q_{2,j+1}$
    Sinon
      $q_{2,i,0} = q_{2,j}$
      $Ob_{2,i}$={conditions et pré-contraintes à satisfaire de $O_{1,k}$}
      $Ch_{2,i}$=**CheminOptimal**($q_{2,i,0}$,$Ob_{2,i}$), (présenté dans la suite de cette section)
      $Ch_{12} = Ch_{12} \cup Ch_{2,i}$
      $q_{2,j+N+1} = \delta_2(q_{2,j+N}, O_{1,k})$ avec N=nb de comportements de transitique dans $Ch_{2,i}$
      $Ch_{12} = Ch_{12} \cup q_{2,j+N+1}$
      i=i+1
    Fin Si
    k=k+1
  Fin Tq
  ...
  Retourner Ch_{12} ;
Fin

ALG. 4: Ajout d'évolutions de la position du produit entre deux évolutions de son état physique.

4.3.1 Opérations affectant la position du produit

La construction de l'espace d'états atteignables limité à des évolutions de la position des produits nécessite de déterminer pour un état toutes les opérations dont l'effet de leur comportement est de modifier la position d'un produit (cf. Figure 8.8). Ainsi, nous définissons une fonction $C_2(q_2)$ qui renvoie pour un état $q_2 \in Q_2$ l'ensemble de ces comportements ($C_2(q_2) = \{c_{2,i}\}$).

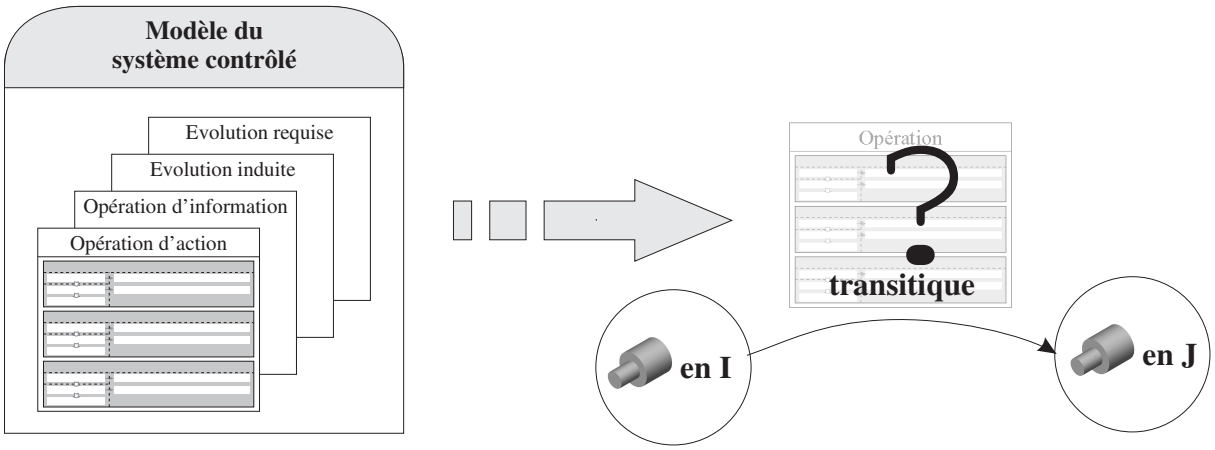


FIG. 8.8 – Extraction du modèle des opérations avec un comportement qui déplace le produit.

Le comportement d'une opération modifiera uniquement la position du produit, s'il est défini par au moins une évolution associée du flux de produits qui modifie la position d'un produit, et si aucune de ces évolutions associées du flux de produit ne modifie l'état physique d'un produit. Ainsi, toute opération qui vérifie le test suivant pour un état $q_2 \in Q_2$ modifie la position du produit :

Il existe au moins une évolution du flux de produits, pour laquelle :
- les conditions sur l'état physique et la position du flux de produits sont vraies ;
- les pré-contraintes sur l'état physique et la position du flux de produits sont satisfaites ;
- l'effet transitoire et l'effet final ne modifient pas l'état physique des produits.

Il n'existe pas d'évolution du flux de produits, pour laquelle :
- les conditions sur l'état physique et la position du flux de produits sont vraies ;
- les pré-contraintes sur l'état physique et la position du flux de produits ne sont pas satisfaites ;
- l'effet transitoire et l'effet final ne modifient pas l'état physique du flux de produits.

Il n'existe pas d'évolution du flux de produits, pour laquelle :
- les conditions sur l'état physique et la position du flux de produits sont vraies ;
- les pré-contraintes sur l'état physique et la position du flux de produits sont satisfaites ;
- l'effet transitoire et l'effet final modifient l'état physique du flux de produits.

Pour l'évolution de la chaîne fonctionnelle, quand il y en a une :
- les pré-contraintes sur l'état physique et la position du flux de produits sont satisfaites.

En vérifiant pour chacune des opérations du modèle du système contrôlé (cf. partie II), si le test ci-dessus est vrai ou faux, la fonction $C_2(q_2)$ détermine l'ensemble des comportements d'opérations qui modifient la position du flux de produits. Disposant maintenant d'une fonction donnant l'ensemble des comportements de transitique possibles depuis un état, nous sommes maintenant en mesure de construire l'espace d'états atteignables depuis un état initial $q_{2,i,0}$.

4.3.2 Espace d'états atteignables

La construction de l'espace d'états atteignables, noté $D_{2,i}$, par des évolutions de la position des produits est identique à la génération de l'espace d'états atteignables par des transformations, D_1, présenté au § 3.2.2 page 122. L'unique différence entre les deux algorithmes est l'appel à la fonction $C_2(q_2)$ à la place de la fonction $C_1(q_1)$. Ainsi, nous ne détaillerons pas davantage la génération de cet espace d'états.

Cependant, la complexité de cet espace d'états atteignables, noté $D_{2,i} = (Q_{A2,i}, C_2, \delta_{2,i}, q_{2,i,0})$, est limité par deux conditions restrictives qui lui sont propres. La première présentée dans la section suivante limite le nombre de produits présents dans le sous-système de production considéré. La seconde qui est similaire à celle présentée lors de l'étape 1 §3.2.3 est liée à l'objectif à atteindre.

4.3.3 Condition restrictive sur le nombre de produits

La demande considérée dans ce chapitre est limitée à la fabrication d'un seul produit. Ce produit entre dans le système de production puis en ressort. Ainsi, toute évolution de l'état d'un autre produit sans intérêt pour la fabrication du produit demandé sera une perte non seulement de temps mais aussi d'argent. Pour un module de coordination considéré, l'environnement est responsable des produits qui entrent dans le système de production ; nous interdirons donc l'arrivée d'un nombre quelconque de produits. Ainsi, le nombre de produits présents dans un état sera limité. Cette condition consiste à ne pas ajouter à l'ensemble des états à visiter Q_v un nouvel état atteint pour lequel le nombre de produits est supérieur à une limite fixée que nous allons maintenant étudier.

Fonction EtatsAtteignables($q_{2,i,0}$,$Ob_{2,i}$) : **système de transition d'états**
  Répéter
    $Q_v = Q_{nv}$
    $Q_{nv} = \{\emptyset\}$
    $l_p = l_p + 1$, avec l_p le nombre limite de pièces
    ...
    [Construction de l'espace d'états atteignables pour l_p]
    ...
  jusqu'à ce que ($Q_{2,Ob} = \{\emptyset\}$ ET $Q_{nv} \neq \{\emptyset\}$)
  Retourner $D_{2,i} = (Q_{A2,i}, C_2, \delta, q_{2,i,0}, Q_{2,Ob})$;
Fin

ALG. 5: Condition restrictive sur le nombre de produits présents.

La condition est de limiter le nombre de produits et non pas d'imposer l'entrée d'un unique produit afin de satisfaire la demande. En effet, certains systèmes nécessitent la présence de plus d'un produit afin d'avoir un effet sur le premier produit entré. Considérons l'exemple d'un vérin dont la sortie peut au maximum à partir de A déplacer un produit en B (cf. Figure 8.9). Alors, la seule opération pouvant avoir un effet sur le produit en B nécessite de positionner d'abord

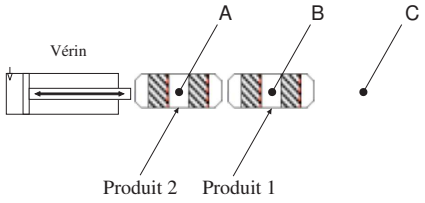


FIG. 8.9 – Système nécessitant l'introduction de plus d'un produit pour en fabriquer un.

un produit en A. Ainsi, le vérin doit rentrer et attendre l'arrivée d'un nouveau produit. Cet exemple montre, que même pour la demande considérée dans ce chapitre, le nombre de produits présents dans le système ne peut pas toujours être limité à un.

Le problème est alors de fixer cette limite. En faisant l'hypothèse que le chemin optimal afin d'atteindre un objectif passe toujours par les états contenant le plus petit nombre de produits, le principe est alors de fixer initialement la limite à un produit puis de l'incrémenter de un si l'objectif ne peut pas être atteint. Quand l'objectif est atteint, la limite l_p servira à informer l'environnement du nombre de produits qu'il doit fournir. La condition conduit au nouvel algorithme 5 de construction de l'espace d'états atteignables par des évolutions de la position du produit. Il faut noter le nouvel ensemble Q_{nv} d'états à ne pas visiter. Ces états sont ceux pour lesquels le nombre de produits présents est supérieur à la limite l_p. L'incrémentation de la limite l_p du nombre de produits implique alors de visiter ces états ; pour cela ils sont transférés dans la liste des états à visiter.

En plus de cette condition restrictive sur l'espace d'états atteignables, une seconde condition, fonction de l'objectif, est présentée dans la section suivante.

4.3.4 Condition restrictive fonction de l'objectif

De façon similaire à la condition restrictive vue à l'étape 1 de transformation, seuls les comportements des évolutions requises et induites qui modifient la position du produit doivent être renvoyés par la fonction $C_2(q_2)$. Les comportements dont les effets conduiraient à ne plus respecter l'objectif $Ob_{2,i}$ doivent être interdits. Quant aux autres conduisant dans un état satisfaisant toujours l'objectif, cet état sera ajouté à l'ensemble des états objectifs $Q_{2,i,Ob}$ et à l'ensemble des états à visiter Q_v. L'application de cette condition restrictive conduit à l'espace restreint d'états atteignables noté $D_{2,i} = (Q_{A2,i}, C_2, \delta_{2,i}, q_{2,i,0}, Ob_{2,i})$.

Au sein de cet espace d'états, un chemin optimal est recherché depuis l'état initial $q_{2,i,0}$ vers un état satisfaisant $Ob_{2,i}$. Et comme nous l'avons précisé précédemment, la recherche du plus court chemin reste basée sur les mêmes techniques que celles présentées lors de l'étape 1 au §3.3. Pour cette raison, nous ne nous attardons pas d'avantage sur ce point. Rappelons que l'étape 2 génère un chemin qui spécifie les évolutions de l'état et de la position du produit afin de répondre à la demande. Ce chemin est noté Ch_{12}. Lors de cette deuxième étape, l'état des chaînes fonctionnelles n'a toujours pas été considéré ; pourtant cet état contraint l'exécution des opérations dont le comportement aboutit aux évolutions du produit spécifiées par l'étape 2.

Ainsi, l'étape 3 se propose, face à ce problème, d'introduire des opérations dont le comportement va modifier uniquement l'état des chaînes fonctionnelles.

5 Phase A, étape 3 : Préparation des chaînes fonctionnelles

La troisième étape vise à préparer les chaînes fonctionnelles afin de les placer dans un état qui satisfait d'une part la demande, et d'autre part les conditions et les pré-contraintes des comportements spécifiés pour l'évolution de l'état et de la position du produit.

Cette étape étant similaire à la précédente, nos propos se focaliseront essentiellement sur les différences avec cette dernière.

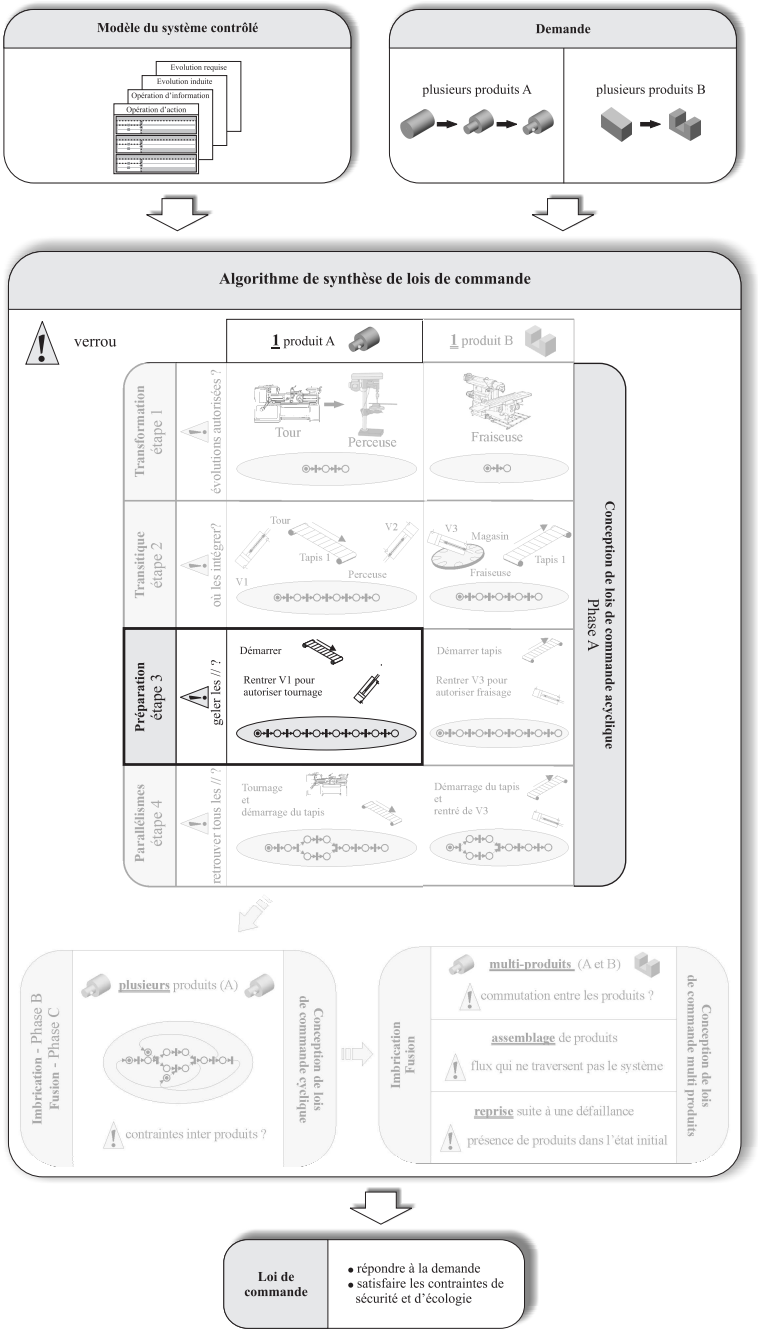


FIG. 8.10 – Étape 3 de conception d'une loi de commande acyclique.

5.1 Principe

L'étape 3 va construire des chemins, notés $Ch_{3,i}$, représentant les évolutions des chaînes fonctionnelles en dehors de celles qui agissent sur le produit. L'ajout des évolutions de l'état des chaînes fonctionnelles aux évolutions de l'état et de la position du produit aboutit à un nouveau chemin, dit complet et noté Ch_{123}, spécifiant toutes les évolutions, aussi bien des chaînes fonctionnelles que du produit, afin de fabriquer le produit. Le principe général de l'étape 3 qui est similaire à celui de l'étape 2, est présenté dans la Figure 8.11.

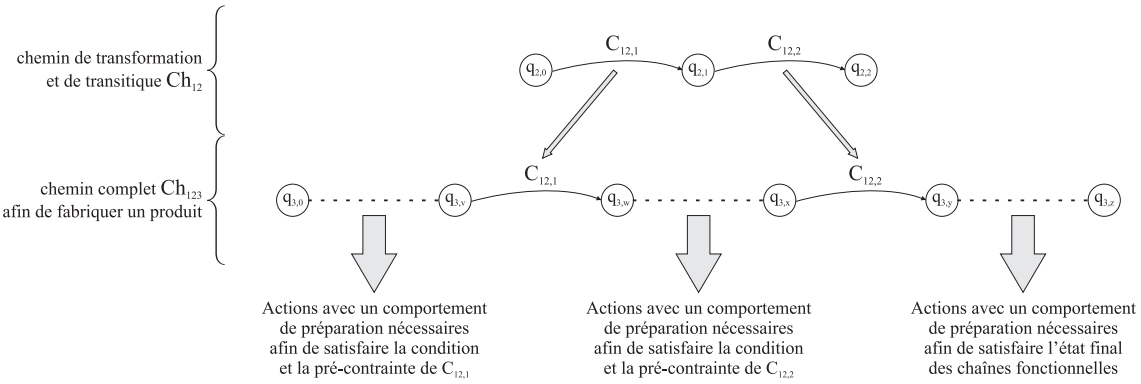


FIG. 8.11 – Principe de l'étape 3 de préparation des chaînes fonctionnelles.

L'algorithme de génération du chemin Ch_{123} qui est identique sur le principe à celui du chemin Ch_{12} n'est pas détaillé ici. Cet algorithme fait appel à une fonction *CheminOptimal* qui construit cette fois-ci un chemin spécifiant exclusivement des évolutions de l'état des chaînes fonctionnelles.

La construction d'un tel chemin, noté $Ch_{3,i}$, afin de préparer les chaînes fonctionnelles diffère légèrement de celle d'un chemin $Ch_{2,i}$ pour l'évolution de la position du produit :

- les comportements des opérations affectant uniquement l'état des chaînes fonctionnelles sont considérés ;
- ainsi, un état q_3 d'un chemin $Ch_{3,i}$ appartient à l'ensemble $Q_3 = \prod_{x \in VE_1 \cup VE_2 \cup VE_3} V_x$ avec V_x l'ensemble des valeurs de la variable d'état x. Un état $q_3 \in Q_3$ est défini par $q_3 = \{(x = c) | x \in VE_1 \cup VE_2 \cup VE_2\}$, ou $c \in V_x$;
- des conditions restrictives propres à cette troisième étape limitent la complexité de l'espace d'états atteignables.

Au vu des points énoncés ci-dessus, la recherche du plus court chemin est réalisée de la même façon que lors des étapes 1 et 2. En revanche, l'espace d'états au sein duquel est recherché ce chemin est différent. Ainsi, la section suivante se focalise sur la construction d'un tel espace d'états basé sur les évolutions des chaînes fonctionnelles.

5.2 Espace restreint d'états atteignables

Afin de trouver le plus court chemin $Ch_{3,i}$ représentant la préparation des chaînes fonctionnelles, un espace restreint d'états atteignables depuis un état initial et pour un objectif est construit. Il est noté $D_{3,i} = (Q_{A3,i}, C_3, \delta_{3,i}, q_{3,i,0}, Ob_{3,i})$.

Le principe de construction de cet espace restreint repose sur trois points :
- une fonction $C_3(q_3)$ qui donne les comportements d'opérations dont l'unique effet est de modifier l'état d'une chaîne fonctionnelle ;
- la construction de l'espace des états atteignables depuis l'état initial $q_{3,i,0}$ à partir de la fonction $C_3(q_3)$;
- une unique condition restrictive fonction de l'objectif limite la complexité de cet espace d'états.

5.2.1 Opérations de préparation des chaînes fonctionnelles

La construction de l'espace d'états atteignables uniquement par des évolutions des chaînes fonctionnelles nécessite de déterminer pour un état tous les comportements d'opérations dont l'unique effet est de modifier l'état d'une chaîne fonctionnelle. Ainsi, nous définissons la fonction $C_3(q_3)$ qui renvoie pour un état $q_3 \in Q_3$ l'ensemble de ses comportements ($C_3(q_3) = \{c_{3,i}\}$).

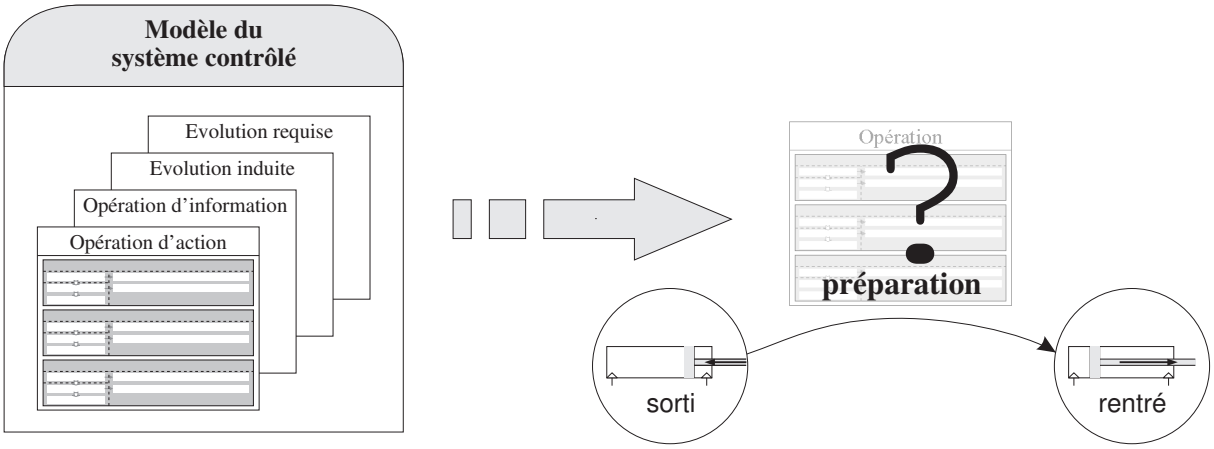


FIG. 8.12 – Extraction du modèle des opérations dont le comportement affecte uniquement l'état d'une chaîne fonctionnelle.

Toute opération qui vérifie le test suivant pour un état $q_3 \in Q_3$ a un comportement qui affecte uniquement l'état d'une chaîne fonctionnelle :

Il n'existe pas d'évolution du flux de produits, pour laquelle la condition est vérifiée ;

Pour l'évolution de la chaîne fonctionnelle :
- la condition sur l'état de la chaîne fonctionnelle est vraie ;
- la pré-contrainte, sur l'état physique des produits, la position du flux de produits et l'état des chaînes fonctionnelles, est satisfaite.

La fonction $C_3(q_3)$ détermine l'ensemble des comportements qui visent à préparer les chaînes fonctionnelles. Cette fonction est utilisée dans la section suivante afin de générer l'espace d'états atteignables uniquement par des évolutions des chaînes fonctionnelles.

5.2.2 Espace d'états atteignables

La construction de l'espace d'états atteignables, noté $D_{3,i}$, limité à des évolutions des chaînes fonctionnelles et depuis un état initial, $q_{3,i,0}$, est identique à la génération de l'espace d'états

atteignables présenté au § 3.2.2 page 122. L'unique différence entre les deux algorithmes est l'appel à la fonction $C_3(q_3)$ à la place de $C_1(q_1)$. En revanche, il ne s'agit plus de se focaliser sur un produit comme dans les étapes 1 et 2 mais sur l'ensemble des chaînes fonctionnelles en interdisant toutes les évolutions du flux de produits.

Une condition restrictive, propre à la résolution du problème de préparation, est présentée dans la section suivante afin de limiter la complexité de l'espace d'états atteignables $D_{3,i} = (Q_{A3,i}, C_3, \delta_{3,i}, q_{3,i,0})$.

5.2.3 Condition restrictive fonction de l'objectif

La prise en compte de l'objectif fixant l'état désiré des chaînes fonctionnelles permet de limiter la complexité de l'espace d'états atteignables de deux façons différentes :

Seuls les comportements des évolutions requises qui modifient l'état des chaînes fonctionnelles de l'environnement sont renvoyés par la fonction $C_3(q_3)$ quand un état objectif est visité (cf. condition restrictive du § 3.2.3 page 124). Le comportement d'une évolution requise qui conduirait à ne plus respecter l'objectif $Ob_{3,i}$ doit être interdit. Quant aux autres, conduisant dans un état satisfaisant toujours l'objectif, ces états sont alors ajoutés à l'ensemble des états objectifs $Q_{3,i,Ob}$ et à l'ensemble des états à visiter Q_v.

Orienter la construction de l'espace d'états qui dans un premier temps va se focaliser sur les comportements dont l'effet est de modifier les variables d'états qui apparaissent dans l'objectif. Puis, si l'objectif ne peut pas être atteint, seuls les comportements des opérations basés sur les chaînes fonctionnelles dont une variable d'état apparaît dans l'objectif sont considérés. Enfin, si l'objectif n'est toujours pas atteint, tous les comportements affectant uniquement l'état d'une chaîne fonctionnelle seront considérés.

Au sein de cet espace restreint d'états atteignables, un chemin optimal est à trouver depuis l'état initial vers un état satisfaisant l'objectif. La recherche du plus court chemin est identique à celle présentée au §3.3 page 125. Pour cette raison, nous ne la détaillerons pas et passerons directement à l'étude de l'étape 4.

6 Phase A, étape 4 : Ré-introduction des parallélismes

Le résultat des étapes 1,2 et 3 est un chemin qui spécifie les évolutions des chaînes fonctionnelles et du produit afin de le fabriquer. En l'absence d'aléas, ces évolutions sont conditionnées par l'exécution d'opérations spécifiées, sur les transitions du chemin, par leur comportement.

Il reste néanmoins une partie de la demande à laquelle la réponse apportée par le chemin complet Ch_{123} trouvé n'est pas toujours optimale. En effet, le temps de cycle de la loi de commande correspondant à la séquence d'opérations spécifiées par le chemin complet Ch_{123} peut être réduit en introduisant des parallélismes d'exécution entre ces opérations.

Nous avons jusqu'à maintenant utilisé un formalisme basé sur des états et des transitions entre états. Ce formalisme était particulièrement adapté puisque les parallélismes d'exécution étaient gelés. La ré-introduction de ces parallélismes dans cette quatrième étape amène à reconsidérer le formalisme utilisé jusqu'à maintenant. En effet, une représentation d'états est très limitée dans l'expression des parallélismes comparée, par exemple, à celle des réseaux de Petri.

Néanmoins, comme nous le verrons dans cette section et dans les deux chapitres suivants, il s'avère nécessaire pour l'algorithme de conserver la connaissance de l'état du système contrôlé depuis lequel les opérations avec leur comportement sont exécutées. Ainsi pour les besoins de l'algorithme de synthèse uniquement, nous devons conserver un formalisme basé sur une représentation d'états tout en représentant par ailleurs des parallélismes. Le formalisme recherché étant propre à l'algorithme de synthèse et donc jamais visualisé par l'expert, il privilégiera l'accès à l'information pour l'algorithme. Ainsi, ce formalisme propre à l'algorithme sera basé sur une représentation d'états couplée à une représentation particulière de contraintes de précédence entre les comportements des opérations.

Afin de présenter ce formalisme, considérons une séquence de trois opérations représentées par leur comportement respectif (cf. Figure 8.14). Deux contraintes de précédence sont imposées

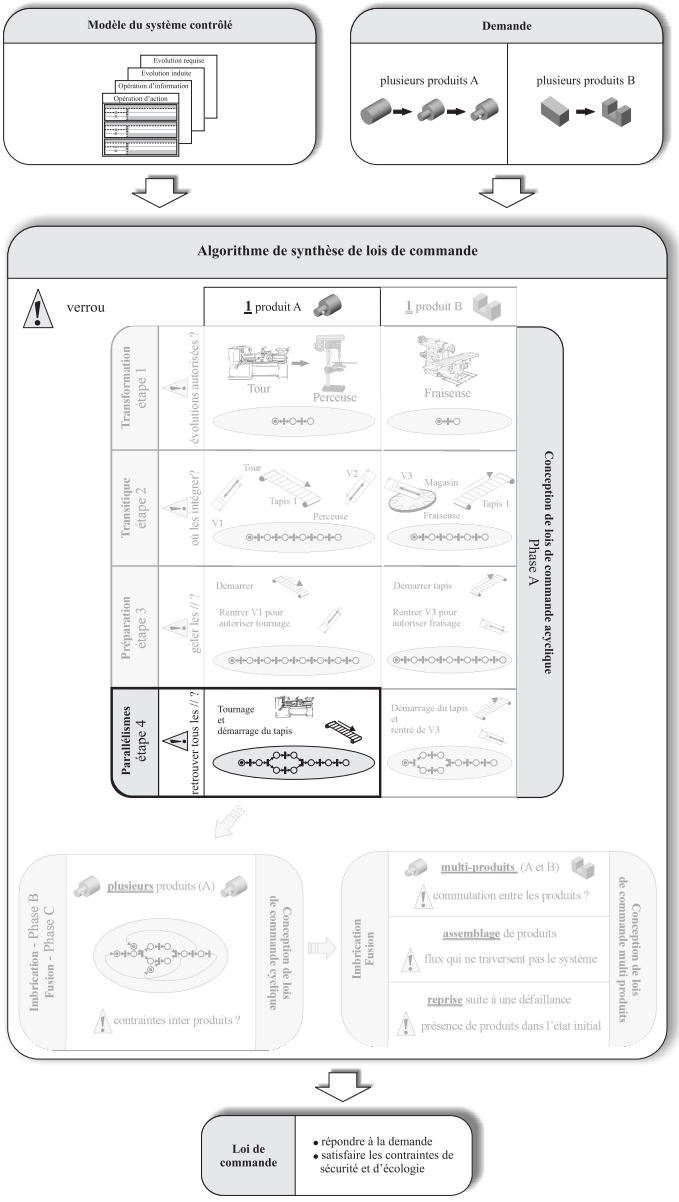


FIG. 8.13 – Étape 4 de conception d'une loi de commande acyclique.

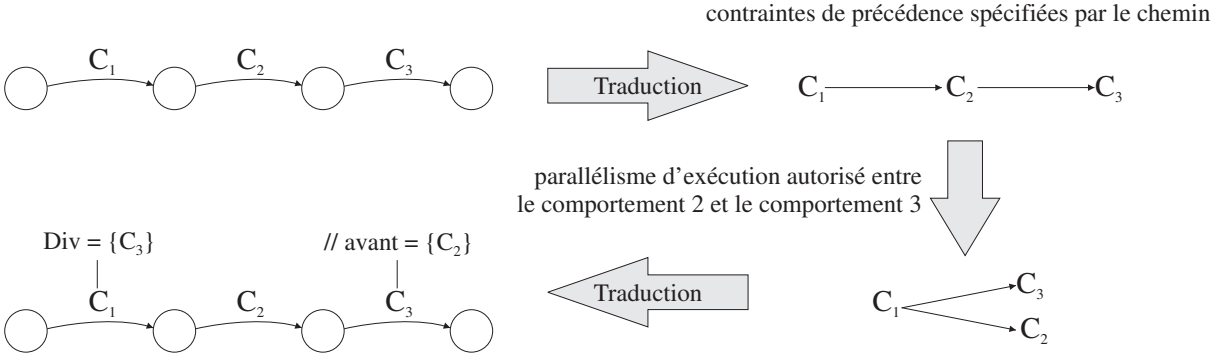


FIG. 8.14 – Représentation des parallélismes en conservant un formalisme basé sur les états du système contrôlé.

par le chemin qui spécifie cette séquence : le comportement 1 avant le comportement 2, et le comportement 2 avant le comportement 3. Considérons maintenant que les comportements 2 et 3 puissent être exécutés en parallèle. Les contraintes de précédence sont alors : le comportement 2 succède toujours le comportement 1, et le comportement 3 peu immédiatement suivre le comportement 1.

Le parallélisme d'exécution entre les comportements 2 et 3 supprime donc la contrainte de précédence entre les comportements 2 et 3, et ajoute une contrainte de précédence entre les comportements 1 et 3. Cette dernière contrainte de précédence conduit à une divergence en ET après le comportement 1, puisque les comportements 2 et 3 seront exécutés. Afin de représenter cette divergence, une liste *divergence* associée au comportement 1 est crée, notée *div*. Le chemin, qui spécifie la séquence, représentant déjà la contrainte entre le comportement 1 et le comportement 2, seul le comportement 3 est ajouté à cette liste *divergence*. Après avoir représenté la nouvelle contrainte de précédence, il est nécessaire de supprimer celle entre les comportements 2 et 3. Pour cela, nous associons au comportement 3 une liste des parallélismes d'exécution avec les comportements placés avant celui-ci dans la séquence initiale. Cette liste est notée // *avant*. Dans notre exemple, cette liste contiendra alors le comportement 2.

Basé sur le formalisme proposé, qui rappelons le, est propre à l'algorithme de synthèse, nous présentons tout d'abord la recherche des parallélismes d'exécution entre un comportement et ceux le précédent. Puis suite à la présentation de la notion de parallélismes induits, nous présentons la démarche globale d'introduction des parallélismes d'exécution.

6.1 Parallélismes entre un comportement et ceux le précédents

L'exemple de parallélismes d'exécution de l'exemple de présentation du formalisme présenté précédemment se limite à considérer qu'il existe un parallélisme d'exécution entre un comportement et uniquement celui le précédent. Mais il faut aussi envisager que des parallélismes d'exécution avec d'autres comportements en amont sont autorisés. Ainsi, cette section propose de traiter cette proposition.

Considérons un chemin complet Ch_{123} composé de N transitions qui sont chacune étiquetée avec un comportement $(C_k, k \in [1, N])$ d'une opération. Ces comportements transforment le pro-

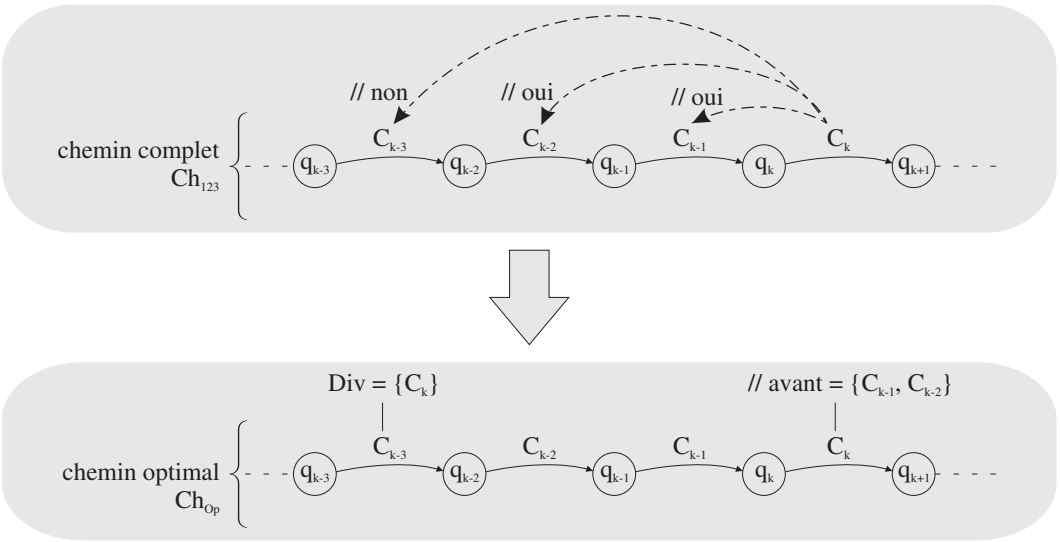


FIG. 8.15 – Principe de la recherche des parallélismes entre un comportement et les comportements en amont.

duit, modifient sa position, ou bien préparent les chaînes fonctionnelles. L'ajout au chemin Ch_{123} des informations relatives aux parallélismes d'exécution conduit à un chemin, dit optimal du point de vue du temps total d'exécution de la loi de commande correspondante. Il est noté Ch_{Op}.

Les deux listes (*// avant* et *div*) d'un comportement C_k d'une opération sont complétées de la façon suivante :

Liste *// avant*. Si le comportement C_k et le comportement C_{k-1} peuvent être exécutés en parallèle suite à la vérification de la propriété 4 page 92, alors le comportement C_{k-1} est ajouté à la liste des parallélismes autorisés avant C_k. Puis de même entre C_k et le comportement précédent C_{k-2}, jusqu'à trouver un comportement d'une opération dont le parallélisme d'exécution est interdit avec C_k. Le comportement C_k est alors ajouté à la liste *div* de ce dernier comportement. Cette règle ne s'applique pas quand le parallélisme d'exécution est impossible entre C_k et C_{k-1} car la contrainte de précédence entre ces deux comportements est déjà spécifiée par le chemin complet Ch_{123}.

Liste *div*. Par exemple, sur la Figure 8.15, nous supposons que les comportements C_k et C_{k-3} ne vérifient pas la propriété générique 4 page 92 sur le parallélisme d'exécution. Le comportement C_k est alors ajouté à la liste d'opérations à exécuter après C_{k-3}. Le chemin Ch_{Op} contient alors l'information sur l'exécution en parallèle des comportements C_{k-2} et C_k après le comportement C_{k-3} ; C_{k-2} puisqu'il succède C_{k-3}, et C_k puisqu'il appartient à la liste d'opérations à exécuter après C_{k-3}.

La recherche des parallélismes autorisés entre un comportement C_k et ceux en amont s'arrête quand la propriété générique 4 n'est pas vérifiée. En effet, un chemin complet Ch_{123} a été construit pour un produit, il n'est alors pas possible d'inverser l'ordre de deux comportements dont l'effet serait différent de celui spécifié par le chemin complet Ch_{123}.

Le cas où un comportement C_k est exécutable en parallèle de tous les comportements le précédant est un cas particulier. En effet, avec le principe énoncé ci-dessus, le comportement

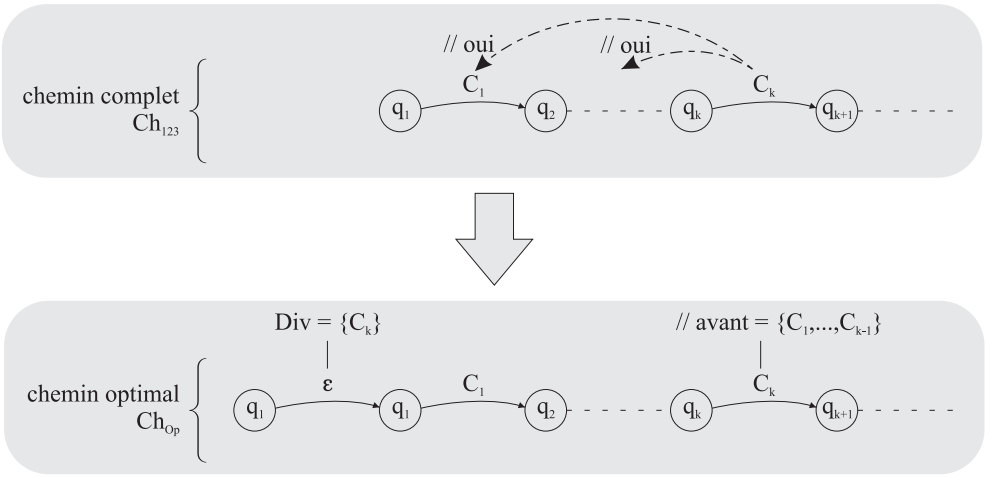


FIG. 8.16 – Cas particulier où l'exécution d'un comportement est autorisée en parallèle de tous les comportements le précédent.

C_k ne sera ajouté à aucune liste Div. Alors l'opération avec le comportement C_k ne sera jamais lancée. Il est nécessaire de créer un état précédent l'état initial du chemin complet Ch_{123}. Cet état est identique à l'état initial et le comportement associé à la transition est le comportement neutre ε, voir la Figure 8.16. Le comportement C_k est finalement ajouté à la liste Div des comportements à exécuter après ε.

Le principe énoncé ci-dessus se focalise uniquement sur les parallélismes autorisés entre un comportement C_k et tous ceux en amont. Afin d'optimiser le temps d'exécution de la loi de commande, il est nécessaire d'appliquer ce principe à tous les comportements. Mais par un phénomène de parallélismes induits présentés dans la section suivante, les parallélismes d'exécution sont à rechercher en parcourant le chemin complet Ch_{123} vers l'avant.

6.2 Parallélismes induits

Dans la section précédente, nous avons posé le principe de recherche des parallélismes d'exécution entre un comportement C_k et ceux en amont (C_i avec $i \in [1, k-1]$). Mais la liste des parallélismes autorisés avant chacun des comportements C_i (avec $i \in [1, k-1]$) était vide. Au risque de violer des contraintes de sécurité et d'écologie, il est nécessaire de s'intéresser aux éléments de cette liste quand elle n'est pas vide.

La Figure 8.17 représente un chemin Ch_{123} pour lequel les parallélismes autorisés ont été déterminés pour les comportements C_i (avec $i \in [1, k-1]$). Cette Figure décrit alors le processus de détermination des parallélismes d'exécution entre le comportement C_k et ceux le précédent.

Les comportements C_k et C_{k-1} peuvent être exécutés en parallèle. En revanche, le parallélisme d'exécution est interdit entre les comportements C_k et C_{k-2}. De plus, la liste des parallélismes, autorisés entre C_{k-2} et ceux en amont, contient deux éléments : les comportements C_{k-4} et C_{k-3}. Dans cette situation, le comportement C_k exécuté après C_{k-2} peut l'être en parallèle des comportements C_{k-4} et C_{k-3}. Au risque de violer les contraintes de sécurité et d'écologie, il est alors obligatoire de vérifier le parallélisme d'exécution entre C_k et les deux comportements C_{k-4} et C_{k-3}.

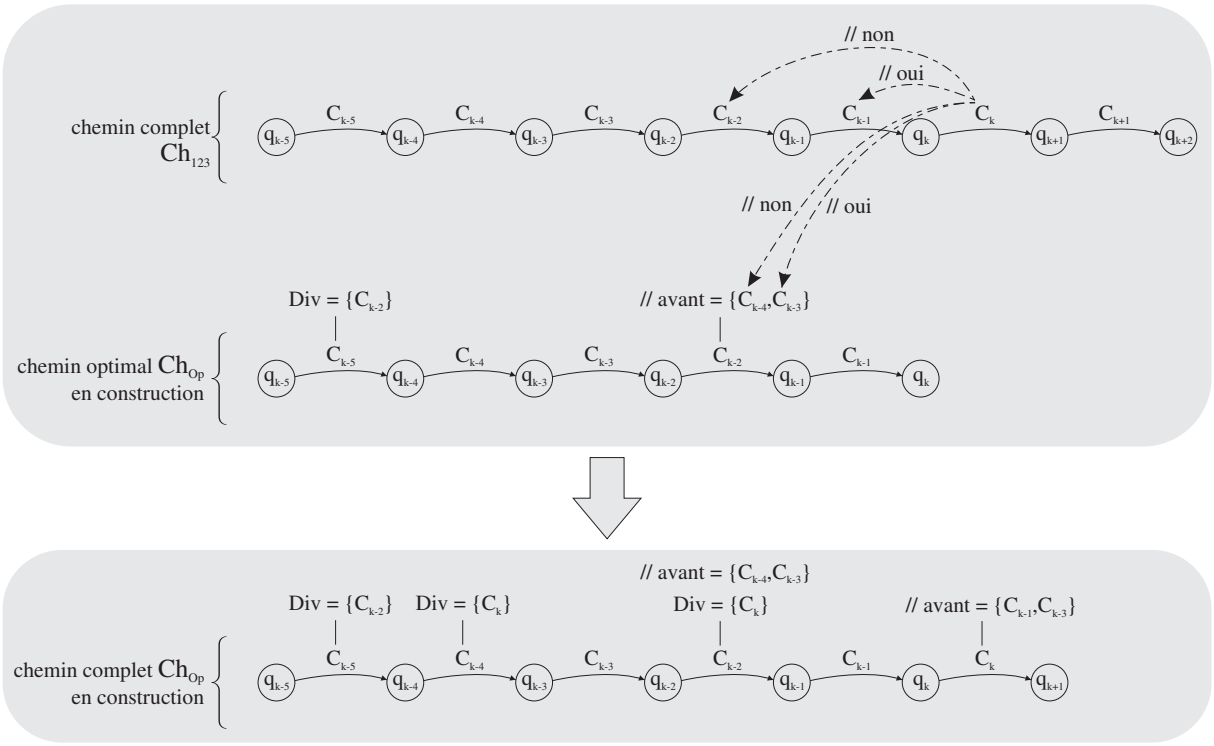
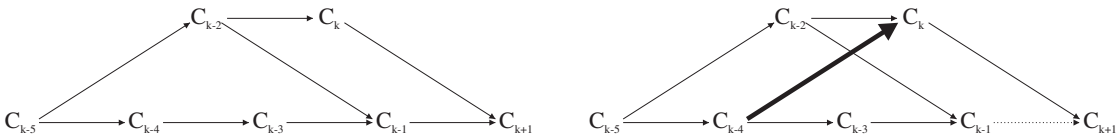


FIG. 8.17 – Parallélismes induits par ceux d'une autre opération.

Si l'exécution en parallèle des comportements C_k et C_{k-3} est autorisée, C_{k-3} est ajouté à la liste des parallélismes autorisés avant C_k. A l'inverse, si ce parallélisme est interdit, C_k est ajouté à la liste d'opérations à exécuter après $C_k - 3$. A condition que le parallélisme soit autorisé entre C_k et C_{k-3}, le même principe est appliqué entre C_k et C_{k-4}.

La Figure 8.18 représente d'une part le graphe de précédence entre les comportements du chemin Ch_{123} de la Figure 8.17 sans tenir compte des parallélismes induits, et d'autre part le graphe de précédence avec la prise en compte des parallélismes induits. Nous remarquons que dans la Figure 8.18(a), il n'existe pas de contraintes de précédence entre C_{k-4} et C_k quand les parallélismes induits ne sont pas considérés. A l'inverse, la prise en compte des parallélismes induits impose une contrainte de précédence entre C_{k-4} et C_k (cf. Figure 8.18(b)).



(a) Graphe de précédence sans prise en compte des parallélismes induits.

(b) Graphe de précédence avec prise en compte des parallélismes induits.

FIG. 8.18 – Comparaison des graphes de précédence avec et sans prise en compte des parallélismes induits.

7 Conclusion

Par la conception d'une loi de commande qui réponde à une demande de fabrication d'un produit, ce chapitre s'est attaché à démontrer tout l'intérêt de la décomposition en trois sous-problèmes et du gel des parallélismes d'exécution. En effet, ces deux principes limitent considérablement la complexité des espaces d'états générés, autorisant ainsi le recours à une technique de recherche du plus court chemin. Suite aux trois étapes, considérant successivement les transformations du produit, les déplacements du produit et enfin la préparation des chaînes fonctionnelles, les parallélismes d'exécution ont été introduits afin d'optimiser le temps d'exécution de la loi de commande.

Les étapes 1, 2, 3 et 4 présentées dans ce chapitre construisent un chemin Ch_{Op} qui spécifie une loi de commande en réponse à une demande imposant :

- un état d'entrée $q_{(1,0)}$ d'un produit et son état de sortie spécifié par un objectif Ob_1 ;
- un état initial $q_{(3,0)} \in Q_3$ des chaînes fonctionnelles pour lequel il n'y a pas de produits dans le système de production.
- l'état final des chaînes fonctionnelles spécifié par un objectif Ob_3
- la position finale, par un objectif Ob_2, des produits qui restent présents dans le système de production
- la minimisation du temps d'exécution pris en compte dans les étapes 1, 2, 3 et 4 ;
- un ou plusieurs autres critères pris en compte dans les étapes 1, 2 et 3.

Les étapes 1, 2, 3 et 4, présentées dans ce chapitre, définissent une fonction Γ dont le résultat est la loi de commande répondant à la demande.

$$\Gamma(q_{(1,0)}, Ob_{(1)}, q_{(3,0)}, Ob_{(2)}, Ob_{(3)}, criteres) = Ch_{Op}$$

Suite au gel des parallélismes, ce chapitre a ré-introduit ceux entre deux opérations destinées à la fabrication d'un produit. Le chapitre suivant se consacre aux problèmes qui requièrent d'introduire des parallélismes entre des opérations liées à des produits différents (phase B et C).

Chapitre 9

Phases B et C : Fabrication de Plusieurs Produits

1 Introduction

Le chapitre précédent nous a amené à proposer un premier mécanisme de synthèse capable de générer une loi de commande pour un produit basé sur des optimisations locales.

Dans le cadre de ce chapitre, nous nous proposons d'étendre ce mécanisme à la problématique du pilotage de plusieurs produits. Nous verrons ainsi, que d'un point de vue général, gérer un problème avec plusieurs produits revient à spécifier tout d'abord l'ensemble des lois de commande optimales pour chacun des produits, puis, de les fusionner "astucieusement". Ainsi, la première section de ce chapitre traite du problème de fabrication de plusieurs produits identiques, qui revient à synthétiser une loi de commande cyclique. Suite à cette section définissant un mécanisme d'imbrication, la suivante se consacre à trois problèmes multi-flux de produits : la fabrication de produits de types différents, la fabrication par assemblage d'un produit, et enfin la reprise suite à une défaillance de la partie opérative.

2 Spécification d'une Loi de Commande Cyclique

2.1 Principe Général

Le problème traité dans ce chapitre est la synthèse d'une loi de commande cyclique afin de répondre à une demande de fabrication de plusieurs produits identiques. Ce problème peut être ramené à celui de la fabrication de deux produits identiques. Cette équivalence sera démontrée au § 2.3.2 de ce même chapitre.

Basé sur le résultat du chapitre 8, une première loi de commande répondant à une demande de deux produits peut être synthétisée en appliquant deux fois la même loi de commande. Ainsi, un premier produit ($p1$) est fabriqué ; il rentre puis il ressort du sous-système de production suite à l'exécution d'une loi de commande spécifiée par un chemin optimal Ch_{Op} ; par application des étapes une à quatre du chapitre 8. A condition que pour ce chemin, l'état initial q_0 et l'état final q_{N+1} soient identiques, un deuxième produit ($p2$) peut être fabriqué en appliquant la même loi de commande basée sur le même chemin Ch_{Op}. Ce scénario est celui de la Figure 9.1(a).

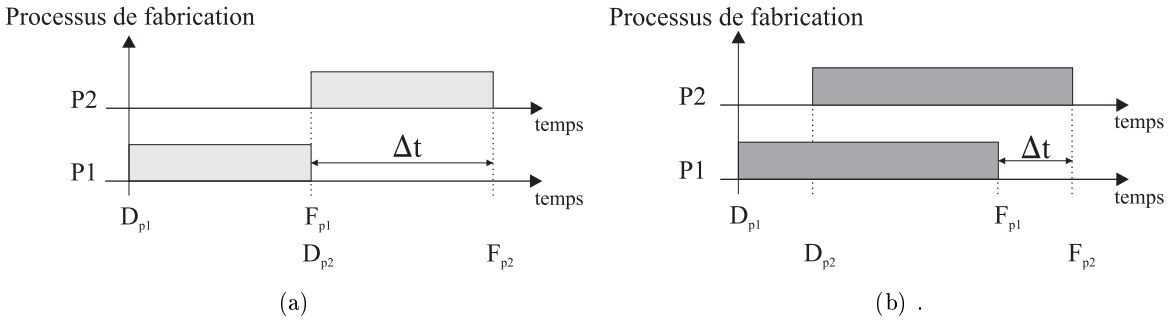


FIG. 9.1 – Durée entre les dates de fin de fabrication de deux produits en fonction de l'imbrication.

Certes cette solution permet de fabriquer deux produits, mais elle n'est pas optimale vis-à-vis de critères tels que la durée totale de fabrication, la durée Δt entre la fin de la fabrication de deux produits, ou bien encore la charge des ressources.

Ainsi, la problématique de conception d'une loi de commande cyclique revient à un problème d'optimisation d'un ou de plusieurs de ces critères (Gentina et al., 2002). Face à la complexité des problèmes multi-critères, nous limiterons notre approche à l'optimisation d'un seul critère. Ainsi afin d'optimiser la cadence de produits fabriqués, nous privilégierons la minimisation de la durée Δt entre la fin de fabrication de deux produits.

Minimiser la durée Δt revient alors à tenter d'imbriquer au mieux les opérations nécessaires à la fabrication de chacun des produits (cf. Figure 9.1(b)). Rappelons que nous supposons ici que les produits sont identiques. Ils suivent le même cheminement au sein du système de production et ils subissent les mêmes transformations. Au final, la fabrication des deux produits sera basée sur le même chemin optimal Ch_{Op}.

Imbriquer les comportements nécessaires pour fabriquer deux produits identiques revient alors à rechercher les contraintes de précédence entre les comportements spécifiés par un chemin optimal $Ch_{p1,Op}$ pour le premier produit et les comportements spécifiés par un chemin optimal $Ch_{p2,Op}$ pour le second produit sachant que $Ch_{p1,Op}$ et $Ch_{p2,Op}$ sont identiques. Ces contraintes conditionnent l'exécution des comportements du chemin $Ch_{p2,Op}$ à la fin de l'exécution de certains comportements du chemin $Ch_{p1,Op}$. Elles assurent le lancement au plus tôt de la fabrication du deuxième produit tout en garantissant de satisfaire les contraintes de sécurité et d'écologie.

Quand toutes les contraintes auront été définies entre les comportements, les deux chemins seront fusionnés puisque identiques. Pour le mécanisme d'imbrication proposé, le résultat est alors un chemin optimal cyclique Ch_{OpC} spécifiant une loi de commande permettant de fabriquer un nombre infini de produits identiques.

En résumé, la résolution du problème traité dans ce chapitre se décompose en quatre phases :

Phase A : rechercher une séquence de comportements afin de fabriquer un produit. De manière à autoriser plusieurs fois l'application de la même séquence, elle résulte de la construction d'un chemin optimal Ch_{Op} dont l'état initial q_0 et l'état final q_{N+1} sont identiques ;

Phase B : imbriquer les comportements liés à la fabrication du deuxième produit avec les comportements destinés à fabriquer le premier produit ;

Phase C : réduire la taille de la loi de commande obtenue par fusion non seulement des deux chemins, $Ch_{p1,Op}$ et $Ch_{p2,Op}$, mais également des listes de contraintes associées à chaque comportement ;

Phase D : traduire la loi de commande générée dans un langage interprétable.

Remarque : afin de ne pas surcharger inutilement la Figure 9.2, cette phase, finalement évidente, n'y est pas représentée.

La première étape est basée sur les résultats du chapitre 8. Il s'agit de la construction d'un chemin optimal Ch_{Op} par appel à la fonction Γ. L'impact de la condition sur l'état final et l'état initial étant minime, nous repoussons volontairement sa présentation à la fin de ce chapitre. Ainsi, la section suivante se focalise sur la phase B d'imbrication de deux chemins.

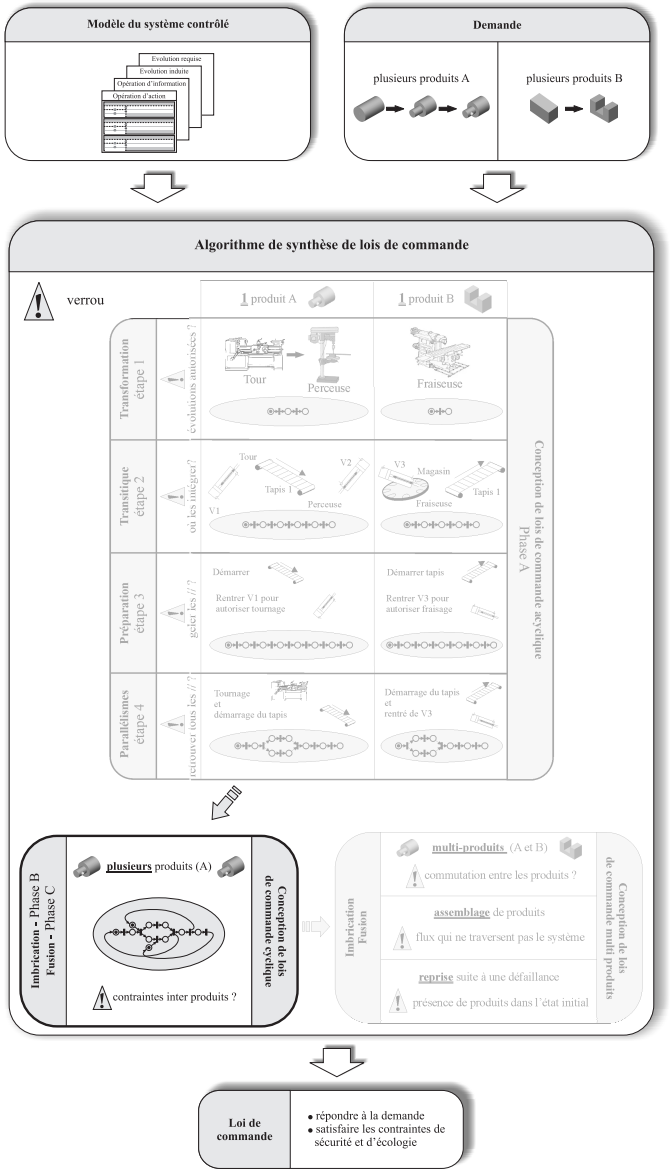


FIG. 9.2 – Phases B et C de conception d'une loi de commande cyclique.

2.2 Phase B : imbrication de deux Chemins Ch_{Op}

Afin de minimiser la durée Δt entre la fin de la fabrication des deux produits, chacun des comportements, $C_{p2,j}$, des opérations destinées à la fabrication du deuxième produit est imbriqué avec la séquence de comportements liés au premier produit. C'est le travail de l'algorithme d'imbrication.

Si un comportement $C_{p2,j}$ ne peut pas être imbriqué alors l'algorithme d'imbrication ne sera pas appliqué à tous ceux en aval de celui-ci.

2.2.1 Phase B : Principe

La minimisation de la durée Δt entre les dates de fin de fabrication des deux produits conduit à minimiser le nombre d'opérations qui restent à exécuter pour le deuxième produit après la dernière opération du premier produit. Ceci passe d'abord par la recherche des parallélismes d'exécution entre les opérations destinées aux deux produits, mais pas uniquement. En effet, un mécanisme d'insertion d'une opération du deuxième produit est envisagé entre deux opérations du premier produit. Avant de détailler ces deux mécanismes, attardons nous un instant sur les raisons qui conduisent d'une part à la possibilité d'insérer une opération, et d'autre part à l'intérêt de ce mécanisme d'insertion.

Depuis un état où deux opérations de chacun des produits sont concurrentes, il est nécessaire de décider de l'opération à exécuter. Dans cet objectif, deux solutions sont envisageables : construire d'une part toutes les solutions pour en choisir une sur la base d'un ensemble de critères, soit envisager une heuristique.

Comme nous le verrons dans la dernière partie de ce mémoire, la complexité de l'espace d'états aurait explosé avec le nombre de produits. Nous rejetons donc cette solution au profit de celle basée sur une heuristique.

Cette heuristique peut être :

- privilégier le produit dont le plus grand nombre d'opérations a déjà été exécuté, ou au contraire,
- privilégier le produit le moins avancé vis-à-vis de sa séquence d'opérations et retarder la fabrication du premier produit.

Comme nous l'avons précisé au début de ce paragraphe, la minimisation de Δt conduit à minimiser le nombre d'opérations qui restent à exécuter pour le deuxième produit après la dernière du premier produit. Ainsi, nous décidons de privilégier l'avancement du deuxième produit.

L'algorithme d'imbrication sera alors basé sur les deux mécanismes suivants :

Mécanisme de parallélisation : rechercher les parallélismes autorisés entre les comportements liés au deuxième produit et ceux du premier. Ceci sera sans impact sur la durée de fabrication du premier produit.

Mécanisme d'insertion : rechercher à insérer un comportement lié au deuxième produit, noté $C_{p2,j}$, entre deux comportements $C_{p1,i}$ et $C_{p1,i+1}$ suite à l'interdiction d'exécution en parallèle de $C_{p2,j}$ et $C_{p1,i+1}$. Ceci modifiera alors la durée de fabrication du premier produit.

Ainsi pour un comportement $C_{p2,j}$, l'algorithme d'imbrication fait appel alternativement à ces deux mécanismes tant que le comportement $C_{p2,j}$ peut être inséré.

Les deux mécanismes sur lesquels se base l'algorithme d'imbrication sont maintenant étudiés.

2.2.2 Mécanisme de parallélisation

La Figure 9.3 illustre le mécanisme de recherche de parallélismes d'exécution entre un comportement $C_{p2,j}$, lié au deuxième produit et les comportements d'un chemin $Ch_{p1,Op}$ destiné à fabriquer le premier produit. Dans cette Figure, la propriété 4 n'est pas vérifiée entre le comportement $C_{p2,k}$ et le comportement $C_{p1,k+2}$. Le comportement $C_{p2,k}$ est alors ajouté à la liste des opérations à exécuter après $C_{p1,k+2}$. Cette liste est celle définie au chapitre précédent. Quant aux comportements $C_{p1,i}$ avec $i \in [k+3, N]$, ils sont tous ajoutés à la liste des parallélismes autorisés entre $C_{p2,k}$ et les comportements liés à la fabrication du premier produit, notée *// produits*.

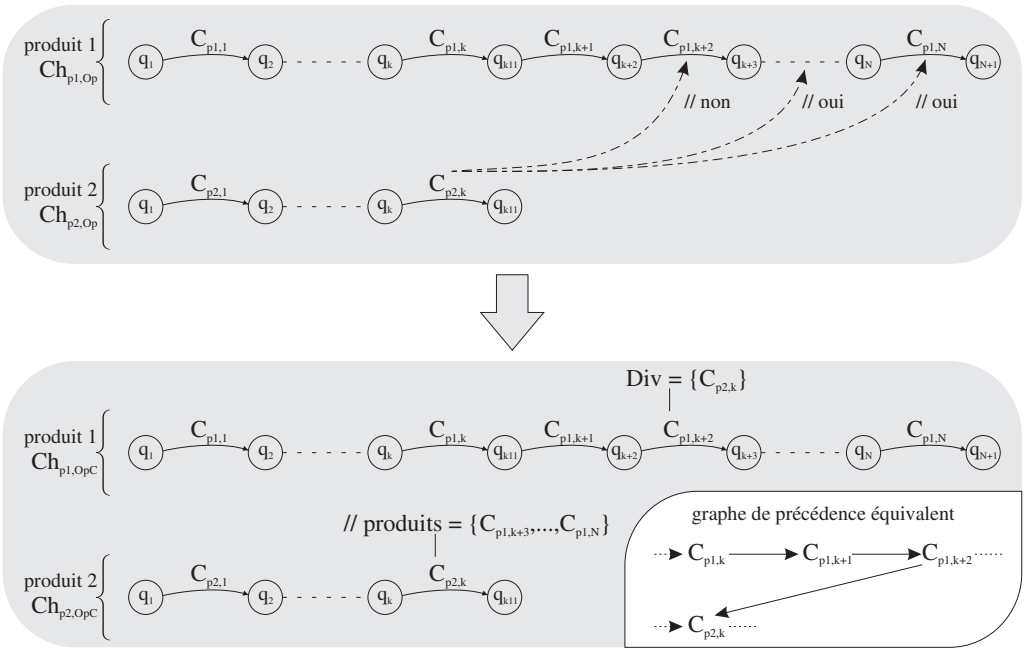


FIG. 9.3 – Parallélismes autorisés entre un comportement $C_{p2,k}$ et les comportements d'un chemin $Ch_{p1,Op}$.

Comme le démontre la section suivante, il est nécessaire de considérer également, dans ce contexte, les parallélismes induits au risque de violer les contraintes de sécurité et d'écologie.

2.2.3 Parallélismes induits

Le phénomène est le même que celui détaillé lors de l'étape 4 d'introduction des parallélismes lors de la fabrication d'un produit (cf. § 6.2 page 142). Pour cette raison, l'exemple donné Figure 9.4, en comparaison avec la Figure 9.3, suffit à comprendre le mécanisme de prise en compte des parallélismes induits et son intérêt.

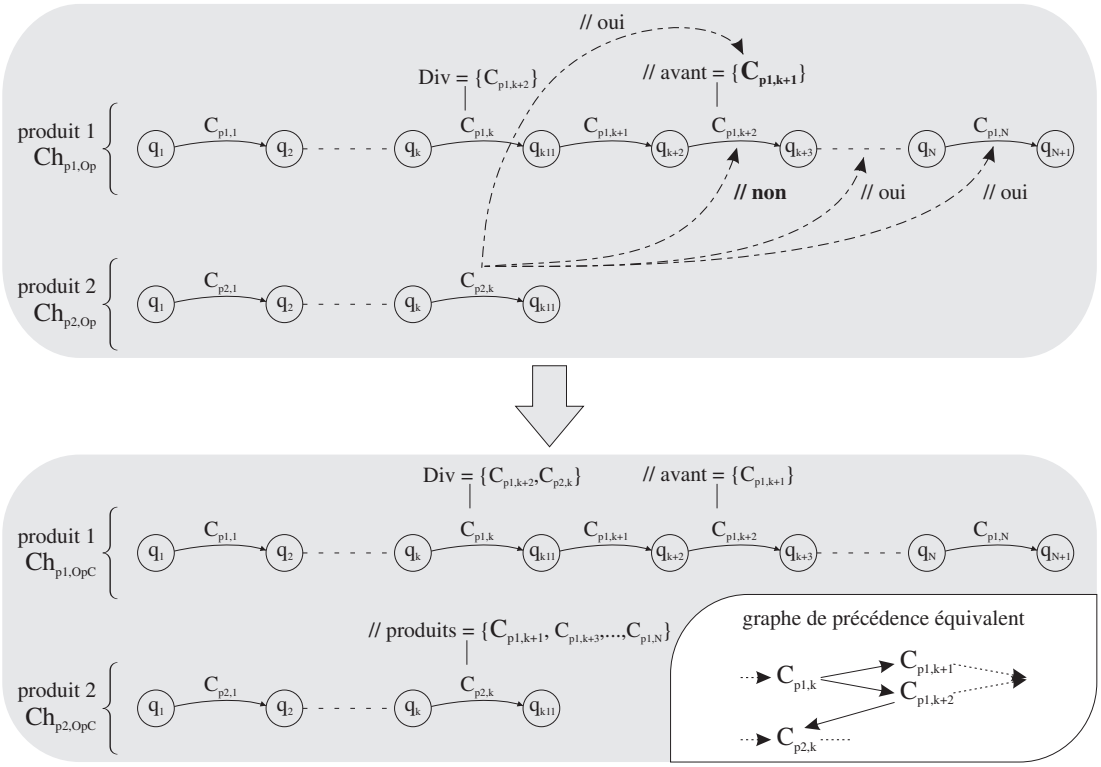


FIG. 9.4 – Parallélismes induits entre un comportement $C_{p2,k}$ et les comportements d'un chemin $Ch_{p1,Op}$.

La recherche des parallélismes présentée ci-dessus ne permet pas toujours d'aboutir à la solution minimisant Δt. Tout particulièrement, quand il est possible d'insérer un comportement $C_{p2,j}$ entre deux comportements $C_{p1,i}$ et $C_{p1,i+1}$. Ainsi, le mécanisme d'insertion est présenté dans la section suivante.

2.2.4 Mécanisme d'insertion

Le mécanisme d'insertion d'un comportement est déclenché après l'échec de la recherche des parallélismes entre les comportements $C_{p2,j}$ du deuxième produit et $C_{p1,i+1}$ du premier produit. En effet, quand le parallélisme d'exécution est interdit entre deux comportements, il peut s'agir d'un cas de concurrence. Si tel est le cas et vis-à-vis de l'heuristique qui rend prioritaire le deuxième produit, il faut insérer un comportement $C_{p2,j}$, lié à ce deuxième produit, entre deux comportements $C_{p1,i}$ et $C_{p1,i+1}$, destinés à la fabrication du premier produit.

Nous présentons d'abord le mécanisme d'insertion avant de détailler la condition d'insertion qui vise à détecter une situation de concurrence.

Afin de présenter le principe du mécanisme d'insertion, la Figure 9.5 donne un exemple pour un comportement $C_{p2,1}$. Si la condition d'insertion, présentée dans la section suivante, est vraie alors le comportement $C_{p2,1}$ est inséré entre les deux comportements $C_{p1,3}$ et $C_{p1,4}$.

Afin de modéliser cette insertion, les informations contenues dans les deux chemins $Ch_{p1,Op}$ et $Ch_{p2,Op}$ doivent être enrichies. Pour cela deux informations sont requises : les contraintes de

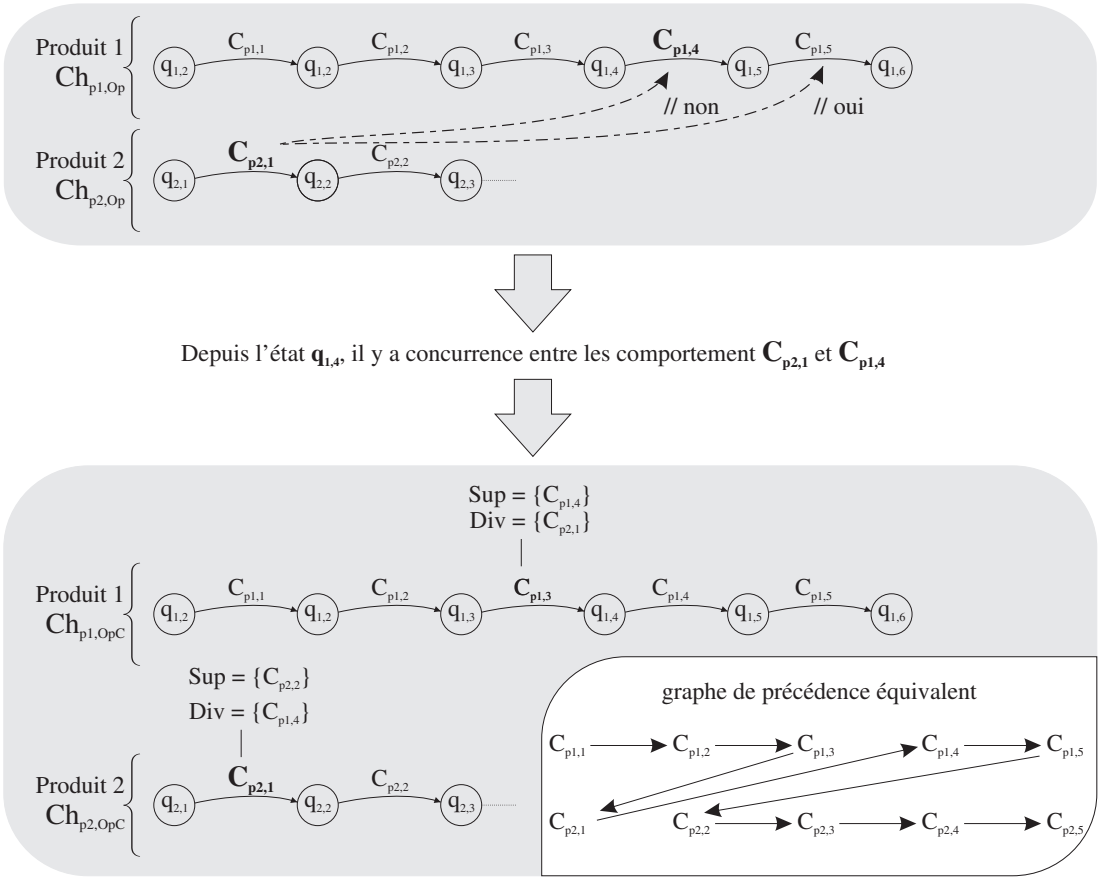


FIG. 9.5 – Principe d'insertion d'un comportement $C_{p2,k}$ entre deux comportements $C_{p1,i}$ et $C_{p1,i+1}$.

précédence supprimées (entre $C_{p1,3}$ et $C_{p1,4}$, et entre $C_{p2,1}$ et $C_{p2,2}$), et les nouvelles contraintes de précédence (entre $C_{p1,3}$ et $C_{p2,1}$, et entre $C_{p2,1}$ et $C_{p1,4}$). Le comportement $C_{p2,2}$ est quant à lui, par défaut, déclenché par la fin du comportement $C_{p1,5}$, jusqu'à son imbrication.

Les informations liées aux nouvelles contraintes de précédence utilisent les listes d'opérations à exécuter après un comportement C_k ; liste (Div). Quant aux contraintes de précédence supprimées, elles nécessitent d'associer à chacun des comportements une nouvelle liste modélisant cette suppression, notée Sup.

2.2.5 Condition d'insertion

La condition d'insertion vérifie qu'il s'agit bien d'un cas de concurrence entre deux comportements liés chacun à un produit. Ainsi, cette condition vise à garantir d'une part la fabrication des deux produits selon les évolutions spécifiées par les chemins $Ch_{p1,Op}$ et $Ch_{p2,Op}$, et d'autre part à satisfaire les contraintes de sécurité et d'écologie.

L'insertion d'un comportement modifie obligatoirement l'état initial du comportement retardé du premier produit. Ainsi, la condition impose de construire ce nouvel état à partir des états des chemins $Ch_{p1,Op}$ et $Ch_{p2,Op}$.

La condition d'insertion, illustrée par la Figure 9.6, est basée sur les quatre points suivants :

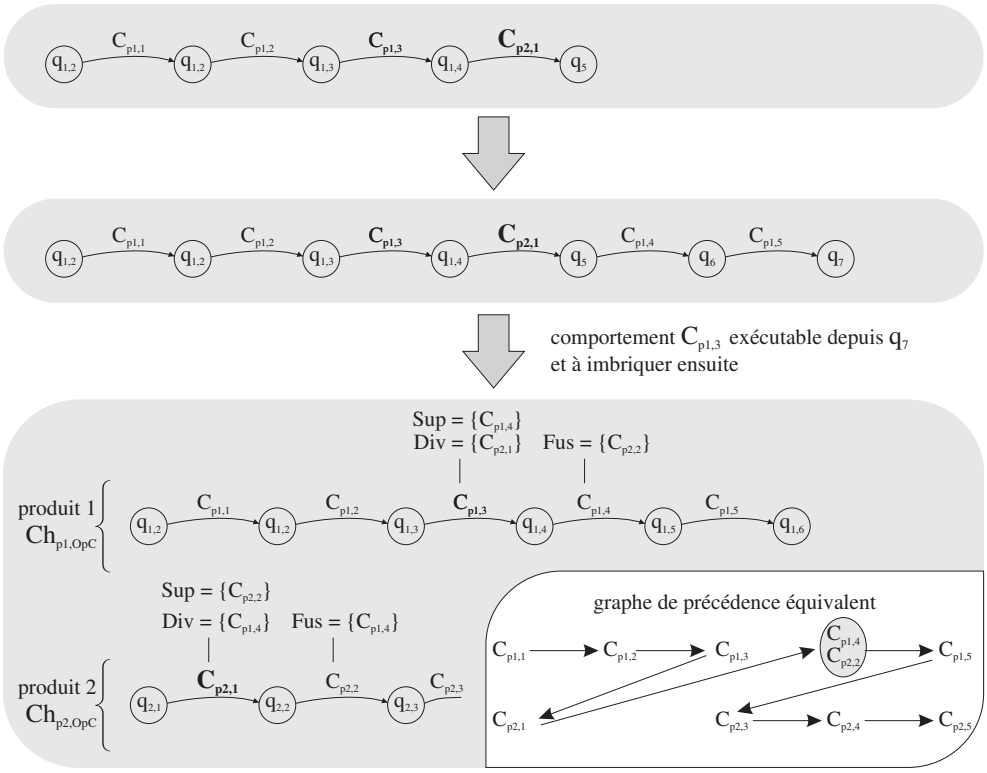


FIG. 9.6 – Condition à satisfaire afin d'insérer un comportement $C_{p2,j}$ entre deux comportements $C_{p1,i}$ et $C_{p1,i+1}$.

1. s'agit-il d'un cas de concurrence entre deux comportements liés chacun à un produit ? Oui, si le comportement à insérer est exécutable depuis l'état précédent le comportement avec lequel le parallélisme d'exécution est interdit. Dans l'exemple de la Figure 9.5, le comportement $C_{p2,1}$ est autorisé depuis l'état $q_{1,4}$.
2. le comportement du premier produit dont l'exécution est retardée est-il encore exécutable depuis l'état atteint suite à l'application du comportement inséré ? Dans l'exemple de la Figure 9.5, le comportement $C_{p1,4}$ est autorisé depuis le nouvel l'état q_5. La même vérification est à effectuer pour tous les comportements suivants du premier produit. Dans l'exemple de la Figure 9.5, le comportement $C_{p1,5}$ est autorisé depuis le nouvel l'état q_6. Suite à cette étape, nous avons vérifié que l'insertion autorise la fabrication du premier produit.
3. si les opérations sur lesquelles sont basés les comportements du premier produit ont un effet sur le second produit, ces effets doivent être ceux des comportements du deuxième produit situés après le comportement inséré. Dans l'exemple de la Figure 9.5, le comportement $C_{p1,4}$ a non seulement l'effet prévu sur le premier produit mais également l'effet du comportement $C_{p2,2}$ sur le deuxième produit. Alors, les comportements $C_{p1,4}$ et $C_{p2,2}$ sont fusionnés car ils résulteront de l'exécution de la même opération.
4. les comportements du deuxième produit dont les effets ne sont pas réalisés par les opérations des comportements du premier produit sont-ils exécutables ? Dans l'exemple de la Figure 9.5, le comportement $C_{p2,3}$ et les suivants sont autorisés depuis l'état q_7.

Notons que la fusion de deux comportements nécessite d'enrichir de nouveau les chemins $Ch_{p1,Op}$ et $Ch_{p2,Op}$ en associant à chacun des comportements, une liste des fusions, notée *Fus*.

2.2.6 Bilan de la phase B

L'étude de la phase B d'imbrication des deux chemins est maintenant terminée. L'algorithme d'imbrication à la base de cette deuxième étape fait appel aux mécanismes de parallélisation et d'insertion. Suite à l'application de l'algorithme d'imbrication à chacun des comportements destinés à fabriquer le deuxième produit, il résulte deux chemins $Ch_{p1,OpC}$ et $Ch_{p2,OpC}$.

Quatre listes sont associées à chacun des comportements pour la fabrication du premier produit (chemin $Ch_{p1,OpC}$) : // *avant*, *Div*, *Sup* et *Fus*. Une cinquième liste est associée aux comportements destinés à fabriquer le deuxième produit (chemin $Ch_{p2,OpC}$) : // *produits*.

L'exécution des opérations, desquelles les comportements de chacun des deux chemins sont issus, en respectant les contraintes de précédence définies à la fois par les chemins et par les cinq listes énumérées ci-dessus fait évoluer le système contrôlé d'un état initial vers un état final tel que deux produits identiques soient fabriqués. Afin d'imposer cette évolution au système contrôlé, il est nécessaire de développer un moteur d'exploitation capable d'interpréter le formalisme utilisé, et donc, les deux chemins en parallèle.

Afin de ne pas devoir mémoriser autant de chemins que de produits à fabriquer et d'exprimer la loi de commande dans un langage interprétable, les deux étapes suivantes (C et D) proposent respectivement de fusionner les deux chemins, et de traduire la loi de commande en un réseau de Petri.

2.3 Phase C : Fusion de deux chemins

Afin d'aboutir à une représentation plus compacte de la solution, cette troisième étape va s'attacher à d'une part montrer le principe de fusion de deux chemins $Ch_{p1,OpC}$ et $Ch_{p2,OpC}$, et d'autre part démontrer que le résultat de la fusion des deux chemins est identique à celui de n chemins, et ainsi, pouvoir fabriquer n produits.

2.3.1 Fusion

Les états et les transitions des deux chemins $Ch_{p1,OpC}$ et $Ch_{p2,OpC}$ sont identiques. Seules les listes associées à un même comportement lié à chacun des produits, $C_{p1,k}$ et $C_{p2,k}$, sont différentes. L'algorithme 6 de fusion de deux chemins consiste alors à fusionner les listes en réalisant l'union des éléments.

La demande considérée jusqu'ici imposait la fabrication de deux produits. En réponse à cette demande, deux chemins destinés chacun à fabriquer un produit ont été construits avec les contraintes de précédence entre leurs comportements. Le chemin résultant de la fusion répond également à une demande d'un nombre infini de produits, comme le démontre la section suivante.

Fonction FusionChemins($Ch_{p1,OpC}$, $Ch_{p2,OpC}$) : **chemin optimal cyclique**

$k = 1$

$q_1 = q_{p1,1} = q_{p2,1}$

Tant que $(k \leqslant N)$ **faire**

$q_{k+1} = q_{p1,k+1} = q_{p2,k+1}$

$C_k = C_{p1,k} = C_{p2,k}$

$q_{k+1} = \delta(q_k, C_k)$

$\backslash\backslash\ avant(C_k) = \backslash\backslash\ avant(C_{p1,k}) \cup \backslash\backslash\ avant(C_{p2,k})$

$Div(C_k) = Div(C_{p1,k}) \cup Div(C_{p2,k})$

$Sup(C_k) = Sup(C_{p1,k}) \cup Sup(C_{p2,k})$

$Fus(C_k) = Fus(C_{p1,k}) \cup Fus(C_{p2,k})$

$\backslash\backslash\ produits(C_k) = \backslash\backslash\ produits(C_{p2,k})$

Fin Tq

Retourner Ch_{OpC} ;

Fin

ALG. 6: Fusion de deux chemins destinés à fabriquer deux produits identiques.

2.3.2 Généralisation à N produits

Afin de démontrer que le résultat de la fusion des deux chemins $Ch_{p1,OpC}$ et $Ch_{p2,OpC}$ est équivalent à la fusion de N chemins destinés à fabriquer N pièces identiques, nous considérons la Figure 9.7 qui représente le premier chemin lié au produit 1, le dernier chemin lié au produit N et le chemin x destiné à fabriquer le produit x.

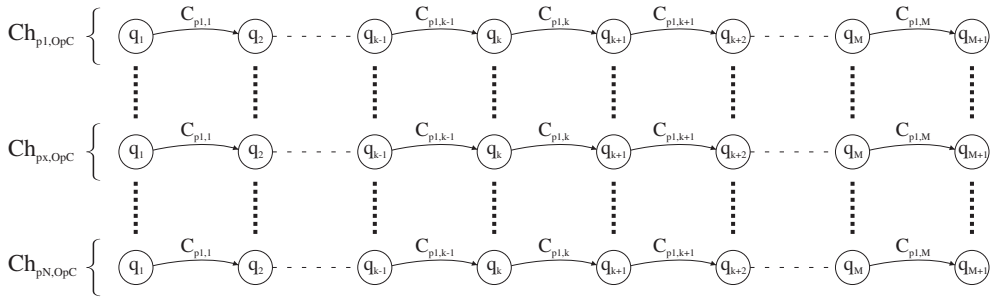


FIG. 9.7 – Les N chemins $Ch_{px,OpC}$ identiques destinés chacun à fabriquer un produit identique.

Afin de minimiser Δt entre la date de fin de fabrication de deux produits x et $x+1$, l'étape B d'imbrication de deux chemins est appliquée pour les chemins $Ch_{px,OpC}$ et $Ch_{px+1,OpC}$ avec $x \in [1, N-1]$. De cette manière, le produit 1 sera suivi au plus près par le produit 2 qui sera lui même suivi au plus près par le produit 3 et ainsi de suite. Le résultat est alors un ensemble de N chemins $Ch_{px,OpC}$. La fusion de ces N chemins conduit à un chemin Ch_{OpC} ayant les mêmes états et transitions que chacun des chemins. De ce point de vue, le résultat est identique au chemin résultant de la fusion de deux chemins. Il reste maintenant à vérifier que les listes associées à chacun des comportements sont les mêmes pour 2 produits et N produits.

Les contraintes de précédence entre deux chemins $Ch_{px-1,OpC}$ $Ch_{px,OpC}$ sont les mêmes que celles entre les chemins $Ch_{px,OpC}$ $Ch_{px+1,OpC}$. Ainsi, il n'existera pas d'autres contraintes de précédence que celles déterminées dans le problème avec deux produits. Finalement, la fusion des deux chemins $Ch_{p1,OpC}$ et $Ch_{p2,OpC}$ lors de cette troisième étape génère un chemin unique Ch_{OpC} qui spécifie une loi de commande cyclique capable de fabriquer un nombre infini de produits identiques.

La section suivante peut donc désormais s'intéresser à la traduction de cette loi de commande cyclique en réseau de Petri (RdP).

2.4 Phase D : Représentation de la Solution par un RdP

Cette quatrième étape revient donc à traduire un chemin, avec les listes associées aux comportements, en réseau de Petri. Cette traduction sera réalisée en trois étapes. En premier lieu, le réseau de Petri non-interprété est extrait du résultat de la phase précédente. Ensuite, son interprétation sera proposée pour des raisons évidentes de commande. Enfin, la détermination du marquage initial du réseau sera traitée.

2.4.1 RdP non interprété

Afin d'expliquer la traduction d'un chemin Ch_{OpC} en un réseau de Petri non interprété, nous proposons de passer par une représentation intermédiaire utilisant le formalisme des graphes de précédence.

La conversion d'un chemin Ch_{OpC} en un graphe de précédence est basée sur la représentation des contraintes de précédence définies d'une part par les transitions du chemin Ch_{OpC} (cf. Figure 9.9(a)), et d'autre part par la liste des divergences associée à chacun des comportements (cf. Figure 9.9(b)).

En revanche, il y a deux cas pour lesquels une contrainte de précédence définie par les transitions du chemin Ch_{OpC} est supprimée. Premièrement lors d'une insertion entre deux comportements, C_n et C_{n+1}, la contrainte de précédence entre ces deux comportements est supprimée. Cette suppression conduit à ajouter le comportement C_{n+1} à la liste (*Sup*) du comportement C_n (cf. Figure 9.9(c)). Le second cas est celui où deux comportements successifs dans le chemin Ch_{OpC}, C_n et C_{n+1}, peuvent être exécutés en parallèle. Dans ce cas, la liste (\\ *avant*) du comportement C_{n+1} contient le comportement C_n (cf. Figure 9.9(d)).

Suite à cette conversion du formalisme propre à l'algorithme en graphe de précédence,

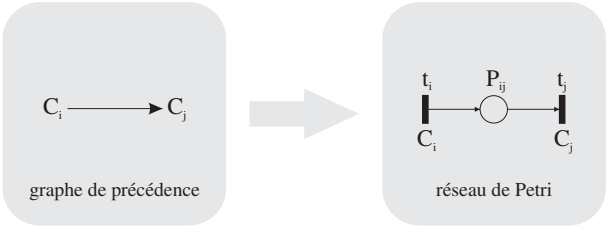


FIG. 9.8 – Représentation d'une contrainte de précédence avec le formalisme réseau de Petri.

nous proposons maintenant la traduction en réseau de Petri non interprété $\mathcal{R}$ basée sur la règle suivante (cf. Figure 9.8) : considérons le réseau de Petri défini par le quadruplet $\mathcal{R} =< P, T, Pre, Post >$ (Valette, 2002), alors une contrainte de précédence entre deux comportements C_i et C_j spécifie les deux fonctions de transitions $Post(P_{ij}, t_i) = 1$ et $Pre(P_{ij}, t_j) = 1$.

La Figure 9.10 illustre l'application de la règle définie ci-dessus pour une convergence (9.10(a)) et une divergence (9.10(b)).

Finalement à partir de la conversion du formalisme propre à l'algorithme de synthèse en graphe de précédence, puis la traduction de ce graphe en réseau de Petri, un chemin Ch_{OpC} résultat de la phase C est traduit en un réseau de Petri non interprété. Toutefois la spécification d'une loi de commande ne se limite pas à une structure de contrôle. Il est en effet nécessaire

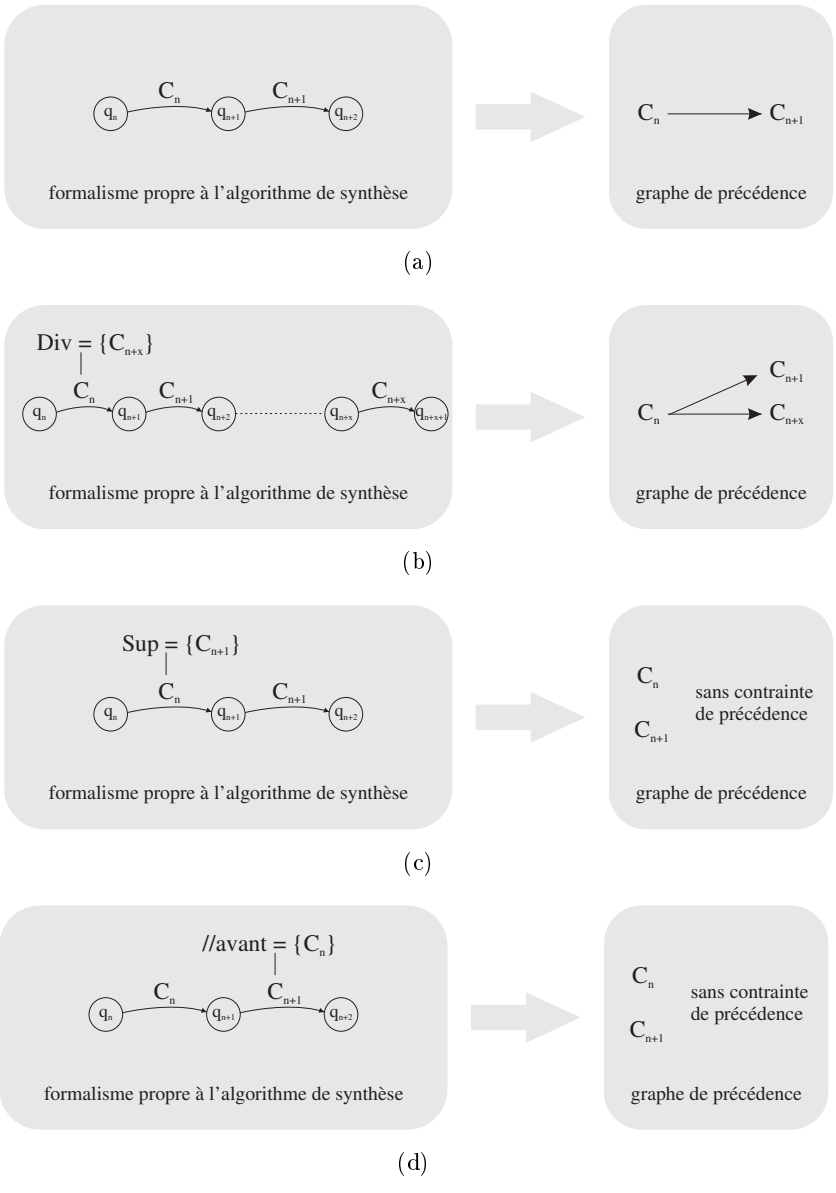
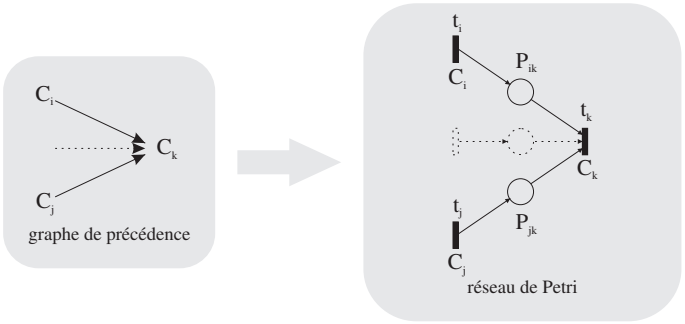
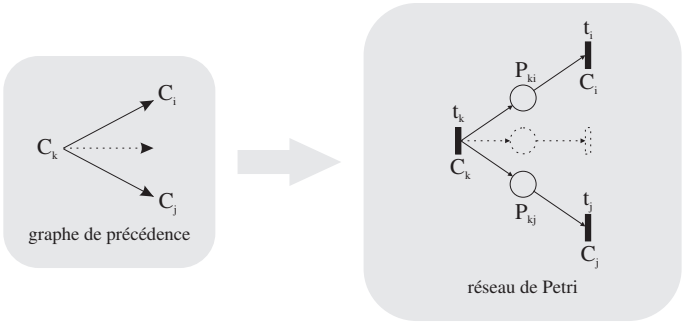


FIG. 9.9 – Conversion du formalisme propre à l'algorithme de synthèse en graphe de précédence.



(a)



(b)

FIG. 9.10 – Traduction en réseau de Petri d'une divergence et d'une convergence du graphe de précédence.

de donner une signification aux places et aux transitions vis-à-vis des requêtes à envoyer pour demander les opérations, des comptes rendus de fin à recevoir et de l'opération en cours. L'interprétation du réseau de Petri $\mathcal{R}$ est présentée dans la section suivante.

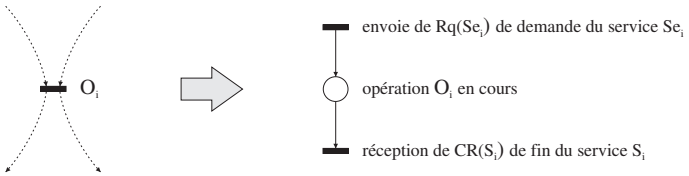
2.4.2 RdP interprété

L'interprétation du réseau de Petri vise à spécifier pour le module de coordination les requêtes à envoyer en fonction des comptes rendus reçus et des informations données par l'environnement. Les seuls éléments que connait le module de coordination sont les grandeurs de commande.

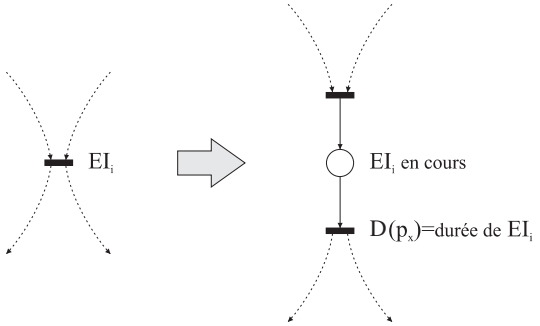
Ainsi, l'interprétation complète du réseau de Petri $\mathcal{R}$ s'exprime sous la forme (Valette, 2002) :
- d'associations aux transitions de comptes rendus de fin de réalisation des services, et/ou d'informations en provenance de l'environnement,
- d'associations aux transitions de requêtes d'appel aux services offerts par les chaînes fonctionnelles,
- de représentation d'une opération en cours d'exécution par une place.

L'interprétation du réseau de Petri à travers les éléments énoncés ci-dessus, repose alors sur les règles illustrées par les figures 9.11(a), 9.11(b) et 9.11(c) qui représentent l'interprétation résultante de l'association à une transition respectivement d'une opération d'action ou d'information, d'une évolution induite et enfin d'une évolution requise.

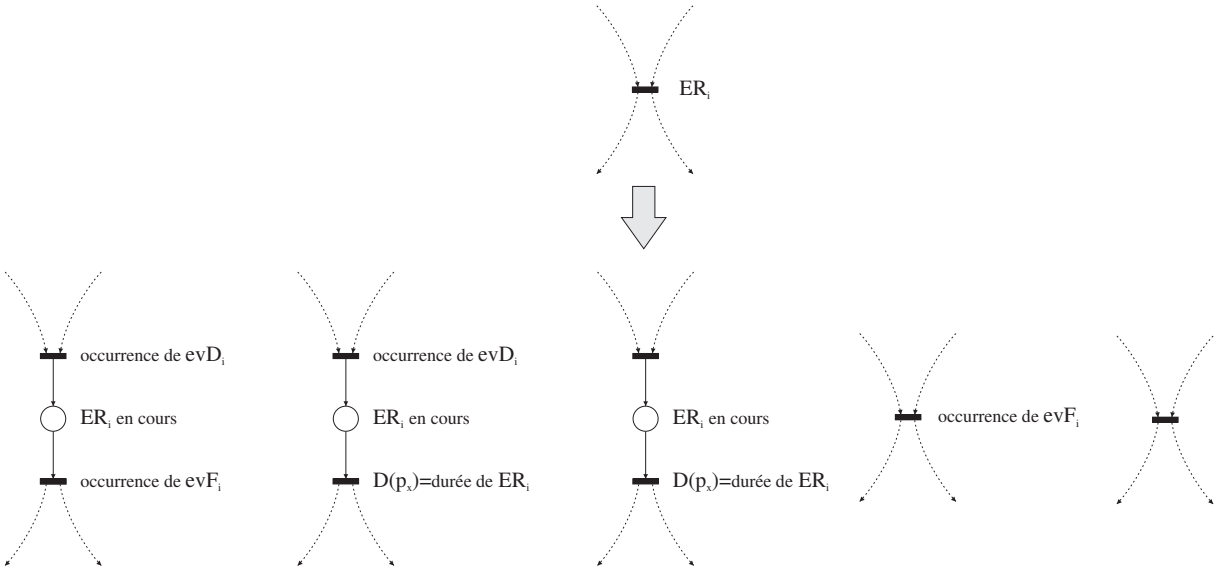
Suite à l'interprétation du réseau de Petri et afin d'aboutir à la spécification d'une loi de commande, un marquage initial doit être défini comme le propose la section suivante.



(a) Une opération d'action ou d'information.



(b) Une évolution induite.



(c) Une évolution requise.

FIG. 9.11 – Interprétation du RdP.

2.4.3 Marquage initial

La spécification d'une loi de commande par un réseau de Petri nécessite non seulement un réseau $\mathcal{R}$ interprété mais également un marquage initial M définissant la ou les premières actions à exécuter. Le couple $\mathcal{R}$ et M forme le réseau marqué $\mathcal{N} =< \mathcal{R}, M >$. Ainsi afin d'aboutir à la spécification d'une loi de commande, il reste à déterminer le marquage initial M à partir du chemin Ch_{OpC}.

Le marquage initial M est déterminé par les deux règles suivantes :

1. placer un jeton dans la ou les place(s) précédent la transition de début de la première opération spécifiée par le chemin Ch_{OpC} ;

2. placer un jeton dans chacune des places, à condition qu'elles soient vides, en amont des opérations dont l'exécution est contrainte par la fin d'une opération située plus loin dans le chemin Ch_{OpC}.

La fin de la phase D fournit un réseau de Petri marqué $\mathcal{N}$ qui est la spécification d'une loi de commande à partir de laquelle un nombre infini de produits identiques peut être fabriqué.

La synthèse de lois de commande proposée dans ce chapitre repose sur le respect de la condition, dite de cycle, qui impose pour le chemin $Ch_{p1,Op}$ lié à la fabrication du premier produit, que son état initial $q_{p1,0}$ et son état final $q_{p1,N+1}$ soient identiques (cf. phase A, § 2 page 145). Afin de généraliser l'approche, nous considérons une demande dont l'état initial et l'objectif ne permettent pas de trouver un chemin pour le premier produit $Ch_{p1,Op}$ qui respecte la condition de cycle.

2.5 Discussions sur la Condition de Cycle

La demande générale que vise à traiter ce chapitre ne conduit pas forcément à vérifier la condition de cycle. Ainsi, nous proposons dans cette section de montrer comment traiter ce problème à partir des résultats précédents.

La demande générale spécifie un état physique d'entrée des produits $q_{(1,0)}$ et un état de sortie des produits par un objectif $Ob_{(1)}$. Elle spécifie également un état initial du système contrôlé $q_{(3,0)} \in Q_3$ avant l'entrée du premier produit. L'état final du système contrôlé à atteindre après la fabrication du dernier produit est spécifié par les objectifs Ob_2 et Ob_3.

Afin d'utiliser les résultats obtenus sur les lois de commande cycliques, la section suivante propose de concevoir des lois de commande de mise en marche et de mise à l'arrêt.

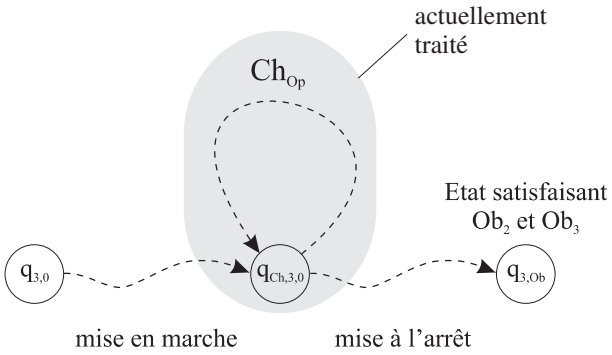


FIG. 9.12 – Représentation d'une loi de commande cyclique avec les lois de commande de mise en marche et de mise à l'arrêt.

2.5.1 Satisfaction de la condition de cycle

Quand la condition de cycle n'est pas satisfaite, il est nécessaire de définir une loi de commande de mise en marche et/ou une loi de commande de mise à l'arrêt. La condition de cycle sera alors satisfaite depuis un état $q_{Ch,3,0}$ atteint suite à l'exécution de la loi de commande de mise en marche. Et les objectifs Ob_2 et Ob_3 imposés par la demande seront satisfaits à leur tour suite à l'application de la loi de commande de mise à l'arrêt depuis l'état $q_{Ch,3,0}$ (cf.

Figure 9.12). Cet état qui est l'état initial d'application de la loi de commande cyclique est aussi l'état de sortie de cycle. La conception d'une loi de commande cyclique requiert un chemin Ch_{Op}, pour fabriquer un produit, dont l'état initial et l'état final sont identiques.

La demande qui impose l'état initial $q_{3,0}$ et l'objectif $q_{3,Ob}$ ne spécifie pas l'état $q_{Ch,3,0}$ dont la connaissance est capitale pour trois raisons : il constitue l'objectif de la loi de commande de mise en marche, c'est l'état initial et final d'un chemin visant à fabriquer un produit, et enfin il représente l'état initial de la loi de commande de mise à l'arrêt (cf. Figure 9.12).

Dans une démarche de configuration manuelle, l'expert utilisant le Guide des Modes de Marches et d'Arrêts (GEMMA) devra d'abord définir les états de commutation entre les modes, puis spécifier les lois de commande pour chacun des modes.

Dans un contexte de fonctionnement normal, notre approche vise à limiter l'intervention de l'expert à la définition de l'état de mise en marche et de l'état d'arrêt. Les états de commutation entre les modes de mise en marche et de production, ainsi que de production et de mise à l'arrêt, seront automatiquement définis par l'algorithme de synthèse.

Ainsi, la fonction Γ sera appliquée depuis l'état $q_{3,0}$ plusieurs fois jusqu'à trouver un chemin $Ch_{px,Op}$ dont l'état initial et l'état final sont identiques. Ils définissent alors l'état recherché $q_{Ch,3,0}$ de commutation permettant de générer la loi de commande cyclique et les lois de commande de mise en marche et d'arrêt. Cette solution est illustrée par la Figure 9.13. Il reste néanmoins à résoudre le problème de récursivité de manière à ne pas chercher indéfiniment un chemin respectant la condition de cycle.

Si l'état initial et les objectifs imposés par la demande peuvent conduire à ne pas satisfaire la condition de cycle pour le premier produit fabriqué, ce phénomène peut se produire également lors d'une insertion réalisée par l'algorithme d'imbrication.

2.5.2 Insertion ne satisfaisant plus la condition de cycle

Considérons une séquence de deux opérations, O_1 et O_2, qui respecte la condition de cycle. L'exécution successive des deux opérations depuis un état q_1 conduit un état q_2 puis de nouveau dans l'état q_1. Suite à l'insertion d'une opération entre les deux actions O_1 et O_2, l'opération 2 n'est plus lancée depuis l'état q_2. L'état atteint suite à cette opération 2 n'est donc plus l'état $q1$. La condition de cycle n'est alors plus vérifiée. Il s'avère nécessaire de définir une loi de commande de mise en marche et une loi de commande de mise à l'arrêt.

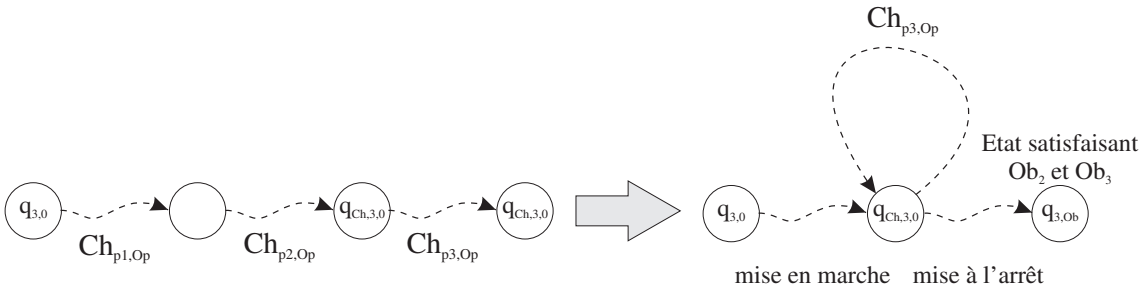


FIG. 9.13 – Détermination de l'état $q_{Ch,3,0}$ à partir duquel la fabrication d'un produit conduit de nouveau dans cet état.

Les propos ci-dessus sont illustrés à partir d'un exemple simple présenté dans la Figure 9.14. Les deux chemins qui sont identiques sont composés de trois comportements. Suite à l'application de l'algorithme d'imbrication, le premier comportement du deuxième chemin est inséré entre le deuxième et le troisième comportement du premier chemin. Il n'y a pas d'autres insertions ou parallélismes.

Le graphe de précédence cyclique équivalent au chemin Ch_{OpC} est applicable uniquement depuis l'état q_3 et ne permet plus d'atteindre l'état objectif q_4. Par conséquent, la loi de commande cyclique spécifiée par ce graphe de précédence nécessite une loi de commande de mise en marche afin d'atteindre l'état q_3 et une loi de commande de mise à l'arrêt afin d'atteindre l'état q_4. La Figure 9.16 représente la spécification en réseaux de Petri des lois de commande de mise en marche, cyclique et de mise à l'arrêt.

Cette étude sur la condition de cycle marque la fin de l'étude de l'algorithme de synthèse de lois de commande cycliques. La conception de ces dernières est basée sur quatre phases : la phase A de génération d'une séquence d'opérations pour un produit, la phase B d'imbrication de deux séquences, la phase C de fusion qui conduit à spécifier une loi de commande cyclique, et enfin la phase D de traduction dans un langage interprétable. Les lois de commande synthétisées jusqu'à présent se limitent au pilotage d'un seul flux de produits.

Basé sur ces résultats, la section suivante traite de la problématique multi-flux de produits, présenté dans la Figure 9.15, qui vise notamment à la fabrication de produits dont les spécifications sont différentes, à la fabrication de produits par assemblage, et à la reprise suite à une défaillance de la partie opérative.

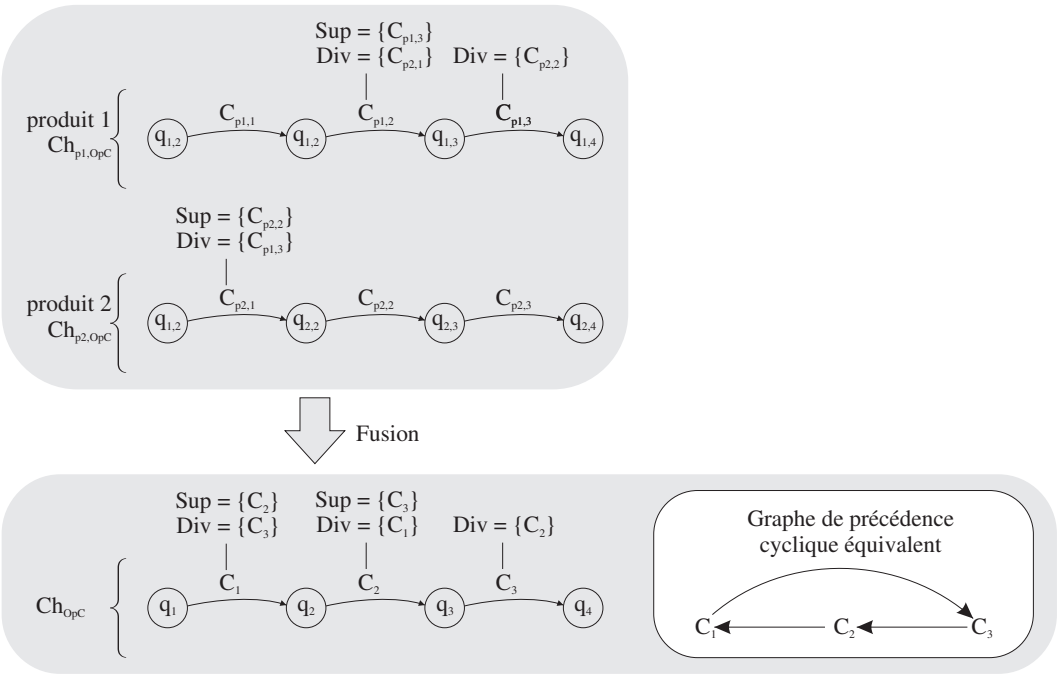


FIG. 9.14 – Exemple de l'insertion d'un comportement conduisant à ne plus satisfaire la condition de cycle.

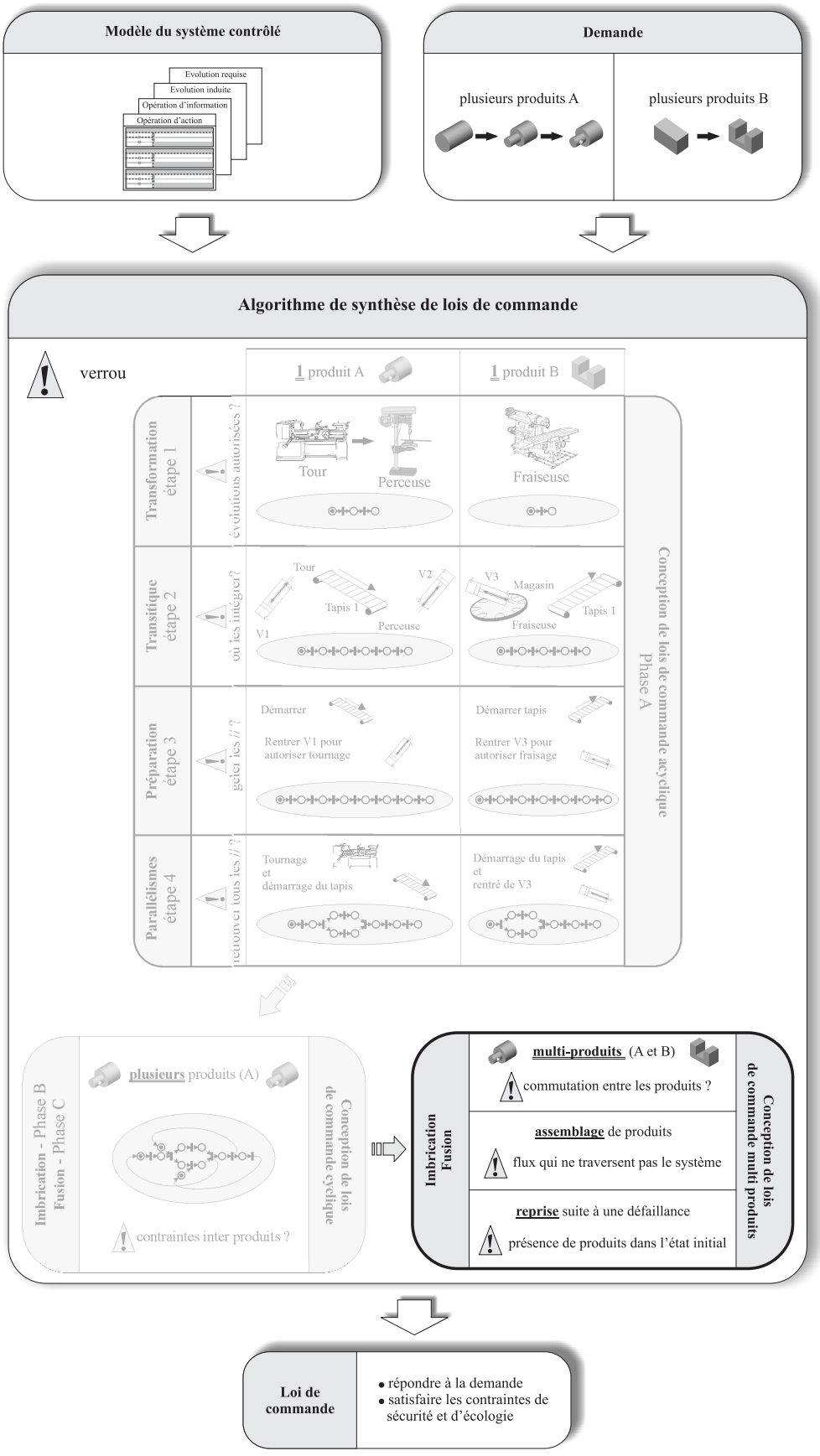


FIG. 9.15 – Conception de lois de commande dans un contexte multi-flux de produits.

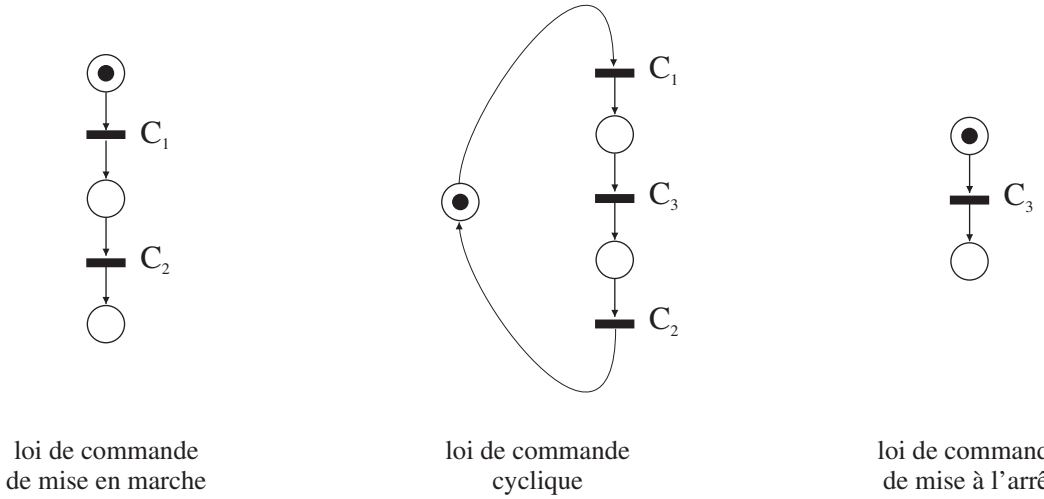


FIG. 9.16 – Lois de commande de mise en marche, cyclique, et de mise à l'arrêt.

3 Multi-Produits : Reprise suite à une défaillance

Suite à la synthèse de lois de commande cycliques, cette section propose de s'intéresser au pilotage de plusieurs flux de produits. Lorsqu'il s'agit de considérer une telle problématique, force est de constater que la difficulté majeure réside dans la coordination de ces flux dans le respect des contraintes de sécurité et d'écologie. Cette difficulté est du même ordre que celle traitée pour la fabrication de produits identiques (coordination des opérations liées à chacun des produits). Pour cette raison, cette section se limitera à donner les verrous, et les clefs pour les résoudre, liés aux problèmes multi-flux (cf. Figure 9.15).

Nous avons choisi de présenter le cas particulier de la reprise qui est très représentatif de la problématique multi-flux. Aussi dans un soucis de concision, les deux autres cas, la fabrication de produits avec des spécifications différentes et la fabrication par assemblage, sont rejetés et traités dans l'annexe B.

La conception de lois de commande à des fins de reprise est un problème complexe pour lequel nous restreignons volontairement l'étude à la génération complète d'une nouvelle loi de commande. Cependant, comme le montrerons les perspectives, d'autres situations devront être envisagées.

Néanmoins ce problème de synthèse d'une nouvelle loi de commande suite à une défaillance reste un problème complexe. En effet contrairement à tous les problèmes traités jusqu'à maintenant, il existe vraisemblablement suite à une défaillance, ayant conduit à un arrêt de la production, des produits présents. Cette différence majeure a par exemple pour conséquence l'impossibilité d'agir sur certains produits, sans agir sur les autres. Comment choisir alors le produit sur lequel agir pour ne pas devoir représenter les évolutions des autres ? Après avoir caractérisé une demande de reprise, nous allons nous intéresser à donner des stratégies basées sur les concepts vus précédemment.

3.1 Caractérisation de la demande

A l'inverse des demandes considérées jusqu'à maintenant qui provenaient toutes d'une requête de configuration, la demande liée à une reprise suite à l'occurrence d'une défaillance émane de la fonction de supervision *Décider* qui se base pour cela sur les résultats des fonctions de surveillance *Diagnostiquer*, *Pronostiquer* et *Suivre*. Ces fonctions devront suivant la défaillance et son impact avoir re-synchronisé le modèle du système contrôlé et de son environnement sur leur état réel.

Il n'est pas facile de donner un format générique d'une demande liée à une reprise puisque les situations peuvent être très variées et dépendront de la stratégie de la fonction *Décider* basée sur la demande en cours et la politique interne de l'entreprise (Mendez, 2002). Dans cette section, nous allons considérer un état initial $q_{3,0}$ avec des produits présents dans le système de production. Cet état initial $q_{3,0}$ est l'état depuis lequel la loi de commande destinée à la reprise sera exécutée. Afin de ne pas complexifier davantage le problème de la reprise, aucun autre produit, mis à part ceux déjà présents, n'entreront dans le sous-système de production considéré, à moins que ce ne soit une nécessité. De ce fait, l'état $q_{1,0}$ servant à spécifier l'état physique d'entrée des produits est vide.

De plus, nous considérons que la demande spécifie premièrement un objectif $Ob_1 = \{Ob_{1,x} \mid x \in [1, P] \text{ avec } Ob_{1,x}\}$ pour un des P produits présents dans le système production lors de la défaillance, deuxièmement un objectif $Ob_2 = \{Ob_{2,x} \mid x \in [1, P]\}$ imposant la position des produits, et enfin un objectif Ob_3 spécifiant l'état à atteindre des chaînes fonctionnelles. Ces objectifs dépendront du résultat de la fonction *Décider*. Afin de les préciser davantage, trois décisions qui couvrent un grand nombre de situations (Berruet et al., 1999) sont envisagées.

La première décision consiste à évacuer l'ensemble des produits sans leur faire subir de transformation et ceci par un stock de sortie quelconque. Suite à la reprise, le problème de conception de loi de commande revient à un des problèmes traités précédemment dans ce chapitre ou dans le chapitre 8. Les objectifs Ob_1, x imposés par la demande sont alors l'état physique des produits dans l'état $q_{3,0}$. L'objectif Ob_2 est l'absence de produit dans le système production. Et enfin, l'objectif Ob_3 contraint l'état final des chaînes fonctionnelles.

La deuxième décision est l'évacuation de l'ensemble des produits dans un état physique qui respecte celui qui aurait dû être atteint sans défaillance. Une fois les produits évacués, le problème de conception de lois de commande est de nouveau un de ceux vus précédemment. Les objectifs Ob_1, x sont ceux imposés par la demande en cours au moment de la défaillance. L'objectif Ob_2 est l'absence de produit dans le système de production. Et enfin, l'objectif Ob_3 contraint l'état final des chaînes fonctionnelles.

La troisième décision est celle où l'état dit de reprise défini par la fonction *Décider* vise à appliquer une loi de commande, nouvelle ou existante, dont le marquage initial tient compte des produits présents. La reprise va alors consister à placer les chaînes fonctionnelles et les produits dans un état particulier afin que cet état concorde avec un des marquages de la loi de commande appliquée ensuite. Pour cette décision de reprise, les objectifs n'ont aucune particularité et seront calculés par la fonction *Décider*.

Suite à cette clarification des situations de reprise considérées et à la demande correspondante, différentes stratégies de conception d'une loi de commande à des fins de reprise sont exposées dans la section suivante.

3.2 Stratégies de reprise

Notre ambition dans ce manuscrit n'est pas d'étudier dans le détail un algorithme de synthèse de loi de commande à des fins de reprise, mais nous avons cependant à cœur de donner des pistes solides à explorer pour les travaux futurs. Nous limiterons nos propos à deux stratégies de conception de lois de commande répondant à une demande de la forme vue au paragraphe précédent. Sont exclues de ces stratégies celles qui consisteraient à construire l'espace d'états atteignables en considérant l'ensemble des produits présents. Celles-ci vont en effet à l'encontre des principes énoncés dans le chapitre 7. Cependant, elles ne doivent pas être oubliées car elles pourraient être parfois les seules stratégies aboutissant à une solution.

Ainsi, afin de toujours respecter les principes énoncés au chapitre 7, les chemins construits vont se focaliser sur un produit. La stratégie concerne alors la prise en compte de l'état initial avec plusieurs produits présents et les contraintes qu'ils engendrent sur les opérations applicables.

La première stratégie est de prendre en compte ces contraintes lors de la construction des chemins. Dans la seconde stratégie, ces contraintes seront considérées lors de l'imbrication des chemins.

La première stratégie consiste à tenir compte de tous les produits présents lors de la construction d'un chemin $Ch_{x,Op}$ pour un produit x. Ainsi depuis l'état initial $q_{3,0}$, il est recherché un produit pour lequel la fonction Γ renvoie un résultat pour les objectifs Ob_1, x et Ob_2, x liés à un produit x. Afin de se focaliser uniquement sur le produit x, la condition restrictive utilisée pour la génération des espaces restreints d'états atteignables limitera les opérations autorisées à celles ayant un effet sur le produit x.

Depuis le dernier état du chemin $Ch_{x,Op}$ renvoyé par la fonction Γ, le processus est appliqué de nouveau jusqu'à ce que tous les produits aient atteint leurs objectifs. Ainsi les objectifs $Ob_1 = \{Ob_{1,x} \mid x \in [1, P]\}$ et $Ob_2 = \{Ob_{2,x} \mid x \in [1, P]\}$ seront satisfaits. Un dernier chemin est généré afin d'atteindre l'état final des chaînes fonctionnelles fixé par l'objectif Ob_3. L'imbrication des $P + 1$ chemins Ch_{Op} est ensuite envisageable.

Cette stratégie, qui consiste depuis un état à balayer l'ensemble des produits jusqu'à en trouver un pour lequel la fonction Γ renvoie un résultat, risque pour un grand nombre

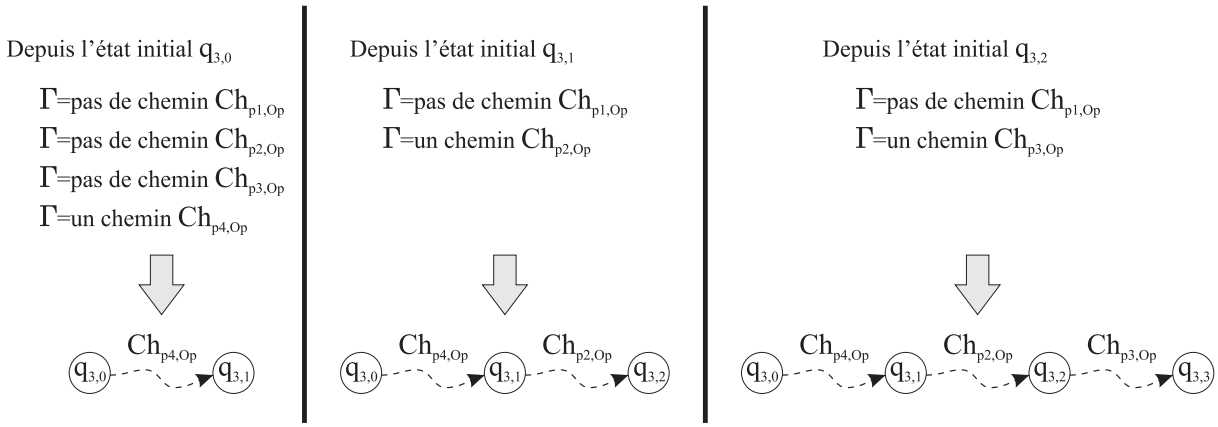


FIG. 9.17 – Première stratégie de conception d'une loi de commande à des fins de reprise.

de produits d'accroître considérablement le temps de conception. En effet, un nombre important d'espaces restreints d'états atteignables peut être construit sans jamais aboutir à un état satisfaisant l'objectif. La deuxième stratégie envisagée contourne ce problème.

La deuxième stratégie requiert un et un seul appel à la fonction Γ pour chacun des produits. Pour cela à partir de l'état initial $q_{3,0}$, il est construit x états initiaux $q_{3,x,0}$. Un état initial $q_{3,x,0}$ est le résultat de la suppression de tous les produits sauf du produit x de l'état $q_{3,0}$.

Pour chacun des produits, il est ainsi généré un chemin $Ch_{x,Op}$ qui est le résultat de la fonction $\Gamma(q_{(1,x,0)}; Ob_{(1,x)}; q_{(3,x,0)}; Ob_{(2,x)}; Ob_{(3,x)})$. L'état $q_{(1,x,0)}$ est l'état $q_{(3,x,0)}$ restreint aux variables d'état physique du produit. Quant à l'objectif $Ob_{(3,x)}$, il ne spécifie aucun état particulier des chaînes fonctionnelles.

Les P chemins $Ch_{x,Op}$ ne tiennent pas compte pour le moment des contraintes résultantes de la présence d'autres produits que le produit x. Pour cela, nous recherchons le chemin dont la séquence d'opérations est applicable depuis l'état $q_{3,0}$. Depuis l'état atteint suite à cette première séquence d'actions, le même processus est appliqué jusqu' à avoir utilisé une fois tous les chemins $Ch_{x,Op}$. Il reste ensuite à générer un dernier chemin afin d'atteindre l'objectif Ob_3 imposé par la demande. L'imbrication des $P+1$ chemins $Ch_{x,Op}$ est là aussi envisageable.

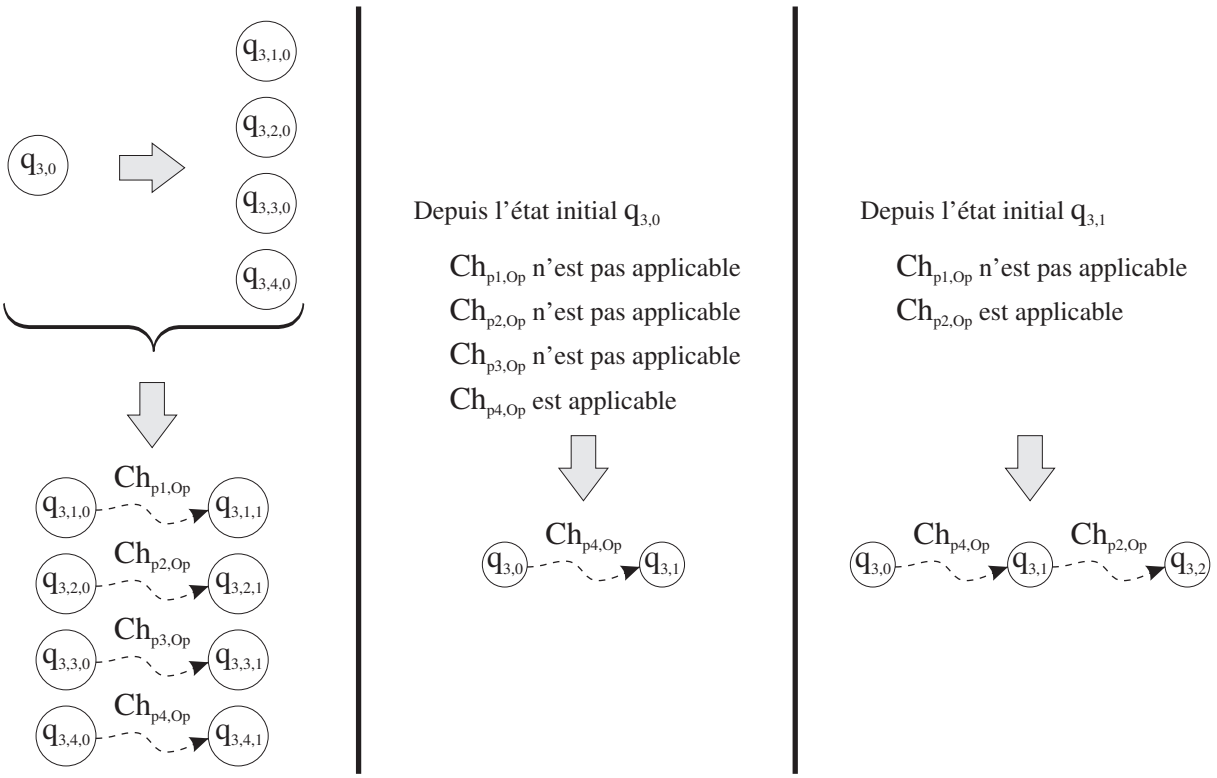


FIG. 9.18 – Deuxième stratégie de conception d'une loi de commande à des fins de reprise.

4 Conclusion

Ce dernier chapitre, consacré à l'algorithme de synthèse, était centré sur la conception de lois de commande destinées à piloter un ou plusieurs flux de produits. L'imbrication de deux séquences, basée sur les mécanismes d'insertion et de parallélisation, a d'abord été développée dans le but d'aboutir à une loi de commande cyclique. Puis, ce mécanisme a été exploité dans un contexte multi-flux de produits. Ainsi, sur la base de la conception de lois de commande acycliques (cf. chapitre 8) et cycliques, la problématique multi-flux de produits a été traitée à travers l'utilisation de ces résultats qui conduit à rechercher des stratégies d'imbrication de séquences. Ces imbrications visent à optimiser le temps de cycle des lois de commande synthétisées sans une représentation de toutes les évolutions du système contrôlé.

Le compromis entre complexité et performance, qui a été recherché tout au long de cette partie, est mis en exergue sur un cas d'étude dans la partie suivante. L'approche dans son ensemble est appliquée à ce cas d'étude, de la modélisation à la synthèse automatique de lois de commande.

Quatrième partie

Application

Chapitre 10

Présentation de la plate-forme de recherche SAPHIR

1 Introduction

Dans cette partie, nous avons souhaité développer un exemple d'application de notre approche de modélisation et de synthèse de lois de commande sur la base du procédé pilote SAPHIR (cf. Figure 10.3 page 174) du Laboratoire d'Automatique de Grenoble. Ce choix nous a semblé judicieux pour plusieurs raisons. Premièrement, cette plate-forme de test et de validation est exclusivement dédiée à la recherche. Deuxièmement, elle est localisée dans le laboratoire où nous avons mené nos travaux de recherche. Elle est donc facilement accessible. Troisièmement, et comme nous allons le voir par la suite, SAPHIR se prête particulièrement bien à l'étude des problèmes liés à la configuration et à la reconfiguration du système de commande.

Dans le cadre de ce premier chapitre, nous allons donc plus particulièrement nous intéresser à la présentation de la plate-forme de recherche SAPHIR. La première section s'attachera à donner les caractéristiques techniques essentielles de la plate-forme. Il s'agira notamment d'en présenter le cahier des charges général, la partie opérative, la partie commande et enfin l'architecture opérationnelle retenue dans le cadre de cet exemple d'application. Après quoi, nous présenterons l'adéquation de SAPHIR à la validation d'approches de synthèse de lois de commande.

2 Caractéristiques techniques

2.1 Cahier des charges

D'un point de vue technique, la plate-forme de recherche SAPHIR est dédiée à l'assemblage d'arbres à cames (cf. Figure 10.1).

Le magasin rotatif de huit emplacements peut accueillir six types de pièces différentes. Ces pièces sont identifiées par pesage. Suite à cette identification, elles sont dirigées vers un portique de tri via un convoyeur central. Selon le type de pièce identifié, le portique oriente la pièce vers les convoyeurs gauche ou droit qui font également office de stocks intermédiaires. En sortie de ces deux convoyeurs, deux postes de positionnement permettent d'indexer les pièces afin que le robot manipulateur puisse les prendre convenablement. Ce dernier peut réaliser en parallèle huit

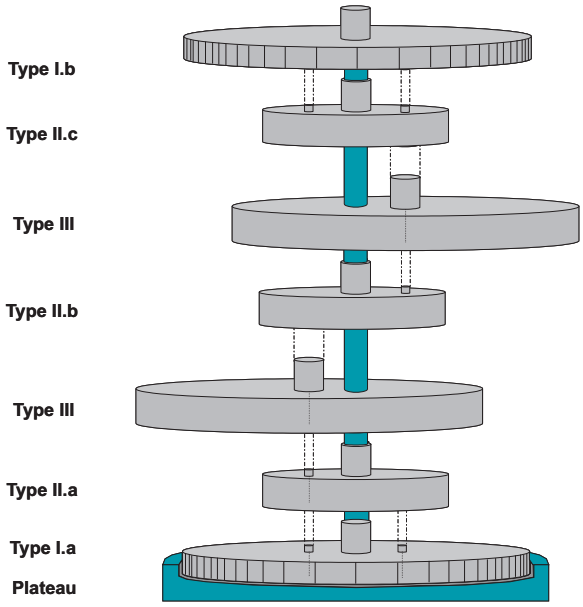


FIG. 10.1 – Exemple d'arbre à câmes

arbres à cames par superposition des pièces. Un opérateur humain est chargé d'approvisionner le magasin de pièces ; un autre doit vider le poste d'assemblage rotatif. Enfin, une caméra sert à la surveillance du procédé, notamment pour les phases d'affinement de diagnostic par des techniques de reconnaissance de forme par exemple.

2.2 La partie opérative

La partie opérative de SAPHIR est constituée de huit éléments (cf. Figure 10.2).

Cependant, dans le cadre de cet exemple, nous nous intéresserons en particulier à la zone d'approvisionnement de pièces du convoyeur central. Aussi, et pour des raisons de concision, nous limiterons cette présentation technique au magasin rotatif, au poste de pesée et au système transitique. Les caractéristiques des autres composants de la plate-forme peuvent être trouvées dans (Zamai, 2001).

2.2.1 Le magasin rotatif

Le magasin de pièces est réalisé sur la base d'un plateau de 300mm de diamètre. Chacun des huit emplacements peut accueillir indifféremment tous les types de pièces (Figure 10.1) grâce aux deux gabarits, l'un de largeur 33mm, l'autre de 66mm. D'un point de vue motorisation, un moteur pas à pas 100/200 pas par tour assure la mise en rotation du magasin (dans les deux sens) via une démultiplication par courroie crantée de rapport 6,6. Une précision de positionnement de 0,7mm est ainsi atteinte. Un capteur inductif assure l'indexation du plateau, et un capteur à réflexion, placé en A (cf. Figure 10.3), permet de détecter la présence de pièce sur les emplacements prévus à cet effet. Un vérin pneumatique double effet doté de deux capteurs magnétiques assure la poussée des pièces sur le poste de pesée.

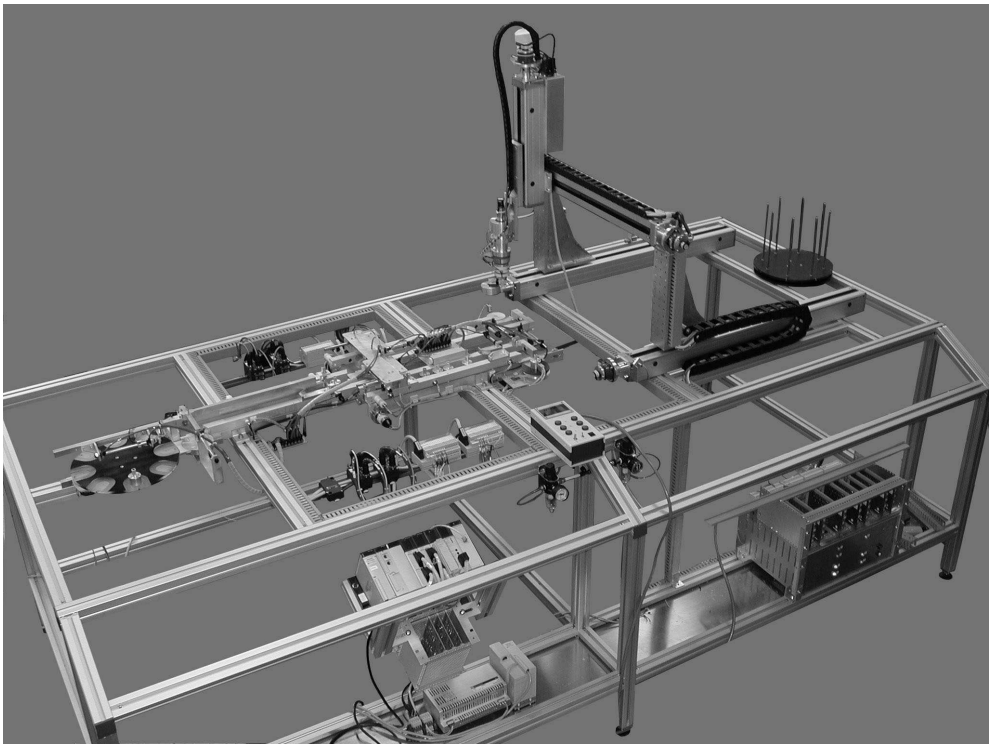


FIG. 10.2 – Partie opérative de la plate-forme SAPHIR

2.2.2 Le poste de pesée

Le poste de pesée est constitué d'une jauge de contrainte permettant la pesée de pièces allant de 20g à 5Kg. Afin d'assurer la poussée de la pièce pesée sur le convoyeur central, un vérin pneumatique double effet équipé de deux capteurs magnétiques a été mis à disposition.

2.2.3 Le système de transitique

Le système de transitique est constitué de trois convoyeurs à bande (un central, deux latéraux). Une courroie crantée industrielle de longueur 65cm assure le convoyage des pièces. Elle est entraînée par un moteur à courant continu équipé d'un réducteur. Ce moteur est commandé via une carte de commande autorisant à la fois des changements de sens de rotation mais également des modifications de la vitesse de rotation. La vitesse *max* a été fixée à 10cm/s. De plus, cette carte interdit tout changement de sens de rotation sans passer par une vitesse nulle ; ceci afin de préserver les engrenages du réducteur. D'un point de vue captage d'informations, quatre capteurs ont été requis. Le premier atteste de la présence d'une pièce en entrée des convoyeurs. Le deuxième témoigne d'une saturation du système de transitique. Le troisième permet de détecter la présence d'une pièce en face de l'ancrage limitant ainsi le nombre de pièces en phase de tri ou de positionnement. Le dernier capteur indique la présence d'une pièce en fin de convoyage. Les ancrages sont quant à eux réalisés au moyen de vérins double effets. Afin de détecter les pièces qui doivent être rebutées, deux capteurs photo-barrages ont été placés sur les convoyeurs latéraux.

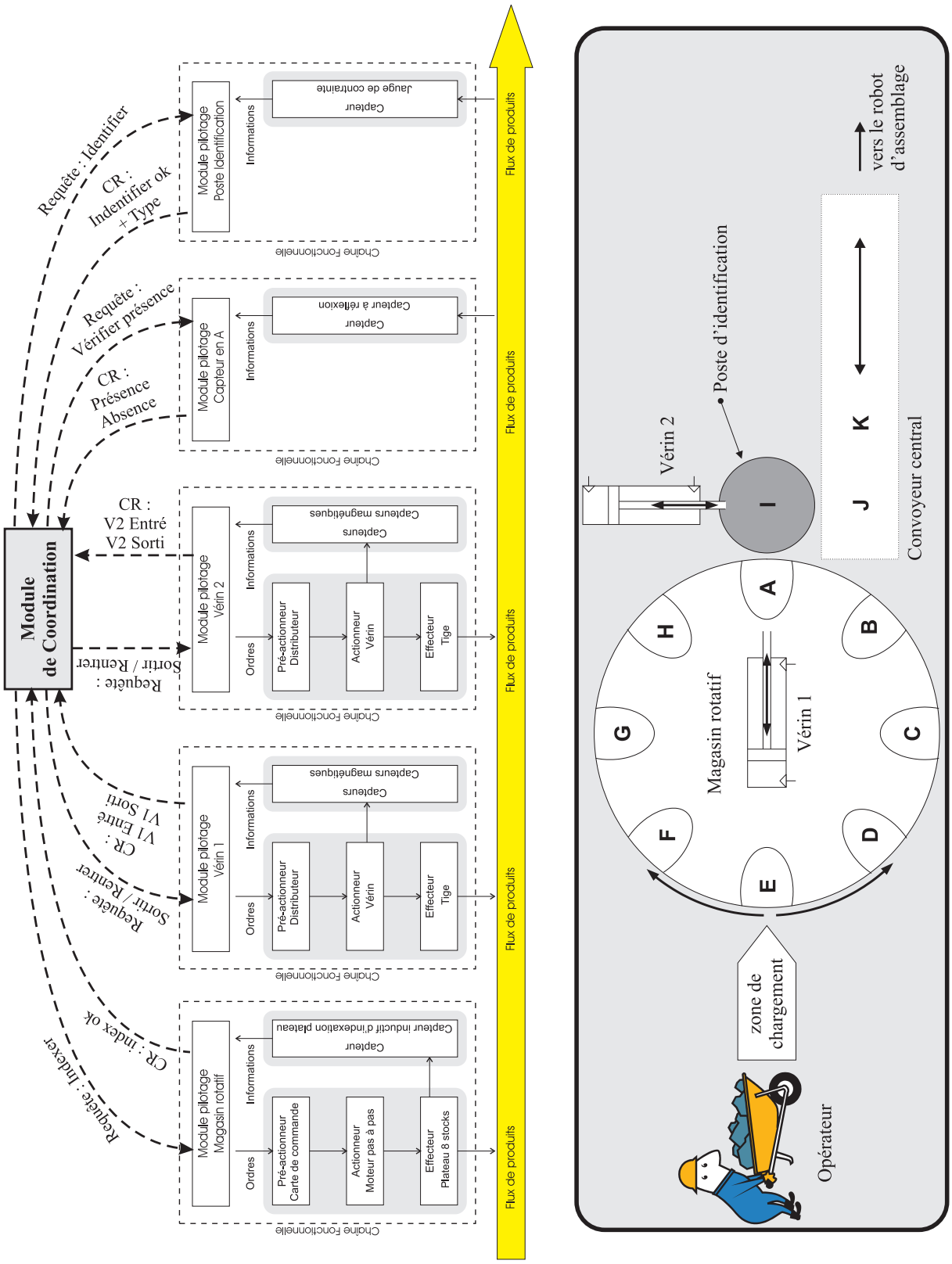


FIG. 10.3 – Architecture opérationnelle du système d'approvisionnement de la plate-forme SAPHIR

2.3 L'architecture du système de pilotage

Les modules de coordination de ce procédé pilote coordonnent quatre zones :

1. le magasin rotatif, le système de pesée et le dispositif d'évacuation de pièces,
2. les systèmes de transitique, de tri et de positionnement,
3. le robot et le poste d'assemblage,
4. et enfin la caméra de vidéo surveillance.

Ils sont chacun implantés, avec leurs chaîne fonctionnelles respectives sur quatre cibles de type PC (VxWorks et LINUX) et automate programmable (TSX PREMIUM).

Dans le cadre de cet exemple d'application, nous étudierons en particulier le module de coordination des chaînes fonctionnelles pilotant le magasin, le système de pesée et d'évacuation. Comme nous le verrons par la suite, ce cas est suffisamment riche pour ne laisser aucun doute quant à la validité de l'approche, et à son applicabilité aux autres modules de coordination.

D'un point de vue technique, cette branche du système global de pilotage est implantée sur un PC Pentium I équipé de VxWorks, d'un joueur de réseau de Petri expérimental développé au LAG dans le cadre de la préparation du mémoire CNAM de Mme Elisabeth Chuiton, et de cinq chaînes fonctionnelles compilées et intégrées au noyau VxWorks. Ces cinq chaînes fonctionnelles (cf. Figure 10.3) respectent naturellement le principe de communication RPC que nous avons introduit dans la partie I de ce manuscrit.

Afin de connecter ce PC cible avec la partie opérative qu'il doit commander, une carte d'acquisition analogique lui a été insérée. En sus de l'acquisition du signal analogique issu de la jauge de contrainte, cette dernière met également à disposition seize entrées et sorties TOR permettant la commande des deux vérins en fonction des signaux émis par les quatre capteurs magnétiques. Une carte de communication 3COM/Etherlink III permet la communication Ethernet/TCP-IP avec les autres modules de coordination de l'architecture.

3 Adéquation à l'étude de la synthèse de lois de commande

Avant d'évaluer notre approche de modélisation et de synthèse de lois de commande, il est nécessaire de s'assurer que la plate-forme SAPHIR se prête bien à la validation des caractéristiques inhérentes à la configuration et à la reconfiguration de la partie logicielle des systèmes de commande. Ainsi, nous nous proposons, pour chacune des caractéristiques fondamentales qui ferme le contexte de notre étude, de montrer que la partie de SAPHIR considérée dans ce document est pertinente.

3.1 Classes d'opérations

Comme nous avons pu le voir dans la partie III l'algorithme de synthèse de lois de commande s'appuie sur une méthode de décomposition de sa perception du système contrôlé en trois grandes catégories d'opérations : celles de transformation du produit, celles de transitique, et celles de préparation. La zone de la partie opérative à laquelle nous souhaitons appliquer notre démarche offre des services de préparation (rentrées de vérins, initialisation du magasin en position indexée, etc...), de transitique (sortir vérins afin de pousser un produit du magasin sur le poste de pesée, ou bien du poste de pesée vers le convoyeur central), et de transformation (identification d'un produit).

3.2 Parallélisme d'exécution

Un des points fondamentaux amené par notre approche réside dans sa capacité à générer une séquence de commande optimale par rapport un critère (dans notre exemple, le temps de cycle). Si une minimisation d'un tel critère est envisageable, c'est que le système contrôlé intègre plus ou moins de parallélismes d'exécution, que ce soit entre opérations de transitique, entre opérations de transformation et/ou encore entre opérations de préparation. C'est justement le cas de notre objet d'étude qui présente des parallélismes entre le pesage et la rotation du magasin, entre la rentrée du vérin 1 et le pesage, etc....

3.3 Opérations simultanées

Dans la partie III, nous avons montré que certaines parties opératives offraient des opérations simultanées sur plusieurs produits (de même type ou différents). En adoptant un point de vue produit, cela pose le problème de l'affectation de l'opérations simultanée aux produits prêts à y accéder. Une fois que l'opération simultanée a été attribuée à un produit, elle doit pouvoir continuer à être affectée aux autres, sous réserve bien entendu de la capacité restante. En ordonnancement, il s'agit du problème de ressources cumulatives à capacités limitées.

Aussi, disposer de telles parties opératives requiert, pour la synthèse de lois de commande, des aptitudes de prise en compte globale de la problématique d'affectation d'un service à plusieurs produits. Ceci est la condition sine qua non pour éviter des aberrations du type "monopoliser une opération simultanée pour un seul produit". C'est bien l'une des qualité de notre approche de synthèse que nous voulons évaluer.

Notre objet d'étude présente également une telle caractéristique, en particulier au niveau de son magasin rotatif constitué de huit emplacements de stockage. Lorsque ce dernier est mis en rotation, il s'en suit une opération de rotation simultanée pour tout produit déjà présent dans un emplacement de stockage.

3.4 Concurrence entre opérations

Générer une séquence de commande revient souvent à résoudre des problèmes de concurrence entre opérations exécutables sur plusieurs produits différents. Face à de telles situations, un problème de choix est révélé. Comme nous l'avons montré dans la partie III, deux techniques de résolution sont envisageables, construire d'une part toutes les solutions pour en choisir une sur la base d'un ensemble de critères, soit envisager une heuristique. C'est ce dernier cas que nous avons retenu dans le cadre de notre approche. Elle s'exprime sous la forme de la règle suivante : "privilégier l'opération pour laquelle le produit concerné est le plus en retard dans la gamme de fabrication".

Il est donc très important de confronter notre algorithme à ce type de problème finalement assez répandu dans le secteur manufacturier. Ce type de situation se retrouve effectivement illustré sur SAPHIR, notamment entre l'opération de rotation du magasin et l'évolution requise de dépose d'un produit dans le magasin par l'opérateur humain.

3.5 Multi-produits

Générer une séquence de commande associée à un seul produit ne présente pas forcément de difficultés majeures selon la nature du système de contrôlé considéré. Toute la séquence est bâtie dans un seul objectif, amener le produit d'un état initial à un état final souhaité. Cependant, comme nous l'avons montré dans la partie III, lorsqu'il s'agit de considérer une production multi-produits, deux problèmes majeurs sont à considérer : la commutation d'un type de production à un autre qui conduit à l'intégration de séquences de préparation des ressources et l'imbrication de n séquences propres à n produits à des fins d'optimisation.

Comme nous avons pu le voir dans le paragraphe de présentation technique de la plate-forme, SAPHIR est nativement construite autour d'une problématique multi-produits particulièrement mise en évidence au niveau du magasin de pièces et du robot d'assemblage.

3.6 Assemblage

Il est très rare, dans un contexte manufacturier, de ne pas être confronté au problème classique de l'assemblage. Cette problématique, bien que d'apparence assez simple, pose de réels problèmes lorsqu'il s'agit de synthétiser automatiquement une séquence d'assemblage. En effet, deux problèmes sont au moins à résoudre : tout d'abord celui de la synchronisation des produits requis pour l'assemblage et ensuite celui du passage de n produits considérés jusqu'au déclenchement de l'opération d'assemblage, à un seul après exécution de cette dernière.

SAPHIR offre une telle opportunité grâce à son robot d'assemblage, mais ce dernier ne fait pas parti du cas d'étude retenu. Cependant, il faut remarquer que le problème de l'assemblage peut être rapproché du cas multi-produit discuté dans la section précédente.

3.7 Reconfigurabilité

Bien évidement, comme annoncé dans le titre même de cette thèse, nous tentons ici d'apporter une brique supplémentaire à l'édifice complexe des architectures de pilotage réactif, en particulier sur le plan de la reconfiguration. Comme nous l'avons vu, les processus de reconfiguration sont complexes, ne serait-ce que sur le plan stratégique (Mendez, 2002). Nous n'allons pas chercher ici à apporter **La** solution à tous ces problèmes, mais tenter, de contribuer à l'une des phases du processus de reconfiguration. Cette phase consistera, à partir d'un modèle du système contrôlé, réactualisé suite à l'occurrence d'une défaillance, à évaluer si une séquence de commande, répondant encore à la demande initiale, est encore disponible.

Nous nous proposons donc de vérifier si le cas d'étude retenu permet d'illustrer cette situation.

Compte tenu des statistiques annoncées dans la partie I et extraites de (Sourise et Boudillon, 1997), mettant en exergue la fragilité des capteurs, nous proposons d'imaginer un sénario réaliste pour lequel le capteur permettant de vérifier si un produit est présent en **A** (produit prêt donc à être poussé sur le poste de pesée), est considéré hors service. Si tel est le cas, il est facile pour un expert de constater qu'il est toujours possible de commander le système en s'appuyant sur la capacité du poste de pesée à détecter également la présence d'un produit. La loi de commande résultante demanderait donc systématiquement à sortir le vérin 1, en aveugle, à chaque indexation du magasin.

Nous voyons bien la encore que le cas d'étude retenu permet d'évaluer notre approche sur ses capacités à générer des séquences de commande, même en mode dégradé, s'appuyant sur les services encore offerts par le système contrôlé.

En tout état de cause, bien que la partie de SAPHIR que nous avons retenue dans le cadre de cette validation par expérimentation ne permette pas d'illustrer entièrement la problématique de l'assemblage, elle couvre cependant tous les autres cas.

4 Conclusion

Dans ce chapitre nous avons présenté la plate-forme de recherche SAPHIR du Laboratoire d'Automatique de Grenoble sur laquelle nous allons évaluer notre technique de modélisation et de synthèse de lois de commande. Afin de mieux préparer la phase de recherche d'adéquation de SAPHIR à l'étude de la problématique de la configuration et de la reconfiguration, nous avons détaillé ses caractéristiques techniques à la fois sous les angles partie opérative et architecture de pilotage.

Compte tenu de la complexité de l'ensemble du procédé SAPHIR et de l'objectif visé, la validation de notre approche, nous n'avons pas envisagé dans le cadre de ce mémoire de l'appliquer sur tout le procédé ; seule la partie opérative organisée autour du magasin, du système de pesée et d'évacuation sera sélectionnée pour la démonstration. C'est ce que nous nous proposons de faire dans les deux chapitres suivants.

Chapitre 11

Modélisation du Système d'Approvisionnement

1 Introduction

Dans ce chapitre nous nous proposons d'appliquer la démarche de modélisation des chaînes fonctionnelles basées sur le formalisme proposée dans la partie II de ce document. Cependant, il est nécessaire de souligner le fait que nous n'avons pas encore pu pousser l'étude au point de demander à un ingénieur méthode, extérieur à notre équipe, d'utiliser notre démarche. Aussi, la qualité des modèles présentés dans le cadre de ce chapitre cache, en partie, notre propre niveau d'expertise. Comme nous le verrons dans les perspectives de ces travaux, il sera nécessaire, à terme, d'évaluer exactement quel est le niveau minimum d'expertise requis pour prendre en main la démarche proposée.

La première section de ce chapitre rappelle non seulement la démarche de modélisation proposée et extrait, du cas d'étude considéré, l'ensemble des opérations offertes par les chaînes fonctionnelles. Ensuite, les trois sections suivantes traitent dans le détail trois classes d'opérations différentes : opération d'action, opération d'information et enfin une évolution requise.

2 Démarche générale

Cette section va s'attacher à mettre en place la démarche de modélisation proposée dans la partie II. Rappelons-la en quelques lignes. Pour chaque module de coordination considéré et chaînes fonctionnelles associées, il s'agit en premier lieu de lister les opérations d'actions, d'information, les évolutions induites et enfin les évolutions requises. Ensuite, pour chacune d'entre elles, la méthode préconise de leur associer un "masque" de modélisation qu'il s'agit ensuite d'instancier.

Appliquée au cas d'étude décrit dans le chapitre précédent, seize opérations (cd. annexe C) sont à considérer :

- douze opérations d'actions (indexer le magasin dans le sen horaire, sortir le vérin 1, etc),
- deux opérations d'information (détecter la présence d'un produit dans le magasin en A, et identifier un produit),
- deux évolutions requises (déposer un produit dans le magasin en E, et évacuer un produit

en sortie du système d'approvisionnement).

Du point de vue de la taille d'une opération, les plus simples comme la rentrée du vérin 1 décrivent uniquement une évolution d'une chaîne fonctionnelle. Les plus complexes comme l'indexation du magasin comportent huit évolutions associées du flux de produits.

Dans le souci de ne pas surcharger inutilement cette partie, nous nous proposons de traiter dans le détail seulement trois de ces opérations (une opération d'action "*indexer dans le sens horaire le magasin*", une opération d'information "*détecter la présence d'un produit dans le magasin en A*", et enfin une évolution requise basée sur l'opérateur "*déposer une pièce dans le magasin en E*"). Le lecteur pourra cependant trouver en annexe C les modèles proposés pour les treize autres opérations.

3 Opération d'action : *Indexer dans le sens horaire le magasin*

Cette opération d'action est basée sur un service offert par le module de pilotage du magasin rotatif. Les grandeurs de commande spécifient à qui demander le service (au module de pilotage du magasin) et quel service demander (Indexer dans le sens horaire). Ce service dont la durée est évaluée à quatre secondes fait évoluer le magasin d'une position indexée à la position indexée suivante dans le sens horaire. Lors de cette évolution, le magasin réalise un $1/8^{ieme}$ de tour. Cette évolution est spécifiée dans le modèle par l'évolution du magasin.

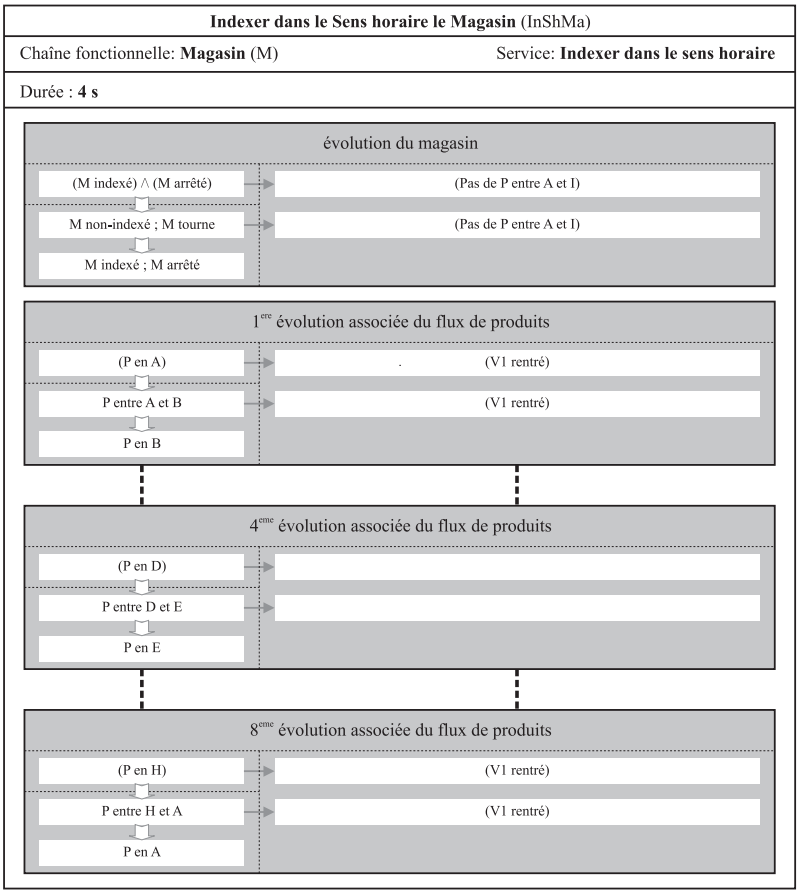


FIG. 11.1 – Opération d'action Indexer dans le sens horaire le magasin.

En fonction du nombre de produits dans le magasin pouvant varier de zéro à huit, il y aura entre zéro et huit produits dont la position évoluera lors de la rotation du magasin. Pour cette raison, l'opération comporte huit évolutions associées du flux de produits décrivant chacune une évolution d'un produit. La Figure 11.1 représente uniquement trois de ces évolutions associées du flux de produits. Après cette présentation des différentes évolutions modélisées, nous allons nous intéresser aux pré-contraintes et aux contraintes.

L'écriture de ces dernières pour l'évolution du magasin doit être faite en ignorant toute évolution associée du flux de produits. Cela revient alors à considérer le magasin comme vide. Dans ce cas, le seul risque à considérer lors de sa rotation est la chute d'une pièce placée entre le magasin et le poste d'identification. La pré-contrainte et la contrainte sont alors spécifiées par la proposition suivante : pas de produit présent entre la position A dans le magasin et la position I sur le poste d'identification. Comme il n'y a pas d'interaction entre le magasin et une autre chaîne fonctionnelle, cette proposition ne porte pas sur l'état d'une chaîne fonctionnelle.

Pour les pré-contraintes et les contraintes des évolutions associées du flux de produits, il est nécessaire de rechercher les interactions d'un produit dans le magasin avec d'une part les autres produits et d'autre part avec les chaînes fonctionnelles. La seule interaction possible est celle avec le vérin 1 qui peut se produire lors de l'évolution d'une pièce à partir des positions H ou A. Pour cette raison, il existe uniquement une pré-contrainte et une contrainte pour les deux évolutions associées du flux de produits dont les conditions sur le flux de produits sont pour l'une la présence d'un produit en A, et pour l'autre la présence d'un produit en H. Pour toutes les autres évolutions associées du flux de produits, il n'y a ni pré-contraintes, ni contraintes.

4 Opération d'information : *Détecter la présence d'un produit*

Le service sur lequel est basée cette opération d'information est offert par le système de pilotage du capteur placé en A. D'après les grandeurs de commande spécifiées (Capteur en A et détecter présence), l'exécution de cette opération nécessite de faire appel au service de détection offert par le système de pilotage du capteur en A. Cette exécution implique toujours une évolution de la chaîne fonctionnelle basée sur ce capteur. Initialement disponible, elle devient indisponible durant toute la durée de l'opération avant de passer de nouveau dans l'état disponible. Quand il y a un produit en A dont la présence est probable, les données accompagnant le compte rendu de fin du service de détection correspondent soit à la présence d'un produit soit à son absence. Du point de vue du niveau de coordination, l'état du produit dans le magasin en A évolue d'un état où sa présence est probable vers un état où elle devient avérée. Cette information étant liée à la réception du compte rendu, l'état du produit n'évolue pas pendant l'exécution de l'opération. Suite à cette description des évolutions spécifiées par l'opération de détection d'un produit dans le magasin, nous étudions maintenant les pré-contraintes et les contraintes associées (cf. Figure 11.2).

La démarche d'écriture des pré-contraintes et des contraintes consiste à écrire d'abord celles de l'évolution de la chaîne d'acquisition puis celle de l'évolution du produit. L'écriture de la pré-contrainte et de la contrainte de l'évolution de la chaîne d'acquisition se fait en supposant que l'exécution de l'opération n'implique pas d'évolution du produit. Ainsi, en l'absence d'un produit sur le magasin en A, le résultat de l'opération doit correspondre à cet état, à savoir, l'absence d'un produit dans le magasin en A. Le capteur considéré détecte une présence dans une zone et non en un point ; il peut alors détecter la présence d'une pièce dans le magasin entre A et B ou

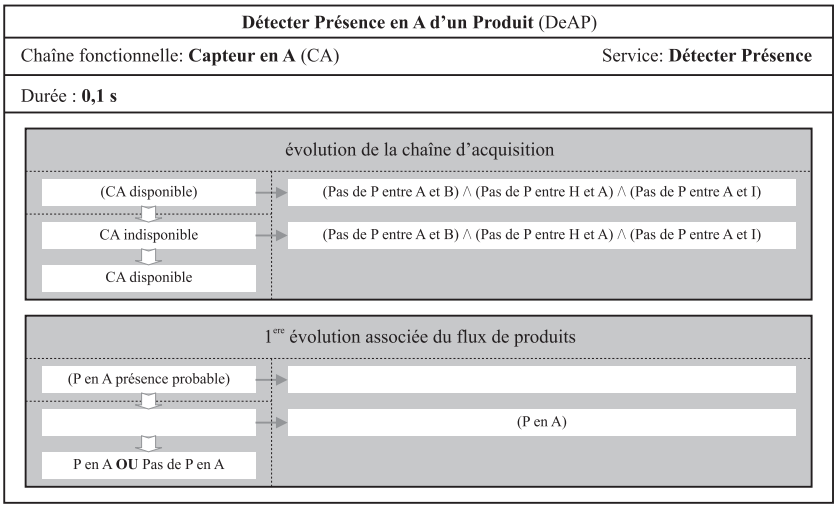


FIG. 11.2 – Opération d'information Détecter présence d'un produit en A.

entre H et A, et de même pour une pièce entre le magasin et le poste d'identification. Le risque est alors de confondre la présence d'une pièce dans le magasin en A avec une des trois positions citées ci-dessus. Ainsi, la pré-contrainte et la contrainte de l'évolution de la chaîne d'acquisition interdisent la présence d'une pièce dans ces trois positions. En supposant connue par le module de coordination l'absence de pièce dans le magasin, il est alors possible d'exécuter un appel au capteur en parallèle d'une rotation du plateau. Ceci eut été impossible si la contrainte et la pré-contrainte imposaient l'indexation du magasin afin d'éviter la présence de pièces entre A et B, et entre H et A.

La pré-contrainte et la contrainte de l'évolution du produit doivent assurer la détection correcte d'un produit dans le magasin en A au risque pour le module de coordination de disposer d'une mauvaise représentation de l'état du flux de produits. Les décisions prises n'étant pas les mêmes en présence et en l'absence d'une pièce, cette désynchronisation entre l'état réel du flux de produits et sa représentation par le module de coordination peut conduire à violer les contraintes de sécurité et d'écologie. Pour cette raison, et afin d'assurer une détection correcte, la pièce initialement dans le magasin en A doit rester dans cette position durant toute la durée de l'opération.

5 Evolution requise : *Déposer un produit dans le magasin en* E

Cette évolution requise approvisionne le magasin en pièces. Ces dernières sont déposées par un opérateur qui ne donne aucune information sur ses actes. Le module de coordination ne dispose alors d'aucune information sur le début ou la fin de cette évolution requise. Cette absence d'information est liée à deux hypothèses, d'une part sur les facultés de l'opérateur à déposer correctement les pièces dans un intervalle de temps donné, et d'autre part sur l'absence de risque aussi bien pour l'opérateur que pour les produits déjà présents dans le magasin. Pour ces raisons, l'évolution de l'opérateur sur laquelle est basée cette évolution requise n'est pas spécifiée. En revanche, l'effet sur l'état du flux de produits de cette évolution est modélisé. Sans elle, aucune pièce n'entrerait dans le système d'approvisionnement de la plate-forme SAPHIR qui ne produirait alors aucun arbre à cames. Mais le module de coordination ne disposant d'aucune

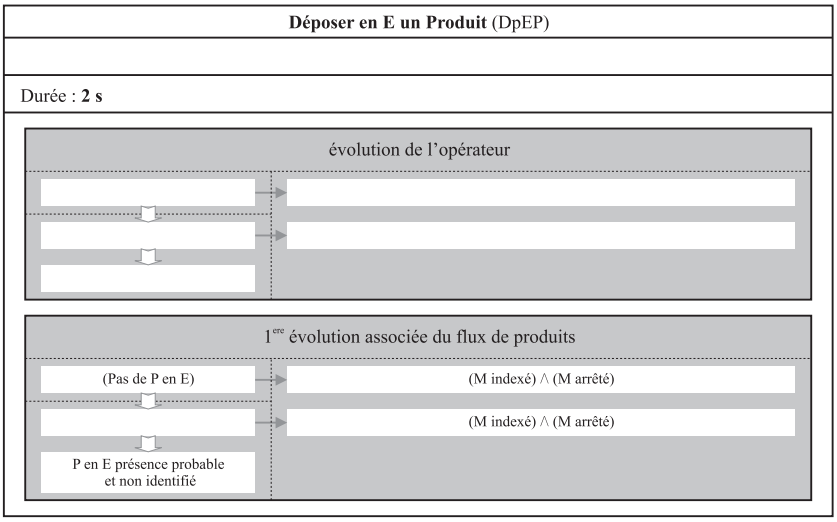


FIG. 11.3 – Évolution requise Déposer un produit en E.

information sur le début ou la fin de cette évolution requise, elle est considérée exécutée chaque fois que le magasin est indexé au minimum pendant une durée égale à celle nécessaire à l'opérateur pour déposer une pièce. Dans ce cas, la présence de la pièce déposée est considérée probable. Avec cette vision, le module de coordination imposera au vérin 1 d'être rentré quand une pièce probable évoluera dans le sens horaire de H vers A. Il est donc très important de modéliser cette incertitude sur l'arrivée des pièces (voir la Figure 11.3). Comme nous avons déjà précisé que le magasin devait être arrêté et indexé pour autoriser cette évolution requise, nous ne détaillerons pas davantage la pré-contrainte et la contrainte.

6 Conclusion

Dans ce chapitre, nous avons présenté non seulement la démarche générale de modélisation que nous proposons ainsi que le détail des modèles à réaliser de trois types d'opérations classiques que tout expert peut être amené à prendre en compte dans un contexte manufacturier. La démarche de modélisation, clairement située au niveau des modules coordination des chaînes fonctionnelles, oriente rapidement l'expert à ne considérer que les opérations offertes par ces dernières, pour ensuite l'amener à décrire leurs interactions. Le formalisme utilisé, volontairement graphique, structuré et intuitif, vise d'une part à améliorer son appropriation et d'autre part à maîtriser la taille du modèle obtenu. En effet, et à titre d'exemple, le lecteur pourra trouver en annexe C.17 une modélisation réseau de Petri du même cas d'étude issu du Master de recherche de M. Eric Deschamps (Deschamps, 2004)(Deschamps et al., 2004). Au delà de la difficulté liée à la maîtrise du formalisme réseau de Petri pour un expert industriel, force est de constater qu'il devient très difficile de "prendre du recul" sur le résultat. Enfin, il est utile de remarquer, que lors d'une éventuelle modification de la partie opérative (introduction ou retrait d'une machine par exemple), le formalisme de "fiches" que nous proposons est certainement plus simple à réactualiser que le modèle réseau de Petri.

Le modèle du système contrôlé considéré étant désormais présenté, nous nous proposons, dans le chapitre suivant, d'évaluer l'approche de synthèse de lois de commande que nous proposons.

Chapitre 12

Synthèse de Lois de Commande

1 Introduction

Basé sur le modèle du système d'approvisionnement présenté dans le chapitre précédent, nous proposons ici de développer deux situations différentes (configuration et reconfiguration) pour lesquelles une loi de commande appropriée devra être générée. Au delà de l'intérêt de ce chapitre de montrer, pas à pas, le comportement des algorithmes de résolution proposés face à un système contrôlé réel (SAPHIR), il est important de noter que les lois générées ont été implantées sur un calculateur de commande et exécutées réellement sur la plate-forme.

Sur le plan organisationnel, ce chapitre est structuré autour de six sections. La première présente la demande générale de configuration du système d'approvisionnement. La seconde s'attache à décrire deux scénarii représentatifs du contexte industriel. Le premier décrit une configuration et son exécution en fonctionnement normal, le deuxième, retrace l'activité du système de pilotage (Surveillance, Supervision, Commande) lors de son passage d'un mode de fonctionnement normal à dégradé. Les sections trois et quatre exposent, dans le détail, le comportement de l'algorithme de synthèse proposé face à ces deux situations. La cinquième section présente les grandes lignes du développement informatique de cette approche de synthèse. Enfin, la section six termine ce chapitre en ouvrant une réflexion sur le lien existant entre les conditions restrictives et la complexité des espaces d'états générés.

2 Description de la Demande

La demande unique qui sera considérée tout au long de cette évaluation spécifie un état initial, un état final et un critère d'optimisation :

un état initial du système contrôlé, $q_{3,0}$, dans lequel le magasin est arrêté, indexé, et vide, tandis que les deux vérins sont rentrés ;

un état d'entrée des produits lors de leur dépose par l'opérateur dans le magasin au point E. L'opérateur ne fournissant aucune information sur ses actions, nous devons alors considérer qu'il peut avoir déposé un produit chaque fois que le magasin est arrêté et indexé pendant une durée équivalente à l'évolution requise associée à l'action de l'opérateur. Sous

cette condition sur la durée d'arrêt du magasin et en l'absence d'un produit en E autorisant la dépose d'un produit par l'opérateur, l'état d'entrée $q_{1,0}$ des produits est défini par un attribut *présence* dont la valeur est probable, et de plus un attribut *identifié* dont la valeur est non. Cette incertitude sur l'état d'entrée des produits conduit à un problème multi-flux de produits. En effet, il faudra définir les opérations pour les produits réellement présents, notés Ppr, et ceux absents du fait de leur présence probable initialement, notés Pab ;

l'état final du système contrôlé, spécifié par les objectifs Ob_2 et Ob_3, est le même que l'état initial du système contrôlé afin de satisfaire la condition de cycle définie au § 2.5 page 159 ;

l'état de sortie des produits est défini par un ensemble d'objectifs $\{Ob_{1,Ppr}, Ob_{1,Pab}\}$ pour les deux types de produits, comme défini pour les problèmes multi-produits au § 3 page 163. Aucune transformation de l'état physique d'un produit absent étant possible, l'objectif $Ob_{1,Pab}$ est vide. Pour les produits présents, l'objectif $Ob_{1,Ppr}$ impose d'avoir identifié les produits. Pour cela, l'attribut *identifié* devra avoir la valeur *oui*.

un critère d'optimisation qui est la minimisation du temps de cycle.

Les deux scénarii basés sur cette demande peuvent maintenant être détaillés.

3 Scénarii d'évaluation

3.1 Scénario basé sur les capacités nominales

Dans ce premier scénario, le système d'approvisionnement dispose de ses capacités nominales. Aussi, son modèle est composé de l'ensemble des opérations et évolutions présentées au chapitre 10.

Basé sur ce modèle, ce premier scénario vise à générer la loi de commande optimale répondant à une requête de configuration issue des niveaux supérieurs de la hiérarchie.

3.2 Scénario de réactivité à une défaillance

Dans ce scénario, nous allons supposer l'occurrence d'une défaillance remettant en cause l'intégrité d'un des services offerts par les chaînes fonctionnelles. Ainsi, suite à sa détection (fonction *détecter*) et sous l'hypothèse de la mise à jour du modèle par les fonctions *Diagnostiquer* et *Suivre*, nous allons étudier la capacité de la méthode de synthèse proposée à regénérer une nouvelle loi de commande répondant encore à la demande initiale. Au delà de l'intérêt réel de l'algorithme à garantir une continuité de service, nous verrons qu'il peut se révéler fort utile, voire même incontournable, pour les fonctions *Décider* et surtout *Pronostiquer*.

La défaillance considérée est celle du service "*détecter la présence d'un produit*" rendu par le capteur placé en A. Pour des raisons de concision, nous supposerons lors de l'occurrence de cette défaillance qu'il n'y a pas de produit dans le système d'approvisionnement.

4 Comportement de l'algorithme de synthèse dans le scénario 1

Rappelons que dans ce premier scénario, le modèle du système d'approvisionnement contient toutes les opérations présentées au chapitre 11.

4.1 Lois de commande sans parallélisme ni insertion

Cette section applique les concepts développés pour concevoir des lois de commande gérant des produits avec des spécifications différentes. Dans un soucis de clarté, ni le parallélisme d'exécution ni l'insertion sont considérés dans cette section.

L'incertitude sur l'état d'entrée des produits pour l'exemple d'application choisi conduit à un problème avec deux types de produits ; produit présent *Ppr* et produit absent *Pab*. Ainsi pour chacun des deux types de produits, une séquence cyclique d'actions est construite à partir des étapes A, B et C présentées au chapitre 9. Ces séquences représenteront deux lois de commande, l'une destinée à gérer des produits présents qui se succèdent, et l'autre qui gère une succession de produits absents. Mais ne choisissant pas le type de produits déposé en E par l'opérateur, un produit absent peut succéder à un produit présent, et inversement. Ainsi, il s'avère nécessaire de rechercher les contraintes de commutation entre ces deux lois de commande (voir § 1.2 page 218). La loi de commande résultante de la fusion avec prise en compte des contraintes de commutation est finalement traduite en réseau de Petri comme le propose l'étape D présentée au chapitre 9.

Cette section s'articule donc autour de quatre points :
- construction d'une séquence d'opérations de transformations, de transitique et de préparation pour chacun des produits ;
- recherche des contraintes afin d'aboutir à une séquence cyclique pour chacun des produits ;
- génération par fusion des séquences cycliques pour chacun des produits ;
- recherche des contraintes de commutation entre les deux séquences cycliques et fusion.

Considérée comme triviale, la traduction en réseau de Petri de la loi de commande ne sera pas détaillée ici. Seul le réseau de Petri obtenu sera donné en annexe.

4.1.1 Construction d'une séquence d'opérations

La construction d'une séquence d'opérations s'appuie sur l'étape A du chapitre 11 faisant appel à la fonction Γ. Cette étape A construit deux chemins définissant chacun la séquence d'opérations nécessaire à atteindre l'objectif pour chacun des deux produits *Ppr* et *Pab*. La non prise en compte, dans cette section, des parallélismes et notamment ceux entre des opérations liées au même produit conduit à une fonction Γ limitée aux étapes 1, 2 et 3 présentées au chapitre 8.

Pour les produits absents, il n'y a pas d'opérations de transformation. Ainsi, le chemin $Ch_{Pab,1}$ résultant de l'étape 1 est vide. Lors de l'étape 2, dédiée aux opérations de transitique, le chemin optimal $Ch_{Pab,2}$ représenté dans la Figure D.1 conduit à la séquence d'opérations de transitique suivante : déposer un produit en E (DepE), puis quatre fois l'appel à l'opération indexer dans le sens horaire le magasin (ShMagasin), et enfin détecter la présence en A d'un produit (DetA) qui renvoie l'information absence d'un produit. Toutes les conditions et les

pré-contraintes sur les ressources étant satisfaites, aucune opération de préparation n'est ajoutée suite à l'étape 3. Ainsi, le chemin Ch_{123} obtenu pour les produits absents suite à l'étape A est celui de la Figure 12.2.

L'étape A appliquée ci-dessus pour un produit absent est ensuite appliquée pour les produits présents. Afin de répondre à l'objectif $Ob_{1,Ppr}$ imposé par la demande, le chemin $Ch_{Ppr,1}$ résultat de l'étape 1 se limite à l'opération identifier produit (IdP).

Les conditions et les pré-contraintes de cette opération nécessitent, afin de satisfaire la demande, une séquence de transitique avant et après IdP. La séquence de transitique avant amène un produit dont la présence est certaine sur le poste d'identification. Cette séquence résulte du chemin optimal trouvé dans l'espace d'états atteignables présenté D.2. La séquence de transitique après l'opération IdP est trouvée dans l'espace d'états de la Figure 12.1. Suite à cette étape 2, il en résulte une séquence d'opérations de transformation et de transitique : déposer un produit en E (DepE), puis quatre fois l'opération indexer dans le sens horaire le magasin (ShMagasin), détecter la présence en A d'un produit (DetA) qui renvoie l'information présence d'un produit, sortir le vérin 1 (SV1), identifier un produit (IdP), sortir vérin 2 (SV2), et enfin évacuer un produit en K (EvaK).

Afin de satisfaire les conditions et les pré-contraintes de l'opération sortir vérin 2, l'étape 3 aboutit à intercaler avant cette opération, l'opération rentrer vérin 1 (RV1). Enfin, pour satisfaire l'objectif Ob_3 imposé par la demande, l'opération rentrer vérin 2 (RV2) est ajoutée à la fin de la séquence. Ainsi, le chemin complet Ch_{123} pour les produits présents suite à l'étape A est celui représenté dans la Figure 12.2 et directement issu de l'atelier logiciel que nous avons développé.

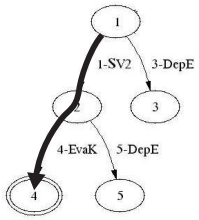


FIG. 12.1 – Espace d'états atteignables par les opérations de transitique suite à l'identification.

L'étape suivante consiste pour chacun des produits à définir les contraintes de manière à rendre cyclique chacun des chemins.

4.1.2 Contraintes liées à la succession de N produits

Les contraintes liées à la succession de N produits présents résultent de l'application de l'étape B du chapitre 9. Mais sans le mécanisme de parallélisation et d'insertion, elle se limite alors à vérifier qu'il est possible d'appliquer la même séquence d'opérations pour un produit, plusieurs fois consécutivement. Étant donné les objectif Ob_2 et Ob_3 qui imposent de revenir dans l'état initial, la condition de cycle est satisfaite. Ainsi, cette étape B se résume à ajouter une contrainte de précédence entre la dernière opération pour un produit présent et la première opération pour le produit présent suivant.

4.1.3 Construction d'une séquence cyclique

L'obtention d'une séquence cyclique pour chacun des produits est basée sur l'étape C du chapitre 9 qui permet de fusionner deux chemins successifs liés à un même produit. Le résultat de cette fusion est représenté par les deux chemins complets Ch_{123} respectivement pour les produits présents et absents (cf. Figure 12.2). Suite à l'ajout du nom de la première opération à la liste *Div* de la dernière action, les chemins complets sont cycliques (la dernière action "pointe" sur la première).

Suite à cette étape C, deux lois de commande sont disponibles ; une, destinée à piloter le système d'approvisionnement pour N produits présents qui se suivent, et l'autre pour les produits absents. Cependant, ne maîtrisant pas l'ordre d'arrivée des produits, absents ou présents, il reste à prendre en compte l'alternance entre ces deux types de produits. C'est ce que nous nous proposons de présenter dans la section suivante.

4.1.4 Contraintes de commutation et fusion

Le problème de la commutation entre les deux chemins Ch_{123} est très simple ici de par l'état initial de chacun des chemins, satisfaisant la condition de cycle, qui est le même. La commutation entre les deux types de produits se résume à une contrainte de précédence de manière à attendre que la séquence liée à un type de produit soit terminée avant d'exécuter la séquence liée à l'autre type de produits. La commutation d'un produit présent à un produit absent induit alors une contrainte de précédence entre le dernier comportement pour le produit présent et le premier comportement pour le produit absent ; et inversement pour la commutation d'un produit absent à un produit présent.

Les comportements identiques pour les deux types de produits sont fusionnés comme le montre la Figure 12.2.

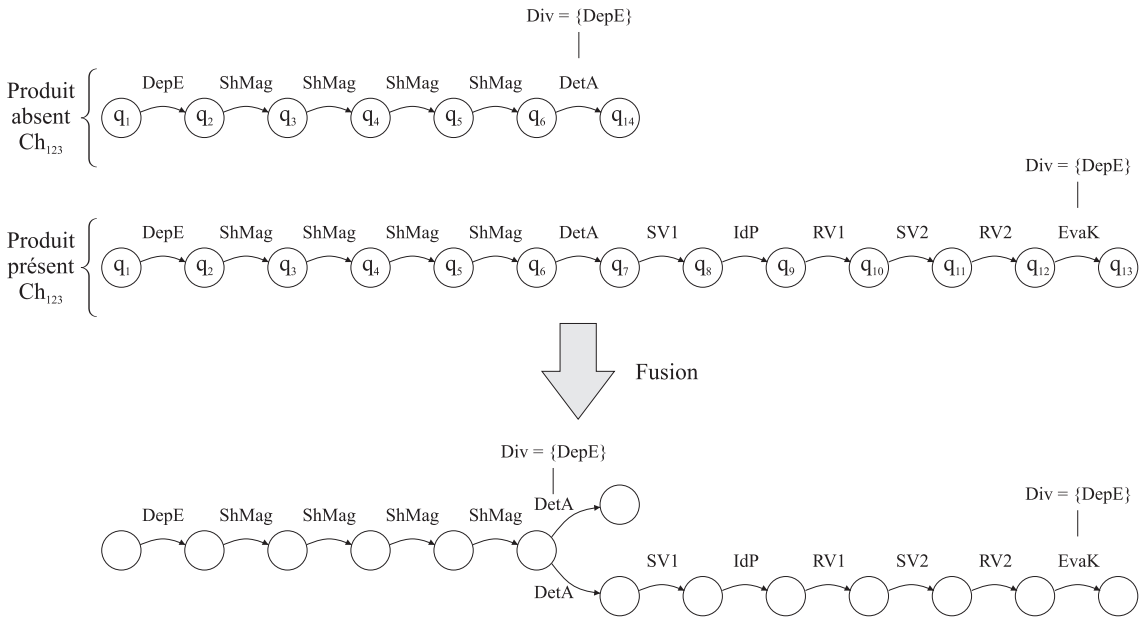


FIG. 12.2 – Fusion des comportements.

La représentation en réseau de Petri de la séquence résultante de la fusion est présentée dans la Figure D.3. Le temps de cycle de cette loi de commande n'est pas optimal car ni le parallélisme d'exécution ni le mécanisme d'insertion n'ont été considérés. Ainsi, la section suivante propose la prise en compte des parallélismes limités aux opérations liées à un même produit.

4.2 Ajout des parallélismes (étape 4)

La prise en compte des parallélismes d'exécution entre des opérations d'une même séquence liée à un même produit a été étudiée dans l'étape 4 du chapitre 8. Elle conduit alors à une étape A composée des étapes 1, 2, 3 et 4.

Dans le chemin Ch_{123} composé d'opérations de transitique et de préparations lié à un produit absent (cf. Figure 12.2), il n'existe pas de parallélisme d'exécution entre les opérations. En revanche, pour un produit présent, il existe un parallélisme d'exécution entre l'opération rentrer vérin 1 (RV1) et l'opération identifier produit (IdP). Ainsi, suite à l'opération sortir vérin 1 (SV1), les opérations RV1 et IdP seront lancées simultanément. La contrainte de précédence entre SV1 et IdP est donnée par le chemin Ch_{123}. En revanche, la nouvelle contrainte de précédence entre SV1 et RV1 conduit à ajouter RV1 à la liste *Div* de SV1, et *IdP* à la liste *//avant* de RV1. Ce parallélisme d'exécution conduit à la séquence d'opérations présentée dans la Figure 12.3.

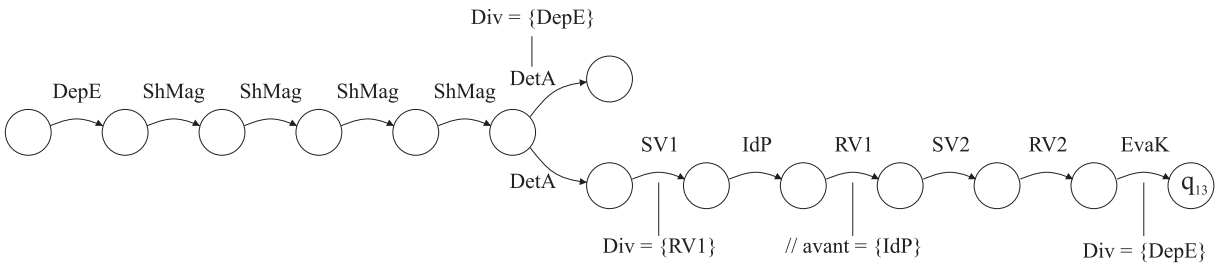


FIG. 12.3 – Prise en compte des parallélismes d'exécution.

Le réseau de Petri représentant cette séquence est donné dans la Figure D.4 de l'annexe D. Celle-ci fait clairement apparaître le parallélisme entre l'opération *rentrer vérin 1* et *identifier produit*. En revanche, il n'existe aucun parallélisme entre des opérations liées à deux produits distincts. L'introduction de ces parallélismes est présentée dans la section suivante.

4.3 Mécanisme de Parallélisation inter-produits (étape B sans insertion)

Nous nous proposons donc maintenant de compléter la séquence jusqu'à présent obtenue en faisant appel au mécanisme de parallélisation qui recherche les parallélismes entre des opérations liées à des produits différents. Ces parallélismes sont à rechercher pour toute la combinatoire de précédence entre produits différents (entre un produit présent suivi d'un autre produit présent, un produit absent suivi d'un autre produit absent, ou encore quand un produit présent succède à un produit absent et inversement). Finalement, pour deux types de produits, le mécanisme de parallélisation sera appliqué quatre fois comme le présente la Figure D.5 de l'annexe D. Dans

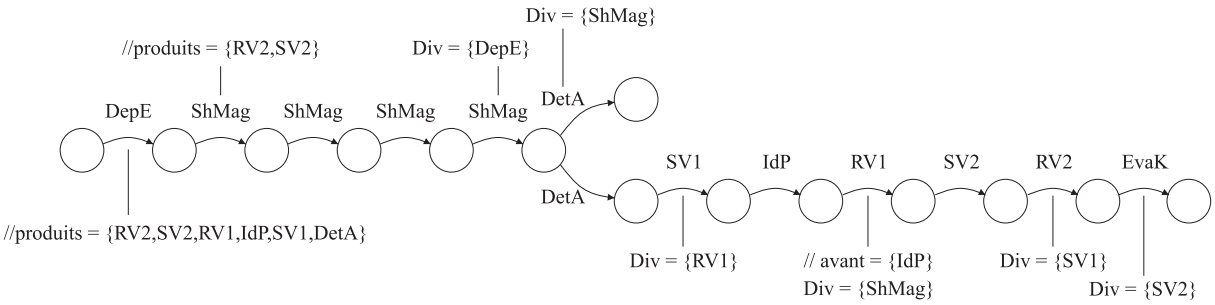


FIG. 12.4 – Parallélisation d'opérations inter-produits.

cette Figure, toutes les listes *Div* sont représentées, en revanche une liste *//produits* qui est identique à celle de l'opération précédente n'est pas représentée.

Le mécanisme de parallélisation va, par exemple, autoriser le parallélisme d'exécution entre l'évolution requise *déposer un produit en E* et l'opération *détecter en A la présence d'un produit*. Ces deux opérations ont chacune un effet sur un produit différent. Suite à la fusion des listes *Div* et *//produits*, il en résulte la séquence donnée dans la Figure 12.4.

La traduction en réseau de Petri de cette séquence est donnée dans la Figure D.6 de l'annexe D. Afin d'appliquer jusqu'au bout l'étape B d'imbrication, la section suivante présente la prise en compte du mécanisme d'insertion.

4.4 Mécanisme d'insertion inter-produits (étape B complète)

Le mécanisme d'insertion vise à réduire la distance, du point de vue opérations réalisés, séparant deux produits successifs. Il intervient lorsque le mécanisme de parallélisation présenté précédemment renvoie un parallélisme impossible entre deux opérations de deux séquences différentes. Par exemple, l'évolution requise *déposer produit en E* (DepE) ne peut pas être exécutée en parallèle de la quatrième opération *indexer le magasin dans le sens horaire* (ShMag4). L'algorithme d'imbrication fait donc appel au mécanisme d'insertion. Celui-ci conduit à insérer DepE avant ShMag4. Le mécanisme de parallélisation appliqué de nouveau renvoie une incompatibilité entre DepE et la troisième opération *indexer dans le sens horaire le magasin* (ShMag3). DepE est alors insérée avant ShMag3. L'imbrication de DepE s'arrête suite à l'impossibilité d'insérer DepE avant la première opération *indexer dans le sens horaire le magasin* (ShMag1). Cette insertion reviendrait à autoriser l'opérateur à déposer deux pièces sur le même emplacement de stockage du magasin. Les conséquences du mécanisme d'insertion sur les listes *Div* et *Fus* sont données dans la Figure D.7 de l'annexe D.

Le mécanisme d'insertion appliqué au problème complet conduit à la séquence représentée dans la Figure 12.5. La séquence de commande donnée dans la Figure 12.5 est finalement traduite et représentée dans la Figure D.8 de l'annexe D sous le formalisme réseau de Petri. Cette loi de commande est le résultat final de l'algorithme de synthèse en réponse à la demande de configuration exposée § 2 page 185.

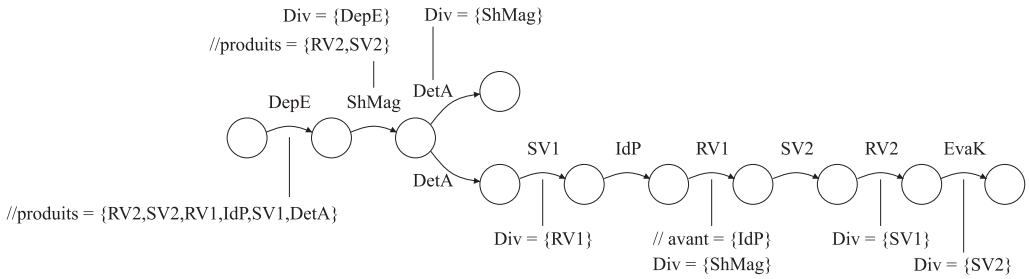


FIG. 12.5 – Application du mécanisme d'insertion.

Examinons maintenant le comportement de l'algorithme de synthèse dans le contexte particulier du scénario 2 pour lequel les capacités opérationnelles du système contrôlé sont réduites.

5 Comportement de l'algorithme de synthèse dans le scénario 2

Ce second scénario, présenté au § 3.2 page 186, considère la défaillance complète du capteur en A et la réactualisation du modèle du système contrôlé (suppression du service *détecter un produit en A*).

Examinons maintenant le résultat de notre algorithme de synthèse appliqué à ce nouveau contexte (même demande, modèle réactualisé), et évaluons sa capacité à trouver effectivement la seule loi de commande capable d'exploiter les capacités de détection du poste de pesée en lieu et place du dispositif de capteur en A désormais hors service.

Comme nous pouvons le constater au travers de la Figure 12.6, la nouvelle séquence obtenue propose bien d'utiliser le poste de pesée à des fins de détection et de pesage. Il s'en suit forcément, comme nous l'avions prévu, une poussée systématique de tout produit susceptible d'être en A. Le comportement global est bien dégradé, mais continu à satisfaire la demande.

La spécification de cette loi de commande en réseau de Petri est donnée dans la Figure D.9 dans l'annexe D.

Au delà de l'intérêt démontré par l'exemple de générer des lois de commande permettant d'assurer une continuité de service en mode dégradé, force est de constater que l'obtention d'une solution ou non peut contribuer fortement à la prise de décision (fonction *Décider*). Ce type d'algorithme peut donc se révéler fortement intéressant pour conclure quant à l'incapacité ou la capacité (Boufaied, 2000) d'un module de coordination à satisfaire encore la demande.

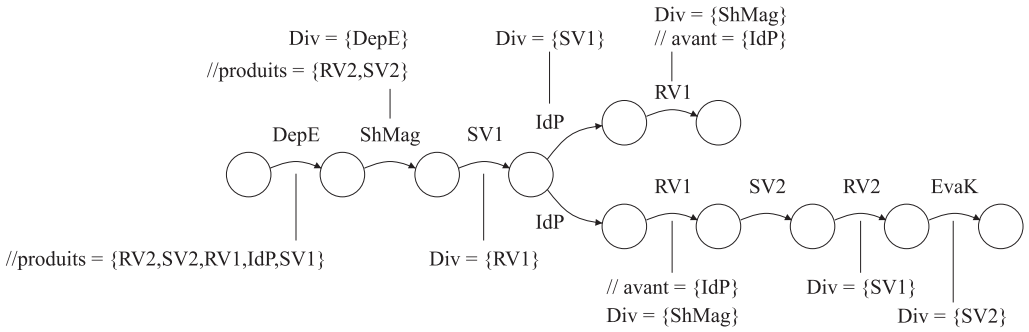


FIG. 12.6 – Nouvelle loi de commande pour le système contrôlé dégradé.

6 Mise en Œuvre Informatique de l'Algorithme

La mise en œuvre informatique, assistée par le service informatique du LAG, de l'approche de synthèse de lois de commande est, à ce jour, quasiment achevée. Il reste à intégrer à l'atelier logiciel d'une part le mécanisme d'insertion et d'autre part le mécanisme de prise en compte de la problèmatique multi-produits présentée au chapitre 11. Pour ces raisons, le résultat de l'atelier logiciel appliqué dans le cadre du scénario 1 est limité aux réseaux de Petri de commande pour les produits absents (cf. Figure D.10) et pour les produits présents (cf. Figure D.11).

Au delà de ces considérations, la Figure 12.7 donne une vision globale de l'architecture de l'atelier logiciel à ce jour réalisé.

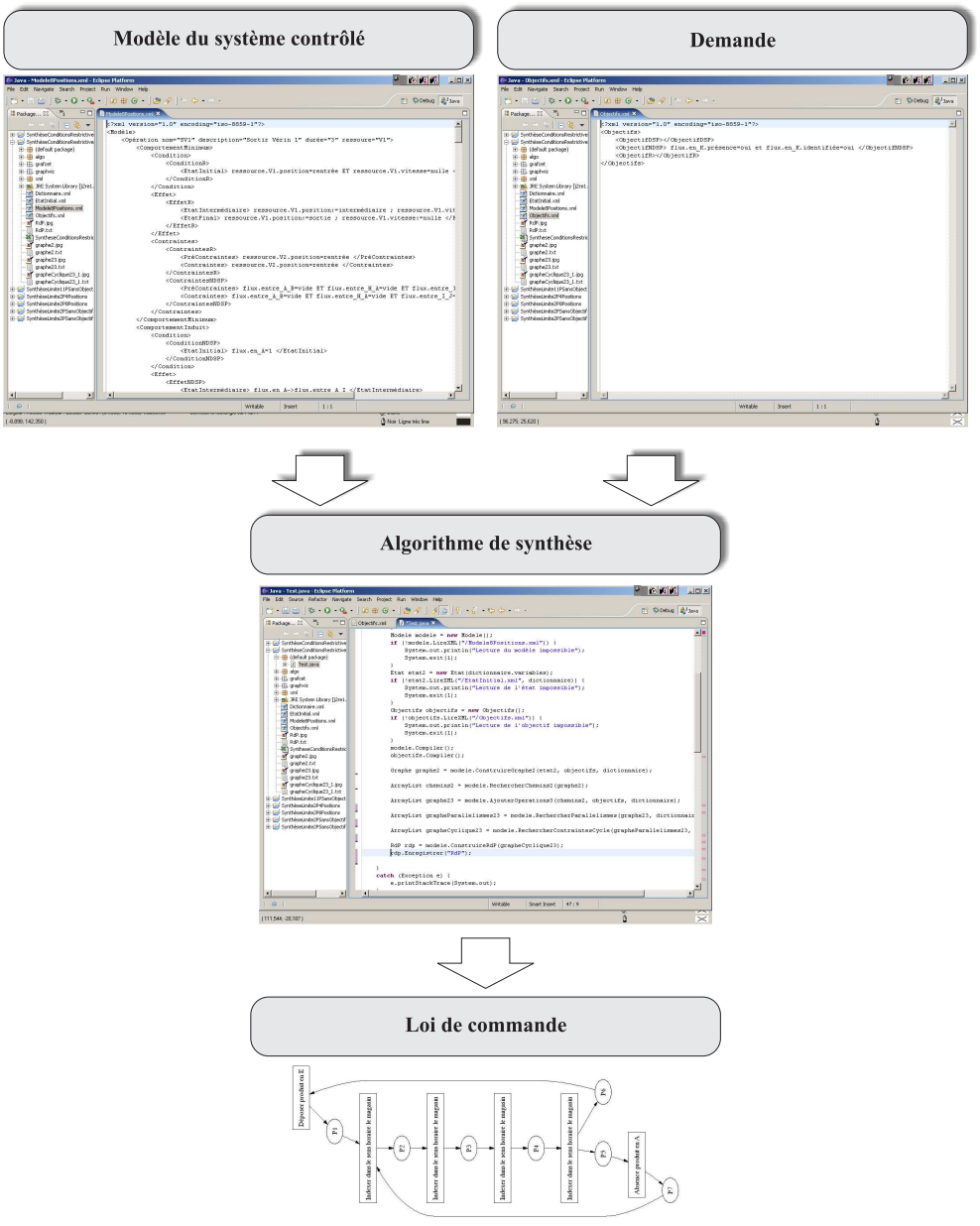


FIG. 12.7 – Atelier logiciel pour la modélisation et la synthèse de lois de commande.

La mise en œuvre, basée sur le Langage Java, afin d'envisager son portage sur des plateformes de type webserveur d'automates par exemple, a été réalisée sur la chaîne de développement Eclipse. Dans un soucis de standardisation, le modèle du système contrôlé ainsi que la demande décomposée en un état initial et un objectif ont été volontairement représentés au format *XML* (cf. Figure D.12).

Le résultat direct de l'algorithme de synthèse mis en œuvre est donné sous la forme d'un fichier texte décrivant l'ensemble des arcs du réseau de Petri. Basé sur cette mise en œuvre, le temps de calcul nécessaire à la génération des deux réseaux de Petri cités ci-dessus, sur un calculateur de type pentium III à 700 MHtz avec 512 Mo de mémoire vive, est d'environ 2 secondes.

Le passage à une représentation graphique a été réalisé grace au logiciel *Dot* développé par E. Koutsofios et S. North (http ://www.graphviz.org/) (Gansner et al., 1993).

Fort de ce développement informatique, nous avons exploité les algorithmes de génération des espaces d'états afin d'évaluer le rapport réalisme/complexité de l'approche proposée. C'est ce que nous nous proposons de discuter dans la dernière section de ce chapitre.

7 Réalisme de la solution proposée

Afin d'évaluer le réalisme de l'approche proposée dans un cadre industriel complexe, nous avons mené deux études expérimentales. L'une vise à montrer l'influence du nombre de chaînes fonctionnelles sur la complexité de l'espace d'états atteignables, et la seconde se focalise sur l'impact de la condition restrictive sur le nombre de produits.

L'impact de la condition restrictive basée sur l'objectif étant trop dépendante de l'objectif, nous n'avons pas étudié son influence.

La première étude a consisté à générer d'abord l'espace d'états atteignables par les opérations de transformation et de transitique pour un produit considéré, sur la base des cinq chaînes fonctionnelles de l'exemple. Puis progressivement, nous avons supprimé le vérin 2, puis le poste d'identification, et ainsi de suite jusqu'à garder uniquement le magasin. Le résultat de cette étude est présenté dans la Figure 12.8. Il montre, pour notre exemple, la relation visiblement linéaire qui existe entre la complexité (définie ici en nombre d'états et de transitions) de l'espace d'états atteignables et le nombre de chaînes fonctionnelles. Dans un contexte industriel, l'ordre de grandeur du nombre de chaînes fonctionnelles pilotées par un module de coordination dépasse rarement la dizaine. Ainsi au vu de ces résultats, à confirmer par une étude théorique de complexité, l'approche semble suffisamment réaliste pour envisager de l'appliquer sur un procédé manufacturier complexe.

Suite à cette première étude, nous avons fait varier la condition restrictive sur le nombre de produits. La première situation correspond à la situation de départ de l'étude précédente, à savoir, les services offerts par les cinq chaînes fonctionnelles avec le nombre de produits limité à un et sans objectif. Le résultat de l'étude illustré dans la Figure 12.9 montre la relation visiblement d'ordre exponentielle entre le nombre de produits et la complexité de l'espace d'états atteignables. Ceci démontre tout l'intérêt de la condition restrictive sur le nombre de produits de manière à maîtriser la complexité des espaces d'états atteignables.

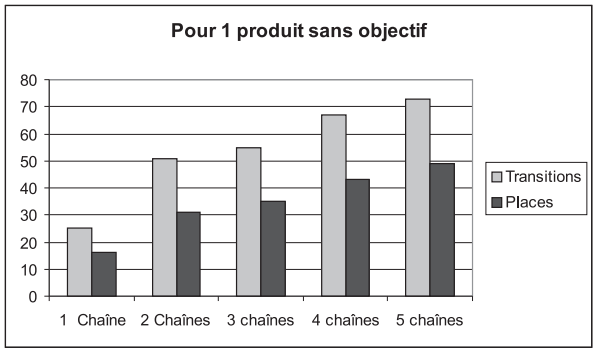


FIG. 12.8 – Nombre d'états et de transitions de l'espace d'états atteignables en fonction du système contrôlé.

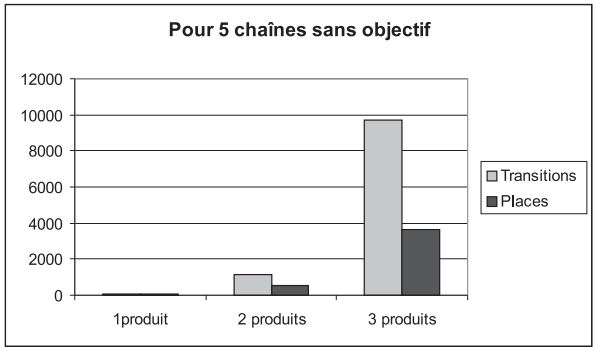


FIG. 12.9 – Nombre d'états et de transitions de l'espace d'états atteignables en fonction de la condition restrictive sur le nombre de produits.

8 Conclusion

L'exemple que nous venons de traiter dans cette partie est basé sur la synthèse de lois de commande du niveau coordination pilotant le système d'approvisionnement de pièces de la plate-forme SAPHIR du Laboratoire d'Automatique de Grenoble.

Il s'est notamment agit de démontrer à la fois la pertinence du formalisme proposé pour appréhender la modélisation de systèmes contrôlés, et les capacités de l'approche de synthèse à générer des lois de commande respectant en tout points les spécifications de production. Dans ce dernier objectif, deux scénarii propices à la synthèse de lois de commande ont été proposés. Le premier nous a permis d'illustrer une situation classique de demande de configuration ; l'autre nous a amené, au sein du processus global de reconfiguration, à synthétiser une nouvelle loi de commande dans un contexte de production dégradée.

Du point de vue modélisation, nous avons pu constater tout l'intérêt du cadre rédactionnel très structuré, intuitif et proche de la vision que peut avoir un ingénieur méthode du système contrôlé. Nous avons cependant souligné le fait qu'il était difficile de valider plus avant la démarche de modélisation proposée, compte tenu du fait que nous l'avons utilisée et appliquée en tant que "créateur". Il est toujours difficile d'être à la fois juge et partie. Nous ne pouvons

donc pas aujourd'hui caractériser précisément le juste niveau d'expertise requis pour prendre en main la démarche et le formalisme proposés. Seule une confrontation directe in situ permettra non seulement de déterminer ce niveau, mais également de critiquer la démarche globale de capitalisation des connaissances. Les aspects vérifications de propriété proposés dans le chapitre 6 n'ont pas été développés dans le cadre de cet exemple. Pour cela, il eut été nécessaire de mettre en place des techniques de validation sortant largement du cadre de cette thèse. Elles devront cependant être étudiées à moyen terme.

Du point de vue de l'algorithme de synthèse, l'exemple nous a conduit pas à pas à suivre la démarche de synthèse, jusqu'à l'obtention finale des lois de commande correspondantes aux deux situations étudiées. Que ce soit pour le scénario de configuration ou celui de reconfiguration, l'exemple a montré les capacités de l'algorithme proposé à gérer l'incertitude d'entrée des produits dans le poste d'approvisionnement, sa capacité également à gérer la simultanéité d'évolution de produits différents, et enfin son efficacité pour garantir les performances recherchées (ici, le temps de cycle). Enfin, nous avons montré son aptitude à faire face à des problèmes liés à la perte de services de la partie opérative. L'algorithme peut dans ce contexte apporter de véritables solutions, notamment sur le plan décisionnel. En effet, il permet d'évaluer, avec confiance, si une continuité de service peut être encore envisagée.

Enfin, du point de vue de la mise en œuvre, l'exemple nous a permis, dans un premier temps, de valider en partie l'atelier logiciel que nous avons développé. Les lois de commande actuellement générées sous forme réseau de Petri ne tiennent en effet pas encore compte du mécanisme d'insertion et de la problématique multi-produit. Cette incomplétude sera à très court terme comblée. En second lieu, disposer d'un tel outil de synthèse nous a permis d'étudier, par expérimentation, la capacité de l'algorithme de synthèse à faire face à la complexité inhérente aux procédés manufacturiers. Les résultats de cette étude semblent prometteurs. Ils nous encouragent donc à les compléter par une étude théorique poussée. Enfin, l'outil développé en Java, nous permet à ce jour d'envisager, sereinement et à moyen terme, son implantation sur différentes cibles (PC, PLC, autre...). L'intégration, depuis les années 2000, de technologies de type serveurs web embarqués dans les automates programmables ouvre des perspectives très intéressantes pour abriter des fonctions évoluées, comme par exemple la synthèse automatique de lois de commande, et pourquoi pas même, des interpréteurs réseaux de Petri ; la partie automate (cycle automate) pouvant alors être limitée aux commandes internes aux chaînes fonctionnelles.

Conclusion générale

Les travaux que nous avons présentés dans ce document traitent de la supervision et de la commande des procédés industriels complexes. Ils font suite à ceux déjà réalisés Mendez (2002) dans ce domaine au sein de l'axe Conduite et Organisation des Systèmes de Production du Laboratoire d'Automatique de Grenoble.

La contribution de nos travaux est double. Elle a non seulement porté sur la proposition d'un formalisme de modélisation adapté à la modélisation des systèmes contrôlés par le niveau 1 du CIM et également à l'élaboration d'une technique de synthèse de lois de commande optimales respectant à la fois les contraintes normatives et législatives liées à l'exploitation de parties opératives ainsi que les spécifications produits.

Le formalisme de modélisation que nous avons proposé est une extension de celui couramment utilisé dans le domaine de la Planification Automatique. Pour chaque type d'opération caractéristique d'un système de production vue au niveau 1 du CIM, à savoir les opérations de transformation des produits, de transitique et de préparation des machines, une structure générique de capitalisation de la connaissance a été proposée. Six avantages au moins découlent de ce formalisme :

1. Fondé sur un vocabulaire à la fois précis et proche de celui utilisé par l'ingénieur méthode, il favorise son appropriation,
2. Clairement localisé au niveau de la coordination des chaînes fonctionnelles, le contexte de son utilisation est bien délimité, permettant ainsi à l'expert de se focaliser sur les champs des opérations à modéliser,
3. Fondé sur une modélisation des opérations, ce formalisme est bien adapté à la maîtrise de la complexité inhérente aux systèmes contrôlés auxquels nous nous intéressons,
4. L'ensemble des propriétés accompagnant ce formalisme, permet d'envisager sereinement la validation du modèle identifié,
5. Les prémisses d'une méthodologie de modélisation sont proposés, permettant d'assister l'expert durant sa phase d'acquisition de la connaissance,
6. La structuration des modèles en "fiches d'opérations", facilite la réactualisation du modèle global, notamment durant les phases "lourdes" de maintenance pour lesquelles des éléments de partie opérative deviennent inutilisables, ou encore pour les situations d'extension d'ateliers.

Fort d'un tel formalisme de modélisation, nous avons ensuite proposé une démarche de génération automatique de lois de commande optimales pour le niveau 1 du CIM, contribuant ainsi à apporter une brique supplémentaire à la Supervision de chaînes fonctionnelles. Cette démarche s'appuie sur une méthode proche de celles utilisées dans le cadre de l'ordonnancement d'atelier et de la recherche opérationnelle. Basée sur une technique de recherche de chemin dans un graphe, et appliquée successivement à des graphes d'états de complexité maîtrisée, cette démarche permet de garantir les performances de la solution générée. De ce point de vue, l'approche proposée, véritablement novatrice dans le domaine, a donné lieu à des résultats significatifs. Ils se caractérisent comme suit :

1. Le mécanisme de décomposition du problème selon un découpage en trois niveaux d'opérations permet de réduire considérablement la complexité du modèle à manipuler sans pour autant faire abstraction de la recherche de performances de la solution finale,
2. La prise en compte native du problème lié à la recherche de performance, permet d'envisager sereinement l'utilisation de la démarche de synthèse de lois de commande pour différents contextes de fonctionnement (configuration, exploitation en mode de production normale, dégradé voire même critique) ne serait-ce qu'en jouant sur le niveau de profondeur algorithmique à atteindre,
3. Basé sur une structuration forte des informations contenues dans le modèle du système contrôlé, il s'en suit une simplicité apparente de l'algorithme proposé, basé notamment sur une utilisation conjointe de l'algorithme de Dijkstra, et de mécanismes de parallélisation et d'insertion.
4. La problématique traitée par l'algorithme de synthèse est très proche de la réalité, couvrant ainsi des problèmes allant de la commande cyclique mono-produit à la commande multi-flux de produits en contexte dégradé.
5. Comme nous avons pu le voir dans ce manuscrit, nous avons eu le souci permanent de valider notre démarche sur un procédé réel. Aussi, nous avons développé, en collaboration avec l'équipe informatique du LAG, un atelier logiciel de synthèse de lois de commande. Les résultats à ce jour obtenus sont significatifs, au point d'envisager à court terme non seulement de le terminer mais également de l'intégrer au sein de l'outil logiciel CERBERE développé au LAG et implanté dans l'architecture de pilotage du procédé SAPHIR.
6. Les premières études expérimentales portant sur la complexité de l'algorithme nous encouragent à croire en sa capacité à traiter des problèmes de taille industrielle,
7. Enfin, il est important de remarquer que la solution générée par l'algorithme de synthèse s'appuie sur un formalisme de représentation propice à son exploitation par des techniques de compilation ou d'interprétation, mais également à des traductions vers des langages de la norme IEC 61131-3 Lee et al. (2004) David (1992), ouvrant ainsi des perspectives d'intégration intéressantes au sein d'architectures d'automates programmables.

Au terme de ces travaux, plusieurs axes de recherche se dégagent pour envisager, du point de vue des perspectives, de prolonger l'étude menée pendant ces trois ans.

A court terme, six axes d'investigations peuvent être envisagés :

- A très court terme, la finalisation de l'atelier logiciel de synthèse de lois de commande devra être menée afin de valider l'approche sur de nombreux autres cas d'études. Au delà de ce développement informatique du mécanisme de synthèse, la phase amont de capitalisation des connaissances devra également être informatisée au travers du développement d'une interface graphique de saisie des modèles sous forme de fiches. L'environnement de saisie devra non seulement intégrer des fonctions d'aide à la modélisation mais également des fonctions de validation des propriétés proposées dans la partie II de ce mémoire.
- Dans une démarche d'intégration future en contexte industriel, des études de traduction des lois de commande générées sur la base de l'outil réseau de Petri dans les langages de la norme IEC61131-3 devront être menées.
- Sur un plan plus technique, les travaux de recherche devront évaluer exactement l'impact des hypothèses faites tout au long de ce mémoire (gel des parallélismes, décomposition, ...) sur les performances de la solution générée et en fonction par exemple du niveau de flexibilité offerte par le système contrôlé Berruet (1998).
- Cette dernière étude pourra alors être avantageusement complétée par une étude théorique stricte de la complexité de l'algorithme proposé. Cela permettra par exemple de caractériser précisément ses aptitudes à calculer des solutions en fonction par exemple du nombre de chaînes fonctionnelles pilotées. De tels résultats permettraient de fournir des données essentielles aux fonctions de supervision, ne serait-ce que pour Décider d'une éventuelle remise en cause de la décision imposée par le niveau supérieur.
- Nous avons considéré dans ce document une traduction ultime des lois de commande générées en réseau de Petri ordinaire interprété. Force est de constater que dans le cadre de la synthèse de lois de commande multi-flux de produits, le formalisme retenu n'est plus adapté. Le recours aux réseaux de Petri Prédicats/Transitions serait certainement davantage approprié et devrait donc être étudié.
- Enfin, l'outil de synthèse développé en Java, nous permet à ce jour d'envisager sereinement son implantation sur différentes cibles (PC, PLC, autre...). L'intégration, depuis les années 2000, de technologies de type serveurs web embarqués dans les automates programmables ouvre des perspectives très intéressantes pour abriter des fonctions évoluées, comme par exemple la synthèse automatique de lois de commande, et pourquoi pas même, des interpréteurs réseaux de Petri ; la partie automate (cycle automate) pouvant alors être limitée aux commandes internes aux chaînes fonctionnelles. Dans cette perspective, des relations devront être établies avec des partenaires industriels fortement sensibilisés à cette problématique (DELMIA Automation, Schneider Electric, Siemens,...).

A moyen terme, nous pouvons mettre en évidence quatre orientations de recherche possibles.

- Premièrement, l'approche proposée devra être étendue aux aspects multi-critères de manière à garantir davantage de performances et affiner ainsi les prises de décisions.
- Deuxièmement, et afin de conforter encore les prises de décision, les fonctions de suivi et de diagnostic devront être étudiées afin d'envisager les mises à jour du modèle, notamment en présence de défaillances de la partie opérative. Pour cela, une première étude devra être lancée afin d'évaluer l'adéquation de la structure des informations soumise aux besoins de telles fonctions, et dans un deuxième temps envisager un enrichissement du modèle par des

informations de type statistiques, historiques, etc...

- Troisièmement, nous avons montré l'intérêt que pouvait avoir un algorithme de génération automatique de lois de commande ne serait-ce que pour envisager un pronostic de situations. Cette voie de recherche devra être poursuivie en tenant compte du contexte de production (modes de marches), de la stratégie de l'entreprise et des normes législatives imposées à l'entreprise (environnement, ...), ou encore à des analyses de risques. Une telle démarche devra bien entendu s'appuyer sur les travaux déjà réalisés dans ce domaine Mendez (2002) et conduire ainsi à une intégration progressive dans l'approche CERBERE.
- Quatrièmement, et en particulier au niveau de la fonction décision, une paramétrisation de l'algorithme de synthèse de lois de commande devra être envisagée afin par exemple d'orienter la recherche de chemins en privilégiant, en fonction de la criticité de la situation, des trajectoires évitant le passage par des états non observables, l'utilisation de parties opératives goulot, etc...
- Cinquièmement, suite à une défaillance, la reconfiguration ne devra plus se limiter à la génération d'une nouvelle loi de commande. En effet, il sera par exemple envisageable de terminer les opérations en cours d'exécution au moment de la défaillance. Une telle perspective devra passer par une analyse fine de modèle permettant de mettre en exergue le caractère préemptif de certaines opérations.

Par ailleurs, et à plus long terme, il serait particulièrement intéressant d'ouvrir cette activité de recherche à d'autres domaines comme par exemple les SHS pour développer davantage les aspects capitalisation de la connaissance, ou encore la mécanique et plus particulièrement avec le secteur de la conception de process et de produits pour envisager ainsi d'extraire automatiquement les contraintes à modéliser des modèles 3D développés.

Bibliographie

Achour, Z. et Rezg, N. (2004). Commande par supervision des systèmes à événements discrets basée sur l'utilisation du grafcet et la théorie des régions. *APII-JESA Journal Européen des Systèmes Automatisés*, A paraître. 46, 96

ADEPA (1981). *Guide de l'Etude des Modes de Marches et d'Arrêts.* Adepa, Agence de la Productique, 17 rue Perier, 92120 Montrouge. 33, 47

AFNOR (1991). *NF X50-151 Analyse de la valeur - Analyse fonctionnelle - Expression fonctionnelle du besoin et cahier des charges fonctionnel.* 20

Amar, S. (1994). *Systèmes Automatisées de Production Manufacturirère : Méthodologie de Conception du Système de Coordination par prototypage Orienté Objet de la Partie procédé.* Ph.d. thesis, Université des Sciences et Technologies de Lille, France. 42, 43

Artigues, C. (1997). *Ordonnancement en Temps Reel d'Ateliers avec Temps de préparation des ressources.* Ph.d. thesis, Université Paul Sabatier, Toulouse, France. 41, 112

Aylett, R. S. (2001). Planning plant operating procedures for chemical plant. *Engineering Applications of Artificial Intelligence*, 14(3) :341–356. 39, 73, 74

Baker, K. (1974). *Introduction to sequencing and scheduling.* Wiley, New York. 40

Belabbas, A. et Berruet, P. (2004). Fms reconfiguration based on petri nets models. In *IEEE International Conference on System Man and Cybernetic (SMC'04)*, The Hague, Netherlands. 43

Ben-Arieh, D., Kumar, R. R., et k. Tiwari, M. (2004). Analysis of assembly operations' difficulty using enhanced expert high-level colored fuzzy petri net model. *Robotics and Computer Integrated Manufacturing*, 20 :385–403. 39

Berruet, P. (1998). *Contribution au recouvrement des systèmes flexibles de production manufacturière : analyse de la tolérance et reconfiguration.* Ph.d. thesis, Université des Sciences et Technologies de Lille, France. 21, 42, 43, 199

Berruet, P., Toguyeni, A. K. A., et Craye, E. (1999). Considering parts in progress in a reconfiguration procedure for fms. In *IMACS CSCC'99, Modern Applied Mathematics Techniques in Circuits, Systems and Control.* 164

Berruet, P., Toguyeni, A. K. A., Elkhattabi, S., et Craye, E. (2000). Toward an implementation of recovery procedures for fms supervision. *Computers in Industry*, 43 :227–236. 30

Boufaied, A. (2000). Etude de la fonction pronostic : cas d'ateliers flexibles de production. Masters dissertation thesis, Université Paul Sabatier, Toulouse, France. 54, 192

Boufaied, A. (2003). *Contribution à la Surveillance Distribuée des Systèmes à Evénements Discrets Complexes*. Ph.d. thesis, Université Paul Sabatier, Toulouse, France. 31

Brandim, B. (1996). The real-time supervisory control of an experimental manufacturing cell. *IEEE Transaction on Robotics and Automation*, 12(1) :1–14. 45

Brauner, N., Castagna, P., et et al., M. L. E. (2004). Les systèmes flexibles de production. *APII-JESA Journal Européen des Systèmes Automatisés*, A paraître. 40

Castillo, L. (2000). A three-level knowledge-based system for the generation of live and safe petri nets for manufacturing systems. *Journal of Intelligent Manufacturing*, 11 :559–572. 39, 73, 74

Castillo, L., Fdez-Oliveras, J., et Gonzales, A. (2000). Automatic generation of control sequences for manufacturing systems based on partial order planning techniques. *Artificial Intelligence in Engineering*, 14 :15–30. 39

Castillo, L., Fdez-Oliveras, J., et Gonzales, A. (2001). Mixing expressiveness and efficiency in a manufacturing planner. *Journal of Experimental and Theoretical Artificial Intelligence*, 13 :141–162. 39

Chafik, S. (2000). *Proposition d'une structure de Contrôle par Supervision Hiérarchique et Distribuée : Application à la Coordination*. Ph.d. thesis, Instut National des Sciences Appliquées de Lyon, France. 45

Chaillet, A. (1995). *Approche Multi Modèles pour la Commande et la Surveillance en Temps Réel des Systèmes à Evénements Discrets*. Ph.d. thesis, Université Paul Sabatier, Toulouse, France. 30, 34, 56

Charbonnier, F., Alla, H., et david, R. (1999). The supervised control of discret-event dynamic systems. *IEEE Transactions on Control Systems Technology*, 7(2) :175–187. 46

Chretienne, P., Coffman, E. G., Lensta, J. K., et Liu, Z. (1997). *Scheduling theory and its applications*. Chichester : John Wiley and Soons. 102

CIM (1989). A reference model for computer integrated manufacturing from the viewpoint of industrial automation. *International Journal of Computer Integrated Manufacturing*, 2(2) :114–127. 22

Clavier, J. (1997). Qualité et qualitique. *Techniques de l'ingénieur, traité Génie Industriel*, (A8750). 20

Combacau, M. (1991). *Commande et surveillance des systèmes à événements discrets complexes : application aux ateliers flexibles*. Ph.d. thesis, Université Paul Sabatier, Toulouse, France. 22, 23, 29, 30, 54, 56

Combacau, M., Berruet, P., Zamaï, E., Charbonnaud, P., et Khatab, A. (2000). Supervision and monitoring of production systems. In *IFAC 2nd Conference on Management and Control of Production and Logistics (MCPL'00)*, Grenoble, France. 26, 28, 33

Cruette, D. (1991). *Méthodologie de Conception des Systèmes Complexes à Evénements Discrets : Application à la Concpetion et à la Validation Hiérrachisée de la Commande de Cellules Flexibles de Production dans l'Industrie Manufacturière*. Ph.d. thesis, Université des Sciences et Technologies de Lille, France. 43

Culita, J. (2001). Autonomous agent based control/supervision architecture for flexible manufacturing systems. *Proceedings of the 9th IFAC/IFORS/IMACS/IFIP Symposium (LSS'01)*, 28(12). 30

da Silveira, M. R. (2003). *Sur la Distribution avec redondance Partielle de Modèles à Evénements Discrets pour la Supervision de Procédés industriels*. Ph.d. thesis, Université Paul Sabatier, Toulouse, France. 55, 67, 88

Dangoumau, N. (2000). *Contribution à la gestion des modes des systèmes automatisés de production*. Ph.d. thesis, Université des Sciences et Technologies de Lille, France. 30, 43

David, R. (1992). *Du Grafcet aux réseaux de Petri*. Hermes, Paris, France. 198

de Boulay, B. (2001). Book review. *Artificial Intelligence*, 125 :227–232. 37

Deschamps, E. (2004). Architecture des modèles du procédé. Masters dissertation thesis, Institut National Polythechnique de Grenoble, France. 69, 183

Deschamps, E., Henry, S., Zamaï, E., et Jacomino, M. (2004). Controlled system model with petri net formalism for reconfiguration. In *IFAC Conference on Manufacturing, Modelling, Management and Control (MIM'04)*, Athens, Greece. 69, 183

Dijkstra, E. W. (1959). A note in two problems in connexion with graphs. *Numerisches Mathematik*, (1) :269–271. 111, 126

Doumeingts, G., Vallespir, B., et Chen, D. (1995). Methodologies for designing cim systems : a survey. *Computers in Industry*, 25(3) :263–280. 22

Emerson, E. A. (1990). *Temporal and Modal Logic*, volume B of *Handbook of Theorical Computer Science*. Elsevier Science. 48, 101

Esquirol, P. et Lopez, P. (1999). *L'ordonnancement*. Collection Gestion, serie : Production et techniques quantitatives appliquées à la gestion. Economica, Paris, France. 41, 42

Fletcher, M., Brennan, R. W., et Norrie, D. H. (2003). Modeling and reconfiguring intelligent holonic manufacturing systems with iternet-based mbile agents. *Journal of Intelligent Manufacturing*, 14 :7–23. 30

Frey, G. et Litz, L. (2000). Formal methods in plc programming. In *IEEE Conference on Systems Man and Cybernetics (SMC'00)*, Nashville, USA. 27, 48, 101

Gansner, E. R., Koutsofios, E., et North, S. C. (1993). A technique for drawing directed graph. *IEEE Transaction on Software Engineering*, 19(3) :214–230. 194

Gendreau, D., Bodart, A., Carre-Menetrier, V., de Loor, P., Deluche, J. B., Dupont, J., Hancq, J., Kril, A., et Nido, J. (2000). *7 Facettes du GRAFCET : Approches Pratiques de la Conception à l'Exploitation*. Cépadus. 23, 48

Gentil, S. et Zamaï, E. (2003). Principes de chaînes de régultaion. *Techniques de l'ingénieur, traité Informatique Industrielle*, (S7090). 51

Gentina, J. C., Korbaa, O., et Camus, H. (2002). *Problèmes d'Ordonnancement Cyclique*. Traité IC2, Section Productique, Ordonnancement de la Production. 146

Ghaffari, A., Rezg, N., et Xie, X. (2003). Feedback control logic for forbidden state problems of marked graphs : Application to a real manufacturing system. *IEEE Transactions on Automatic Control*, 48(1). 45

Ghallab, M., Nau, D., et Traverso, P. (2004). *Automated Planning, Theory and Practice*. Elsevier. 37, 38, 102

Gouyon, D. (2004). *Contrôle par le Produit des Systèmes d'Exécution de la Production : Apport des Techniques de Synthèse*. Ph.d. thesis, Université Henri Poincaré, Nancy, France. 30, 45

Henry, S., Deschamps, E., Zamaï, E., et Jacomino, M. (2004a). Control law synthesis algorithm for discrete-event systems. In *IFAC Conference on Management and Control of Production and Logistics (MCPL'04)*, Santiago, Chili. 119

Henry, S. et Faure, J. M. (2003). Elaboration of invariant safety properties from fault-tree analysis. In *IEEE International Conference on Computational Engineering in Systems Applications (CESA'03)*, Lille, France. 48

Henry, S., Zamaï, E., et Jacomino, M. (2003). Decisional requirements for supervision, monitoring and control structures. In *IEEE International Conference on Computational Engineering in Systems Applications (CESA'03)*, Lille, France. 31

Henry, S., Zamaï, E., et Jacomino, M. (2004b). Real time reconfiguration of manufacturing system. In *IEEE International Conference on System Man and Cybernetic (SMC'04)*, The Hague, Netherlands. 71

Henry, S., Zamaï, E., et Jacomino, M. (2005). Controlled system model adapted for control law synthesis. In *16th IFAC World Congress (IFAC'05)*, Prague, Czech Republic. 99

Heragu, S. S., Graves, R. J., Kim, B., et Onge, A. S. (2002). Intelligent agent based framework for manufacturing systems control. *IEEE Transactions on Systems, man and Cybernetics, Part. A*, 32(5). 30

Holloway, L. E., Guan, X., Sundaravadivelu, R., et Jr., J. A. (2000). Automated synthesis and composition of taskblocks for control of manufacturing systems. *IEEE Trans. On Systems, Man and Cybernetics, Part. B*, 30(5). 50

Holloway, L. E. et Krogh, B. (1990). Synthesis of feedback control logic for a class of controlled petri nets. *IEEE Transaction on Automatic Control*, 35(5) :514–523. 45

IEC-1025 (1990). *Fault-tree Analysis*. Swiss. 105

ISA-S95 (2000). *Entreprise Control System Integration*. 22

ISO-1101 (2004). *Geometrical Product Specifications (GPS) – Geometrical tolerancing – Tolerances of form, orientation, location and run-out*. 20

ISO-9000 (2000). *Systèmes de management de la qualité – Principes essentiels et vocabulaire.* 20

Julia, S., Valette, R., et Fernandes, J. M. (1998). Scheduling batch systems using a token player algorithm. In *IEEE International Conference on System Man and Cybernetic (SMC'1998)*, pages 487–492, San Diego, USA. 32

Kempowsky, T., Subias, A., et Aguillar-Martin, J. (2004). Supervision of complex processes : Strategy for fault detection and diagnosis. In *IFAC Conference on Management and Control of Production and Logistics (MCPL'04)*, Santiago, Chili. 29

Khatab, A. (2000). *Contrôle et contrôle Stabilisant des Systèmes à Evénements Discrets Temporels : Application au Recouvrement des Défaillances des Systèmes de Production.* Ph.d. thesis, Institut National des Sciences Appliquées de Lyon, France. 30

Kjaer, A. P. (2003). The integration of business and production processes. *IEEE Control Systems Magazine*, 23(6) :50–58. 22

Klein, I. (1999). Efficient planning for a miniature assembly line. *Artificial Intelligence in Engineering*, 13(1) :69–81. 39, 73, 74

Kramer, T. R. et Senehi, M. K. (1993). Feasibility study : Reference architecture for machine control systems integration. Nist ir 5297, national institute of standards and technology, gaithersburg, md. 22

Lacomme, P., Prins, C., et Sevaux, M. (2003). *Algorithmes de Graphes.* Eyrolles. 122, 125

Laprie, J. C. (1995). *Guide de la sûreté de fonctionnement.* Cépadus, Toulouse, France. 27

Lee, G. B., Zandong, H., et Lee, J. S. (2004). Automatic generation of ladder diagram with control petri net. *Journal of Intelligent Manufacturing*, 15 :245–252. 198

Lee, J. K. et Lee, T. E. (2002). Automata-based supervisory control logic design for a multi-robot assembly cell. *International Journal of Computer Integrated Manufacturing*, 15(4) :319–334. 46

Luger, G. F. (2002). *Artificial Intelligence : Structures and Strategies for Complex Problem Soving.* Pearson Education. 37

mac Millan, K. L. (1993). *Symbolic Model Checking.* Kluwer Academic. 48, 101

Macedo, P., Sinogas, P., et Tribolet, J. (2004). Information systems support for manufacturing processes : The standard s95 perspective. In *6th International Conference on Entreprise Information Systems (ICEIS'04).* 22

Mader, A., Brinksma, E., Wupper, H., et Bauer, N. (2001). Design of a plc control program for a batch plant vhs case study 1. *European Journal of Control*, 7 :416–439. 50

Mendez, H. (2002). *Synthèse de lois de surveillance pour les procédés industriels complexes.* Ph.d. thesis, Institut National Polythechnique de Grenoble, France. 30, 31, 32, 33, 164, 177, 197, 200

Mendez, H., Zamaï, E., et Descotes-Genon, B. (2003). Quality, productivity, security and ecological constraints for synthesis of monitoring laws. In *5eme Congrès International Pluridisciplinaire Qualité et Sûreté de Fonctionnement (QUALITA 2003)*, Nancy, France. 20, 33

Mendez, H., Zamaï, E., et Descotes-Genon, B. (2004). Quality, productivity, security and ecological criteria for synthesis of monitoring laws. *Quality Engineering*, to be published. 55

Merlaud, C., Perrin, J., et Trichard, J. P. (1995). *Automatique, Informatique Industrielle.* Sciences et Techniques Industrielles. Dunod. 22

Moreno, S. (1997). *le GEMMA - Guide de l'Etude des Modes de Marches et d'Arrêts.* Educalivre, Paris, France. 47

Mouchard, J. S. (2002). *Proposition d'une Approche Méthodique pour la Conception des Systèmes Automatisés de Production, application aux Systèmes Transitiques.* Ph.d. thesis, Université de Bretragne Sud, France. 30

Nagel, M. H. et Tomiyama, T. (2004). Intelligent sustainable manufacturing systems, management of the linkage between sustainability and intelligence, an overview. In *IEEE International Conference on System Man and Cybernetic (SMC'04)*, The Hague, Netherlands. 25

Newborn, M. et Newborn, M. (2001). *Automated Theorem Proving : Theory and practice.* Springer-Verlag. 101

Niel, E. et Craye, E., editors (2002). *Maîtrise des risques et sûreté de fonctionnement des systèmes de production.* Hermès-Lavoisier. 28

Nilson, N. J. (1998). *Artificial Intelligence : A new Synthesis.* Morgan Kaufman. 37

Nourelfath, M. (1997). *Extension de la Théorie de la Supervision à la Surveillance et à la Commande des Systèmes à Evénements Discrets : Application à la Sécurité Opérationnelle des Systèmes de production.* Ph.d. thesis, Institut National des Sciences Appliquées de Lyon, France. 45

Odrey, N. G. et Mejia, G. (2003). A re-configurable multi-agent system architecture for error recovery in production systems. *Robotics and Computer Integrated Manufacturing*, 19 :35–43. 30

Qiu, R., Wysk, R., et Xu, Q. (2003). Extend structured adaptative supervisory control model of shop floor controls for an e-manufacturing system. *International Journal of Production Research*, 41(8) :1605–1620. 46

Ramadge, P. J. et Wonham, W. M. (1987). Supervisory control of a class of discret event processes. *Journal of Control and Optimization*, 25(1). 45

Rezg, N. (1996). *Contribution à la sécurité opérationnelle des systèmes : Mises en oeuvre d'une structure de surveillance basées sur les RdP Controlés.* Ph.d. thesis, Instut National des Sciences Appliquées de Lyon, France. 30

Rossi, A. (2003). *Ordonnancement en Milieu Incertain, Mise en Oeuvre d'une Démarche Robuste.* Ph.d. thesis, Institut National Polytechnique de Grenoble, France. 41, 112

Roussel, J. M., Faure, J. M., Lesage, J. J., et Medina, A. (2004). Algebraic approach for dependable logic control systems designe. *International Journal of Production Research*, 42(14) :2859–2876. 50

Sandewall, E. et Rönnquist, R. (1986). A representation of action structures. In *National Conference on Artificial Intelligence (AAAI'86)*, pages 89–97. 39, 73

Sargent, R. W. H. (1998). A functional approach to process synthesis and its application to distillation systems. *Computers Chemical Engineering*, 22(1-2) :31–45. 47

Schnoebelen, P. (1999). *Vérification de Logiciels : Techniques et Outils de Model-Checking*. Springer. 48

Sornaz, D. N. et Khoshnevis, B. (2003). Generation of alternative process plans in integrated manufacturing systems. *Journal of Intelligent Manufacturing*, 14 :509–526. 39, 42

Sourise, C. et Boudillon, L. (1997). *La sécurité des machines automatisées. Tome 2 : Techniques et moyens de prévention opératifs -Systèmes de commande - Utilisation des machines*. Collection Technique. Groupe Schneider. 27, 177

Tawegoum, R. (1995). *Contrôle Temps Réel du Déroulement des Opérations dans les Systèmes de Production Flexibles*. Ph.d. thesis, Université des Sciences et Technologies de Lille, France. 20, 43

Trentesaux, D. et Sénéchal, O. (2002). Conduite des systèmes de production manucfacturière. In *Techniques de l'ingénieur, traité Informatique Industrielle*, number S7598. 21

Trinquart, R. (2004). Analyse d'accessibilité dans l'espace des plans partiels. *Journal Electronique d'Intelligence Artificielle*, 5. 40

Unger, K. (2001). Manufacturers'needs not changing-but acronyms are. *Industrial Computing*, pages 46–48. 22

Valette, R. (2002). Les réseaux de petri. 156, 157

Veille, J. (2000). Intégration production - entreprise : La norme ansi/isa s95. In *Dixième Journée des CPIM de France*. 22

Villemeur, A. (1988). *Sûreté de fonctionnement des systèmes industriels*. Eyrolles, Paris, France. 27, 105

Weld, D. S. (1994). An introduction to least commitment planning. *Artificial Intelligence Magazine*, 15(4) :27–61. 39, 40

Zamaï, E. (1997). *Architecture de surveillance-commande pour les systèmes à événements discrets complexes*. Ph.d. thesis, Université Paul Sabatier, Toulouse, France. 30, 33

Zamaï, E., Chaillet-Subias, A., Combacau, M., et Bonneval, A. D. (1997). A hierarchical struture for control of discrete events systems and monitoring of process failures. *Studies in Informatics and Control*, 6(1). 30

Zamai, E. (2001). Intégration et développement de nouvelles technologies pour le pilotage et la supervision temps réel d'ateliers manufacturiers. Laboratoire d'automatique de grenoble (lag). 172

Zaytoon, J. (2002). On the recent advances in grafcet. *Production Planning and Control*, 13(1) :86–100. 46, 47

Zaytoon, J. et Carré-Ménétrier, V. (1999). Grafcet et graphe d'états : comportement, raffinement, vérification et validation. *APII-JESA Journal Européen des Systèmes Automatisés*, 33(7) :751–782. 46

Zwingelstein, G. (1995). *Diagnostic des défaillances : Théorie et pratique pour les systèmes industriels*. Traité des Nouvelles technologies. Hermes. 31

Zwingelstein, G. (1999). Sûreté de fonctionnement des systèmes industriels complexes. *Techniques de l'ingénieur, traité Informatique Industrielle*, (S8250). 27

Annexes

Annexe A

Modélisation du système contrôlé

Le lecteur trouvera dans cette annexe, les modèles et les définitions complètes du comportement d'une opération d'information, d'une évolution induite, et d'une évolution requise.

1 Les Opérations d'Information

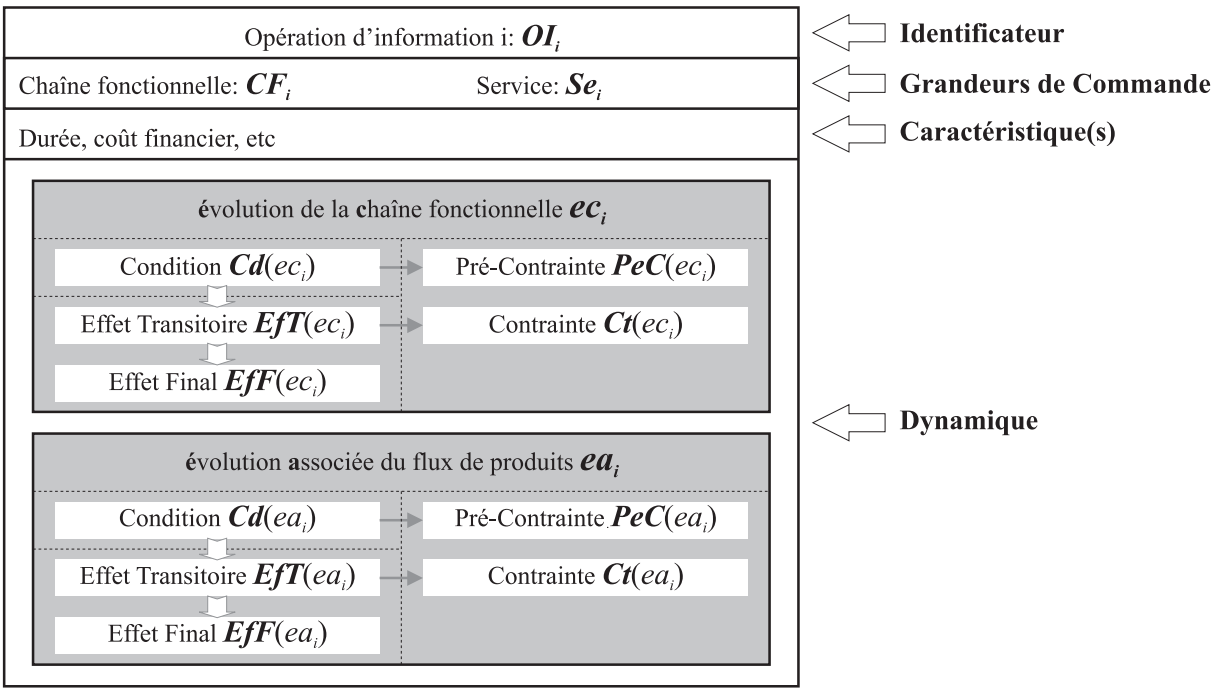


FIG. A.1 – Champs de données et notations du modèle générique d'une opération d'information.

2 Les Évolutions Induites

Le système de notation adopté pour les comportements d'une opération d'action, présenté § 5.1 page 78 est conservé pour les comportements d'une évolution induite avec pour seule différence la notation d'un comportement, $CEI_{i,k}$ à la place de $COA_{i,k}$.

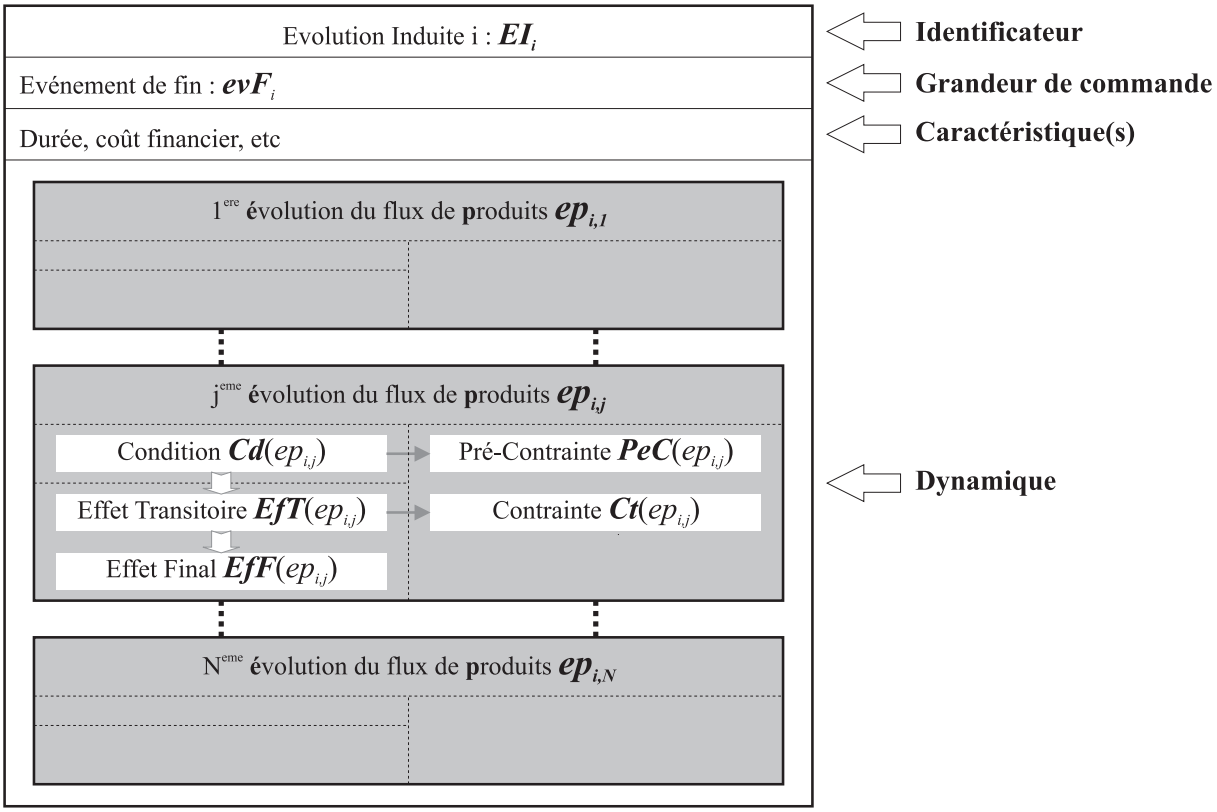


FIG. A.2 – Champs de données et notations du modèle générique d'une évolution induite.

Les définitions d'un comportement d'une évolution induite sont basées sur les mêmes concepts et principes que celles d'un comportement d'une opération d'action. Par conséquent, nous donnerons uniquement ici les quatre définitions sans commentaires spécifiques pour aboutir à la Figure 5.10 qui représente le comportement d'une évolution induite avec ses événements de début et de fin d'un point de vue grandeurs de commande.

Ainsi, nous commençons par définir l'ensemble des états initiaux depuis lesquels un comportement d'une évolution induite est provoqué par l'instabilité de ces états initiaux. Et ensuite, nous définissons l'état atteint depuis un état instable appartenant à cet ensemble d'états initiaux.

Définition 10 *L'ensemble des états initiaux, $Q_{It(CEI_{i,k})}$, est l'ensemble des états depuis lesquels il y a occurrence de l'événement de début de l'évolution induite $EI_{i,k}$ avec le comportement $CEI_{i,k}$. Formellement, cet ensemble est représenté par :*

$$Q_{It(CEI_{i,k})} = \{q \in Q_S / \bigwedge_{j \in J_k} [Cd(ep_{i,j}) \wedge PeC(ep_{i,j})] \bigwedge_{j \in \overline{J_k}} \neg Cd(ep_{i,j}) = VRAIE\}$$

Définition 11 *Depuis un état q appartenant à l'ensemble des états initiaux du comportement $CEI_{i,k}$, $q \in Q_{It(CEI_{i,k})}$, et après l'occurrence de l'événement de début de l'évolution induite $EI_{i,k}$, noté $d(EI_{i,k})$, le système contrôlé atteint un état q'_1.*

Les variables d'états de l'état q'_1 ont pour valeurs celles de l'état q modifiées par les effets transitoires $EfT(ep_{i,j})$ pour $j \in J_k$.

La fonction de transition partielle est définie par $q'_1 = \delta(d(EI_{i,k}), q)$.

$$q \xrightarrow{d(EI_{i,k})} q'_1$$

Comme cela a été précisé au § 7, l'état initial d'une évolution induite est un état dit instable puisque le flux de produits continue d'évoluer depuis un tel état. Ainsi, il n'existe pas de grandeur de commande associée au début d'une évolution induite. La fonction de transition s'écrit alors $q'_1 = \delta(q)$.

$$q \rightarrow q'_1$$

Quand une évolution induite est en cours, d'autres actions peuvent être exécutées à condition que l'état du système contrôlé et de son environnement reste dans un ensemble d'états intermédiaires de manière à ne pas violer de contraintes de sécurité et d'écologie. Cet ensemble d'états intermédiaires est maintenant défini.

Définition 12 *L'ensemble des états intermédiaires,* $Q_{Id(CEI_{i,k})}$*, est l'ensemble des états dans lesquels le système contrôlé et son environnement doivent rester suite à l'occurrence de l'événement de début de l'évolution induite* $EI_{i,k}$ *avec le comportement* $CEI_{i,k}$ *et avant l'occurrence de l'événement de fin de l'évolution induite* $EI_{i,k}$*. Formellement, cet ensemble est représenté par :*

$$Q_{Id(CEI_{i,k})} = \{q \in Q_S / \bigwedge_{j \in J_k} [EfT(ep_{i,j}) \wedge Ct(ep_{i,j})] = VRAIE\}$$

Depuis un des états de cet ensemble d'états intermédiaires, la fin de l'évolution induite va faire évoluer le flux de produits comme défini ci-dessous.

Définition 13 *Depuis un état* q'_2 *appartenant à l'ensemble des états intermédiaires du comportement* $CEI_{i,k}$*,* $q'_2 \in Q_{Id(CEI_{i,k})}$*, et après l'occurrence de l'événement de fin de l'évolution induite* $EI_{i,k}$*, noté* $f(EI_{i,k})$*, le système contrôlé atteint un état* q''*.*

Les variables d'états dans l'état q'' *ont pour valeurs celles dans l'état* q'_2 *modifiées par les effets finaux* $EfF(ep_{i,j})$ *pour* $j \in J_k$*.*

La fonction de transition partielle est définie par $q'' = \delta(f(EI_{i,k}), q'_2)$.

$$q'_2 \xrightarrow{f(EI_{i,k})} q''$$

3 Les Évolutions Requises

Le système de notation adopté pour les comportements d'une opération d'action, présenté § 5.1 page 78, est conservé pour les comportements d'une évolution requise avec pour seule différence la notation d'un comportement, $CER_{i,k}$ à la place de $COA_{i,k}$.

L'unique différence entre le comportement d'une opération d'action et d'une évolution requise est le module de coordination responsable du déclenchement. Comme nous l'avons indiqué pour les évolutions induites, nous donnerons uniquement ici les quatre définitions sans commentaires spécifiques pour aboutir à la Figure 5.12 qui représente le comportement d'une évolution requise avec ses événements de début et de fin d'un point de vue grandeurs de commande.

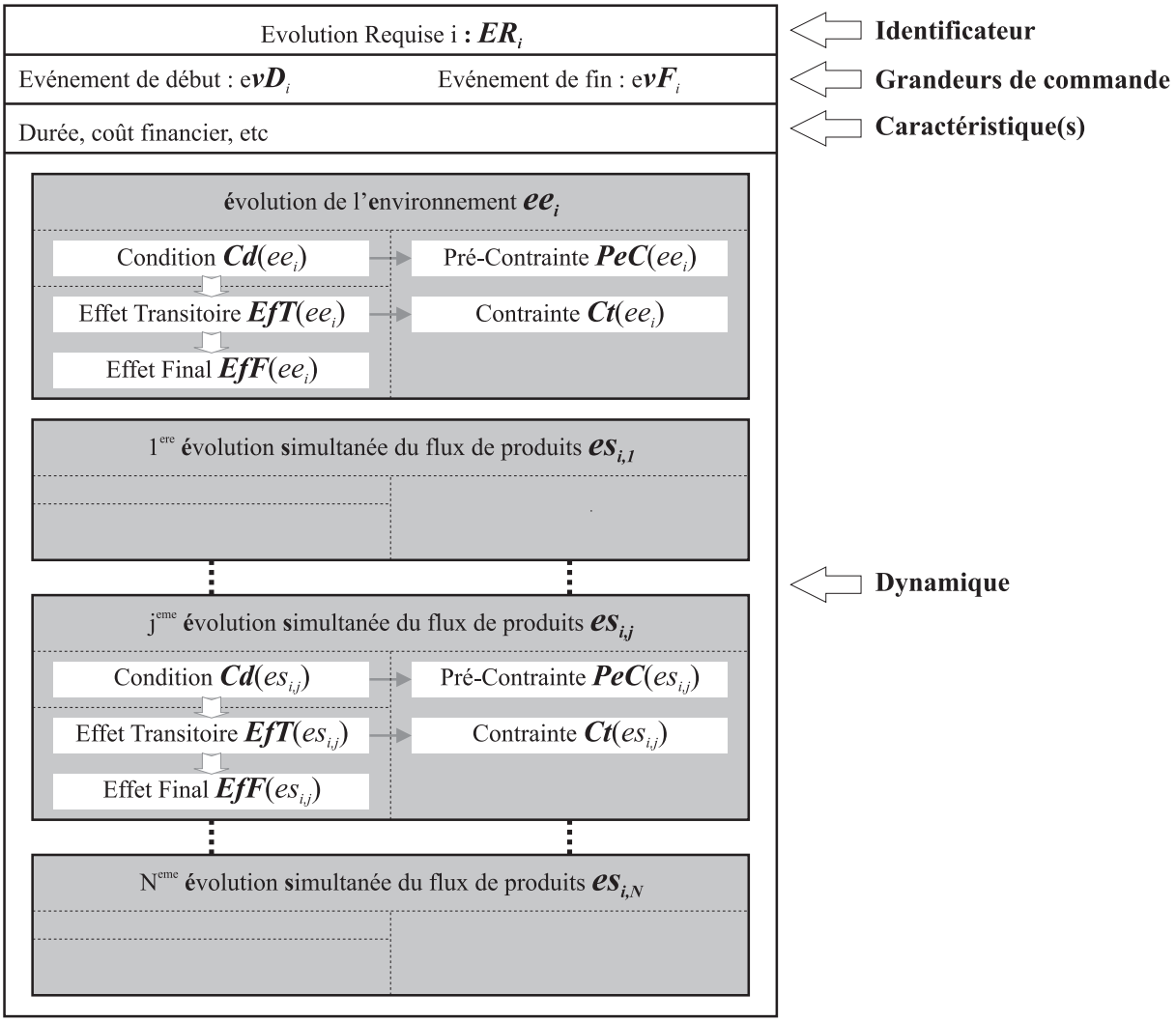


FIG. A.3 – Champ de données et notations du modèle générique d'une évolution requise.

L'ensemble des états depuis lesquels le comportement d'une évolution requise est autorisé est d'abord défini. Puis, l'effet sur l'état du système contrôlé du lancement d'une évolution requise est défini.

Définition 14 *L'ensemble des états initiaux, $Q_{It(CER_{i,k})}$, est l'ensemble des états depuis lesquels l'occurrence de l'événement de début de l'évolution induite $ER_{i,k}$ est autorisé et a pour comportement $CER_{i,k}$. Formellement, cet ensemble est représenté par :*

$$Q_{It(CER_{i,k})} = \{q \in Q_S / Cd(ee_i) \wedge PeC(ee_i) \bigwedge_{j \in J_k} [Cd(es_{i,j}) \wedge PeC(es_{i,j})] \bigwedge_{j \in \overline{J_k}} \neg Cd(es_{i,j}) = VRAIE\}$$

Définition 15 *Depuis un état appartenant à l'ensemble des états initiaux du comportement $CER_{i,k}$, $q \in Q_{It(CER_{i,k})}$, et après l'occurrence de l'événement de début de l'évolution induite $ER_{i,k}$, noté $d(ER_{i,k})$, le système contrôlé atteint un état q'_1.*

Les variables d'états dans cet état q'_1 ont pour valeurs celles dans l'état q modifiées par les effets transitoires $EfT(ee_i)$ et $EfT(es_{i,j})$ pour $j \in J_k$.

La fonction de transition partielle est définie par $q'_1 = \delta(d(ER_{i,k}), q)$.

$$q \xrightarrow{d(ER_{i,k})} q'_1$$

Avec un point de vue système de commande par la prise en compte des grandeurs de commande, l'occurrence de l'événement de début de l'évolution requise est connue ou non suivant les informations fournies par l'environnement. Quand il existe des informations permettant de connaître son occurrence, cet événement est noté evD_i. La fonction de transition s'écrit alors $q'_1 = \delta((evD_i), q)$.

$$q \xrightarrow{evD_i} q'_1$$

Durant une évolution requise et afin de toujours satisfaire les contraintes de sécurité et d'écologie, le système contrôlé et son environnement peuvent évoluer à condition de rester dans un ensemble d'états intermédiaires défini ci-dessous.

Définition 16 *L'ensemble des états intermédiaires, $Q_{Id(CER_{i,k})}$, est l'ensemble des états dans lesquels le système contrôlé et son environnement doivent être suite à l'occurrence de l'événement de début de l'évolution requise $ER_{i,k}$ ayant un comportement $CER_{i,k}$ et avant l'occurrence de l'événement de fin de l'évolution induite $ER_{i,k}$. Formellement, cet ensemble est représenté par :*

$$Q_{Id(CER_{i,k})} = \{q \in Q_S / EfT(ee_i) \wedge Ct(ee_i) \bigwedge_{j \in J_k} [EfT(es_{i,j}) \wedge Ct(es_{i,j})] = VRAIE\}$$

Depuis un état de cet ensemble d'états intermédiaires, la fin de l'évolution requise mène le système contrôlé et son environnement dans un état tel que défini ci-dessous.

Définition 17 *Depuis un état q'_2 appartenant à l'ensemble des états intermédiaires du comportement $CER_{i,k}$, $q'_2 \in Q_{Id(CER_{i,k})}$, et après l'occurrence de l'événement de fin de l'évolution requise $ER_{i,k}$, noté $f(ER_{i,k})$, le système contrôlé atteint un état q''.*

Les variables d'état dans cet état q'' ont pour valeurs celles dans l'état q'_2 modifiées par les effets finaux $EfF(ee_i)$ et $EfF(es_{i,j})$ pour $j \in J_k$.

La fonction de transition partielle est définie par $q'' = \delta(f(ER_{i,k}), q'_2)$.

$$q'_2 \xrightarrow{f(ER_{i,k})} q''$$

Avec un point de vue système de commande par la prise en compte des grandeurs de commande, l'occurrence de l'événement de fin de l'évolution requise est connue ou non suivant les informations fournies par l'environnement. Quand il existe des informations permettant de connaître son occurrence, cet événement est noté evF_i. La fonction de transition s'écrit alors $q'' = \delta((evF_i), q'_2)$.

$$q'_2 \xrightarrow{evF_i} q''$$

Annexe B

Deux Problèmes Multi-Flux de Produits

Le lecteur trouvera dans cette annexe, les deux cas particuliers liés à la problématique multi-flux, à savoir, la fabrication de produits avec des spécifications différentes et la fabrication par assemblage.

1 Produits avec des Spécifications Différentes

Afin de répondre à une demande de fabrication de produits avec des spécifications différentes, nous allons d'abord préciser la forme de cette demande. Puis, le principe général de résolution est présenté avant de l'interpréter avec les résultats obtenus aux chapitres précédents 8 et 9.

1.1 Caractérisation de la demande

Dans l'état initial du système contrôlé, considérons d'une part l'absence de produits dans le système de production et d'autre part une demande spécifiant, outre les objectifs de transformation des produits, des objectifs Ob_2 et Ob_3 à atteindre définis en terme d'arrêts en fin de cycle (spécification des états concernant donc les opérations de transitique et de préparation), des critères à optimiser, et les états physiques d'entrée et de sortie des produits pour les P types de produits :

l'état de sortie est défini par un ensemble d'objectifs $Ob_{1,x}$ avec $x \in [1, P]$ spécifiant chacun l'état de sortie d'un type de produits.

l'état d'entrée est spécifié de deux façons différentes suivant le nombre de flux en entrée :

1. P flux de produits en entrée, la demande spécifie P états physiques d'entrée $q_{1,x,0}$ avec $x \in [1, P]$.
2. un seul flux de produits en entrée, l'état $q_{1,0}$ spécifie l'état physique d'entrée des produits.

Un unique flux de produits en entrée correspond soit à un état d'entrée des produits qui est réellement le même, soit il existe un indéterminisme sur l'état d'entrée. Cette deuxième situation est celle où l'environnement approvisionne le système de production sans indiquer le type des produits fournis. Pour le système de production, tous les produits sont alors identiques. La différenciation de l'état physique des produits résultera d'une ou de plusieurs actions d'information

dont le rôle est d'informer le niveau coordination de l'état réel du flux de produit (cas de la plate-forme SAPHIR par exemple).

La demande caractérisée, le principe de conception d'une loi de commande en réponse à cette demande est présenté dans la section suivante.

1.2 Principe

Afin d'exposer les concepts clefs liés à la conception d'une loi de commande visant à fabriquer des produits avec des spécifications différentes, le nombre de produits différents est limité à deux. La résolution d'un problème avec T types de produits sera une généralisation de ces concepts. Ainsi, dès cette section, la demande spécifie uniquement deux types de produits, notés T1 et T2. Un flux de produits est spécifique à la fabrication des produits T1 et un autre flux à la fabrication des produits T2.

La première solution permettant de répondre à cette demande revient à fabriquer $N1$ produits T1, puis $N2$ produits T2 et de nouveau des produits T1 et ainsi de suite. Cette première solution nécessite de commuter de la fabrication de produits T1 à celle de produits T2. La commutation est réalisée entre la fin de la fabrication du dernier produit T1 d'une série et avant le début de la fabrication du premier produit T2 d'une autre série comme le présente la Figure B.1. La taille et le nombre de séries de produits T1 et T2 à fabriquer sont imposés par la demande ou fonction du type de produits bruts qui arrivent.

La solution présentée dans la Figure B.1 n'est pas toujours optimale vis-à-vis du temps total de fabrication. Ce dernier pourrait en effet être réduit en imbriquant la loi de commande de commutation avec la fabrication du dernier produit T1 mais également avec la fabrication du premier produit T2. Quand ces deux imbrications sont possibles, il peut alors également y avoir recouvrement entre la fabrication des produits T1 et des produits T2 (cf. Figure B.2). Des produits T1 et T2 sont alors présents au même moment dans le système de production.

Le même processus d'imbrication est appliqué lors de la commutation de la fabrication de produits T2 à la fabrication de produits T1. Les lois de commutation ainsi que leur imbrication définissent un ensemble de contraintes de commutation entre les lois de commande cycliques destinées respectivement à fabriquer des produits T1 et T2 comme le montre la Figure B.3.

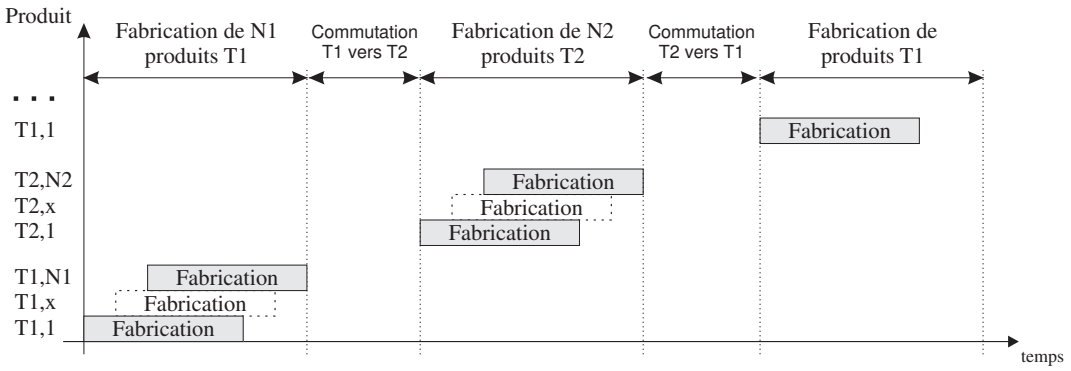


FIG. B.1 – Fabrication alternée de produits T1 et de produits T2.

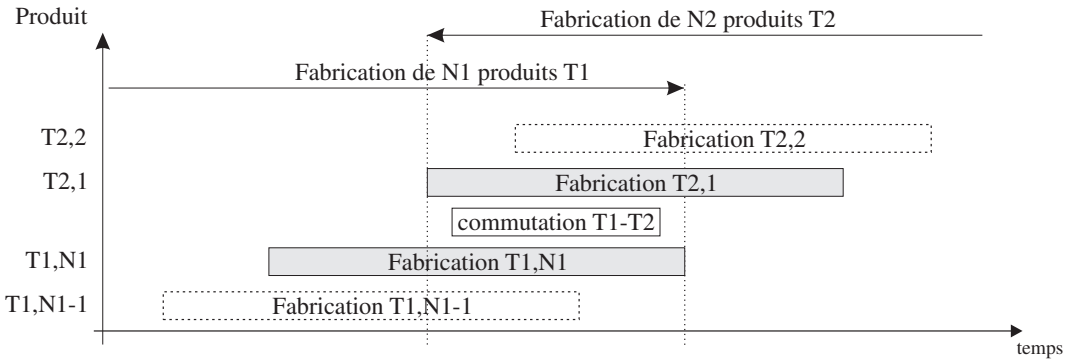


FIG. B.2 – Fabrication en parallèle de produits T1 et de produits T2.

Le principe énoncé ci-dessus est présenté dans la section suivante avec les résultats obtenus aux chapitres 8 et 9.

1.3 Analyse du principe multi-produits

Selon le principe énoncé, la recherche de la solution optimale passe d'abord par la génération de la loi de commande pour fabriquer des produits T1 et celle pour des produits T2, plus la génération de la loi de commande de commutation de la fabrication de produits T1 à des produits T2. Une fois les chemins spécifiant ces lois de commande générés, il reste à les imbriquer.

Ainsi, nous commençons par analyser la fabrication alternative de produits T1 et T2 avec les concepts développés dans les chapitre 7 et 8 relatifs à la condition de cycle et à la génération d'un chemin Ch_{Op} pour fabriquer un produit. Puis, nous détaillerons l'imbrication.

Afin de fabriquer un produit T1, un chemin $Ch_{T1,Op}$ est construit par application de la fonction Γ. Par hypothèse, ce chemin vérifiant la condition de cycle, son état initial qui est noté $q_{T1,3,0}$ est également son état final. Afin d'optimiser la fabrication consécutive de plusieurs produits T1, le chemin $Ch_{T1,OpC}$ satisfaisant toujours la condition de cycle est construit suivant la méthode présentée au chapitre 8. Ainsi, l'état atteint à la fin de la fabrication de $N1$ produits T1 est l'état initial $q_{T1,3,0}$.

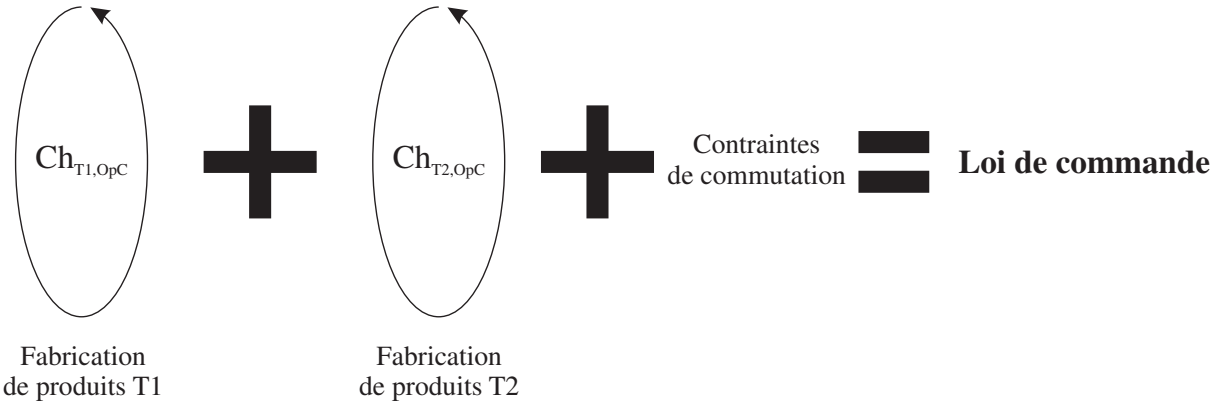


FIG. B.3 – Principe de construction d'une loi de commande multi-flux de produits.

Les mêmes hypothèses et les mêmes notations sont adoptées pour la fabrication des produits T2. Ainsi, pour ces produits, il est d'abord défini un chemin $Ch_{T2,Op}$ puis un chemin $Ch_{T2,OpC}$ dont l'état initial et l'état final est noté $q_{T2,3,0}$.

Ainsi, la commutation de la fabrication de produits T1 à des produits T2 correspond au passage de l'état $q_{T1,3,0}$ à l'état $q_{T2,3,0}$, et inversement pour la commutation T2-T1. Les chemins spécifiant les lois de commande de commutation sont notés $Ch_{T1,T2}$ et $Ch_{T2,T1}$ comme présenté dans la Figure B.4.

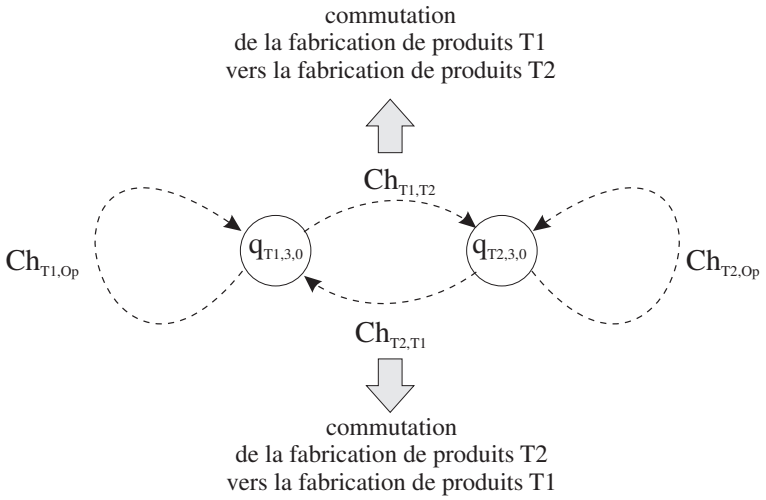


FIG. B.4 – problème lié à la fabrication de plusieurs types de produits.

D'après le principe représenté par la Figure B.2, l'optimisation du temps total de fabrication requiert alors de se focaliser sur l'imbrication des chemins suivants : $Ch_{T1,Op}$ destiné à fabriquer le dernier produits T1, $Ch_{T1,T2}$ qui défini la commutation T1-T2, et $Ch_{T2,Op}$ destiné à fabriquer le premier produit T2. L'imbrication qui concerne ici les trois chemins nécessitera le développement d'un algorithme d'imbrication spécifique mais néanmoins toujours basé sur les mécanismes de parallélisation et d'insertion. Les trois chemins de la Figure B.5 représentés sous forme de graphes de précédence sont le résultat de l'imbrication appliquée aux deux couples de chemins $\{Ch_{T1,Op}; Ch_{T1,T2}\}$ et $\{Ch_{T1,T2}; Ch_{T2,Op}\}$, mais il s'avère nécessaire de rechercher en plus les contraintes entre les chemins $\{Ch_{T1,Op}; Ch_{T2,Op}\}$.

Pour terminer, afin de pouvoir commuter de la fabrication des produits T2 à celle des produits T1 de façon optimale, il est également nécessaire de rechercher l'imbrication des che-

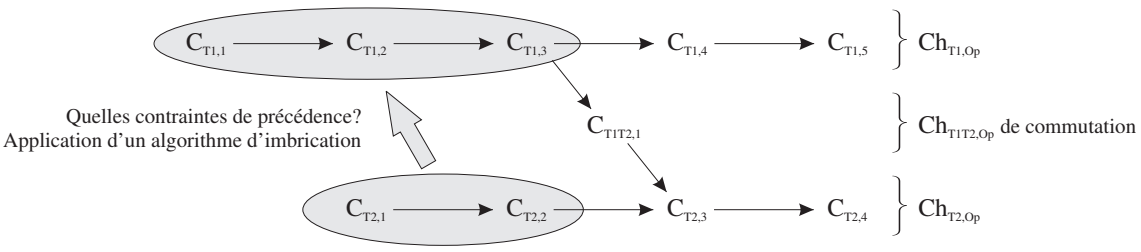


FIG. B.5 – Imbrication du chemin de commutation et des chemins liés aux produits T1 et T2.

mins $Ch_{T2,Op}$, $Ch_{T2,T1}$ et $Ch_{T1,Op}$. Les contraintes de commutation entre la loi de commande cyclique destinée à fabriquer les produits T1 et la loi de commande cyclique destinée à fabriquer les produits T2, et inversement, seront alors traduites en réseaux de Petri comme les lois de commande cycliques.

Après l'étude de la fabrication de produits dont les spécifications sont différentes, nous proposons dans la section suivante la conception de lois de commande capable de fabriquer des produits par assemblage.

2 Assemblage de Produits

Cette section vise à définir les concepts clefs relatifs à la conception d'une loi de commande destinée à la fabrication de produits résultant d'un assemblage. Après avoir caractérisé une demande d'assemblage, le principe de conception est détaillé avant de l'analyser à partir des résultats des chapitres 8 et 9.

2.1 Caractérisation de la demande

La demande spécifie toujours un état initial et un état final du système contrôlé qui sont identiques et pour lesquels aucun produit n'est présent dans le système de production.

En sus du nombre de produits à fabriquer et des critères d'optimisation, il est spécifié par la demande l'état physique d'entrée des produits constituant le produit final dont l'état de sortie est lui aussi spécifié. Ainsi, les états physiques d'entrée et de sortie des flux de produits sont spécifiés comme suit :

l'état d'entrée est donné par P états physiques d'entrée $q_{1,x,0}$ avec $x \in [2, P]$, où P représente le nombre de composants constituant le produit assemblé.

l'état de sortie est spécifié par un objectif Ob_1 qui définit d'une part l'opération d'assemblage et d'autre part les transformations susceptibles d'être requises après l'assemblage.

Dans le cadre de cette section, nous limiterons à deux le nombre de produits nécessaires à l'assemblage d'un produit final. Ils sont notés T1 et T2. La demande correspondant à la représentation de la Figure B.6 spécifie un état d'entrée $q_{1,T1,0}$ du produit T1 et un état $q_{1,T2,0}$ du produit T2 ainsi qu'un objectif Ob_1 définissant l'état de sortie du produit final, noté T12.

Connaissant la forme de la demande relative à un assemblage, la section suivante expose le principe de conception d'une loi de commande en réponse à ce type de demande.

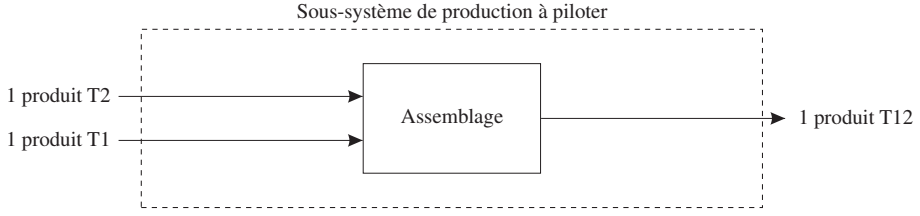


FIG. B.6 – Produits en entrée et en sortie pour une fabrication par assemblage.

2.2 Principe

Afin d'exploiter le résultat du chapitre 8 pour l'assemblage, les séquences d'opérations à exécuter pour chacun des produits (T1, T2, et T12) seront déterminées de manière indépendante sans considération, par exemple pour un produit T1, de la présence des produits T2 et T12. Ainsi, le principe de conception d'une loi de commande dans un but d'assemblage repose sur les hypothèses présentées dans la Figure B.7. Ces deux hypothèses sont basées sur la possibilité de placer un des deux produits dans l'état requis par l'opération d'assemblage, puis d'amener le second produit avant de réaliser l'assemblage.

Si les deux propositions ci-dessous sont réalisables, alors la conception d'une loi de commande répondant à la demande considérée dans cette section se compose de six étapes :

1. choisir à partir du modèle du système contrôlé et de son environnement une opération d'assemblage répondant à l'objectif Ob_1. Il est ensuite extrait de cette opération l'objectif à atteindre tel que depuis un état satisfaisant cet objectif, l'exécution de l'opération d'assemblage est autorisée.
2. séparer cet objectif en deux : une partie relative à l'état (physique et position) du produit T1 et une deuxième partie relative cette fois-ci à l'état (physique et position) du produit T2.
3. construire un chemin depuis l'état initial $q_{3,0}$ jusqu'à un état satisfaisant les objectifs relatifs au produit T1. Il est ensuite construit un deuxième chemin pour le produit T2. Et enfin, un troisième chemin est construit pour passer de l'état après l'opération d'assemblage à un état satisfaisant les objectifs Ob_1, Ob_2 et Ob_3 imposés par la demande.
4. imbriquer les deux chemins relatifs respectivement au produit T1 et au produit T2. Il existe deux solutions pour imbriquer les chemins : soit commencer par le produit T1 et imbriquer le chemin du produit T2, soit le contraire. S'il est possible de contrôler l'arrivée des produits T1 et T2, seule la solution optimale vis-à-vis des critères considérés sera retenue.
5. assembler les trois chemins avec l'opération d'assemblage. Ceci consiste à ajouter une contrainte de précédence entre la dernière opération du chemin relatif à T1 et l'opération d'assemblage, et à faire de même pour T2. De surcroît, une contrainte de précédence est ajoutée entre l'opération d'assemblage et la première opération du chemin relatif au produit T12 assemblé.
6. la contrainte de cycle étant satisfaite pour le chemin global résultant de l'étape précédente ($q_{3,0}$ état initial et état final), il est alors possible de fabriquer un deuxième produit T12 par une nouvelle application de la loi de commande spécifiée par ce chemin. Dans un but d'optimisation, le mécanisme d'imbrication sera alors appliqué.

L'interprétation du principe énoncé limité aux étapes 1 à 4 est proposée dans la section suivante.

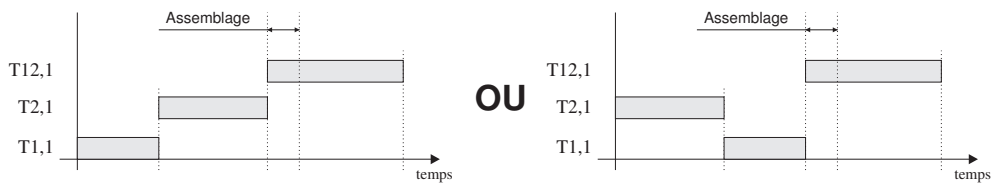


FIG. B.7 – Hypothèses sur la fabrication des produits dans le contexte d'un assemblage.

2.3 Analyse du principe pour l'assemblage

Les analyses données ici sont d'une part celles de l'hypothèse illustrée par la Figure B.7 et d'autre part celles des étapes 1 à 4 du principe de conception.

La Figure B.8 représente l'hypothèse formulée sur la possibilité d'amener un produit en position pour l'assemblage puis le suivant. La conséquence de cette hypothèse est l'existence d'une solution qui est une séquence d'opérations. Cette séquence est le résultat de la mise bout à bout des deux chemins relatifs aux produits T1 et T2 dans un ordre quelconque, de l'opération d'assemblage et enfin du chemin relatif au produit T12 après assemblage.

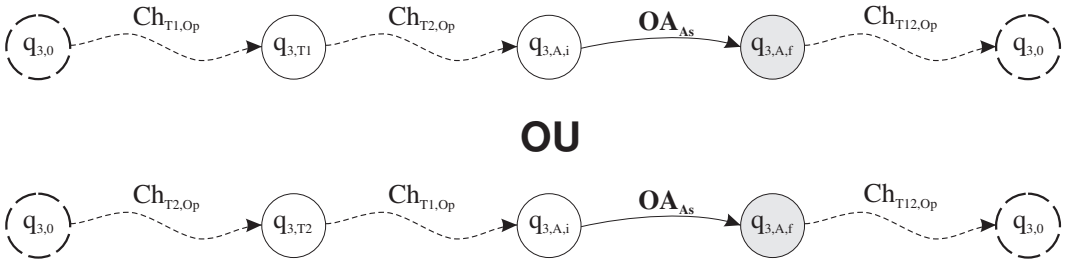


FIG. B.8 – Analyse du principe de conception d'une loi de commande pour un assemblage.

L'interprétation des étapes 1 à 4 est donnée par la Figure B.9. Les conditions et les pré-contraintes de l'opération d'assemblage choisie servent lors de l'étape 2 à spécifier les objectifs $\{Ob_{T1,1}; Ob_{T1,2}; Ob_{T1,3}\}$ pour le produit T1 ainsi que les objectifs $\{Ob_{T2,1}; Ob_{T2,2}; Ob_{T2,3}\}$ pour le produit T2.

L'étape 3 se résume alors à appliquer trois fois la fonction Γ. Le résultat de $\Gamma(q_{(1,T1,0)}; Ob_{(T1,1)}; q_{(3,0)}; Ob_{(T1,2)}; Ob_{(T1,3)})$ est le chemin $Ch_{T1,OP}$ relatif au produit T1. Le chemin $Ch_{T2,Op}$ relatif au produit T2 est quant à lui le résultat de $\Gamma(q_{(1,T2,0)}; Ob_{(T2,1)}; q_{(3,0)}; Ob_{(T2,2)}; Ob_{(T2,3)})$. Enfin, le troisième et dernier chemin est le résultat de $\Gamma(q_{(1,T12,0)}; Ob_{(1)}; q_{(3,A,F)}; Ob_{(2)}; Ob_{(3)})$ où l'état $q_{3,A,F}$ est l'état atteint suite à l'opération d'assemblage. L'état $q_{(1,T12,0)}$ qui est l'état physique initial du produit T12 est identique à l'état $q_{3,A,F}$ mais limité aux variables d'état physique. Enfin, les objectifs Ob_1, Ob_2 et Ob_3 sont imposés par la demande.

L'étape 4 consiste à imbriquer les chemins $Ch_{T1,OP}$ et $Ch_{T2,Op}$. Si les deux propositions formulées dans le § 2.2 ne sont pas conjointement réalisables, cette quatrième étape se limitera à imbriquer uniquement un chemin par rapport à l'autre en fonction de la proposition réalisable.

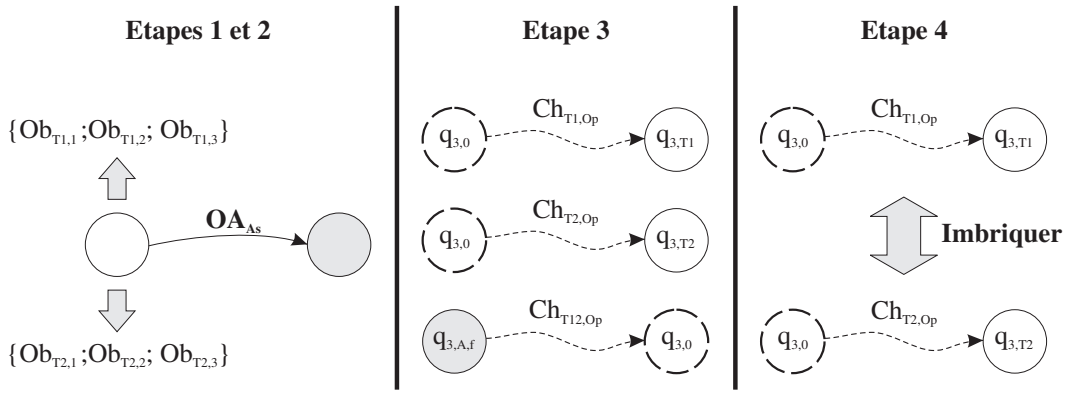


FIG. B.9 – Étape de conception d'une loi de commande à des fins d'assemblage.

Annexe C

Modèle Complet du Système d'Approvisionnement

Le lecteur trouvera dans cette annexe, le modèle complet du cas d'étude de la partie 4 ; le système d'approvisionnement de la plate forme SAPHIR.

Avant de donner ce modèle, précisons davantage la notation utilisée. Les expressions qui sont écrites entre parenthèses sont des expressions logiques, vraies ou fausses. Une expression sans parenthèse désigne l'affectation d'un valeur particulière à une variable d'état.

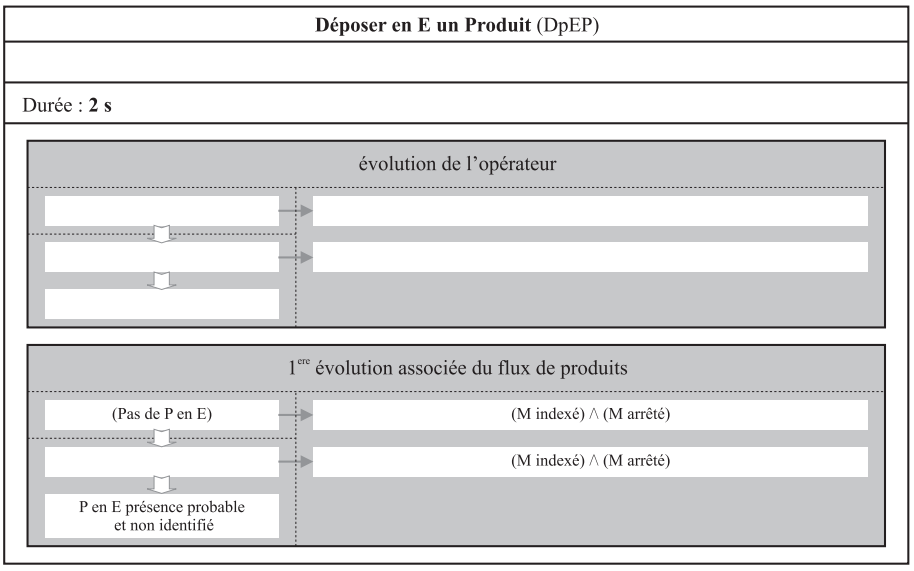


FIG. C.1 – Évolution requise Déposer un produit en E.

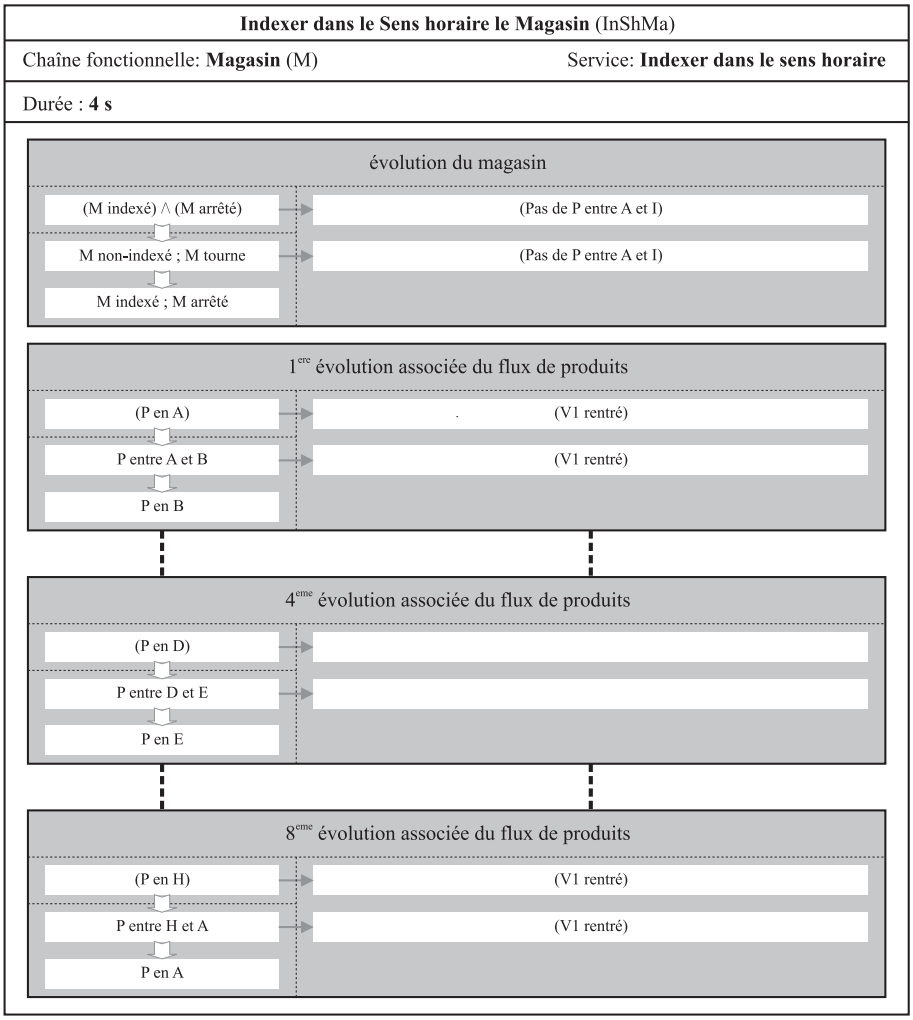


FIG. C.2 – Opération d'action Indexer dans le sens horaire le magasin.

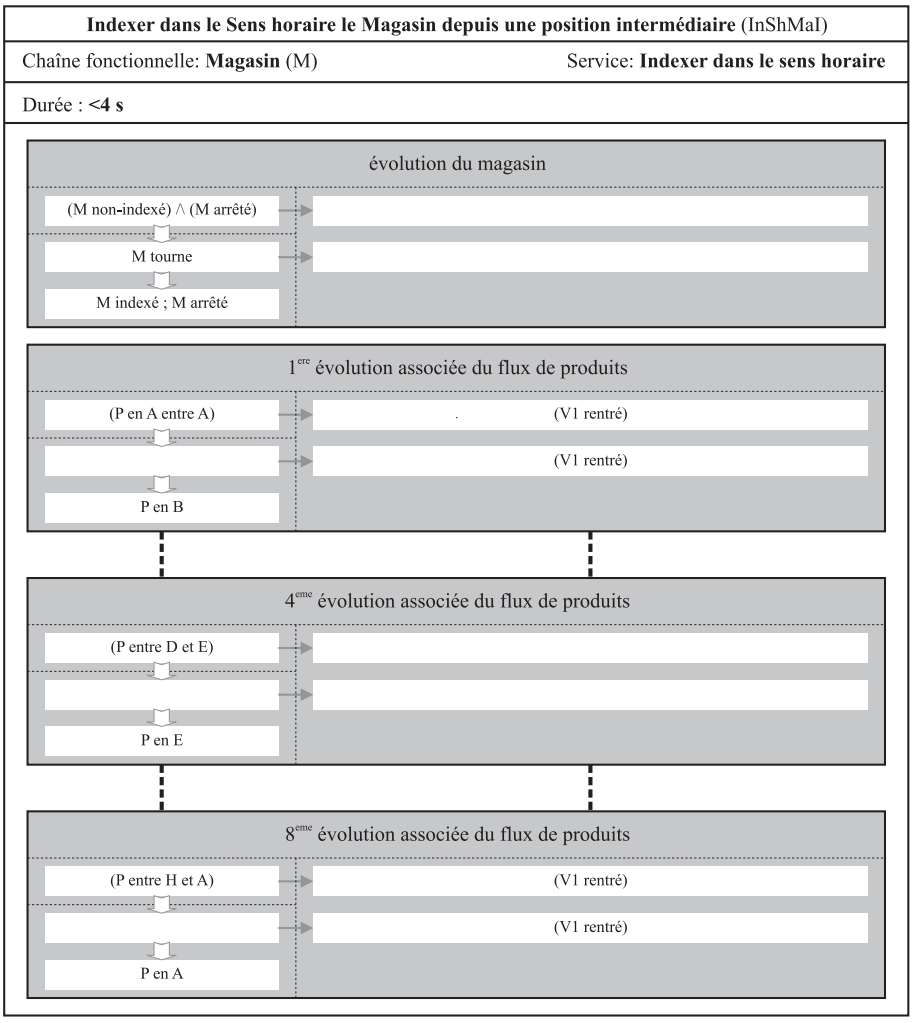


FIG. C.3 – Opération d'action Indexer dans le sens horaire le magasin depuis une position non-indexée.

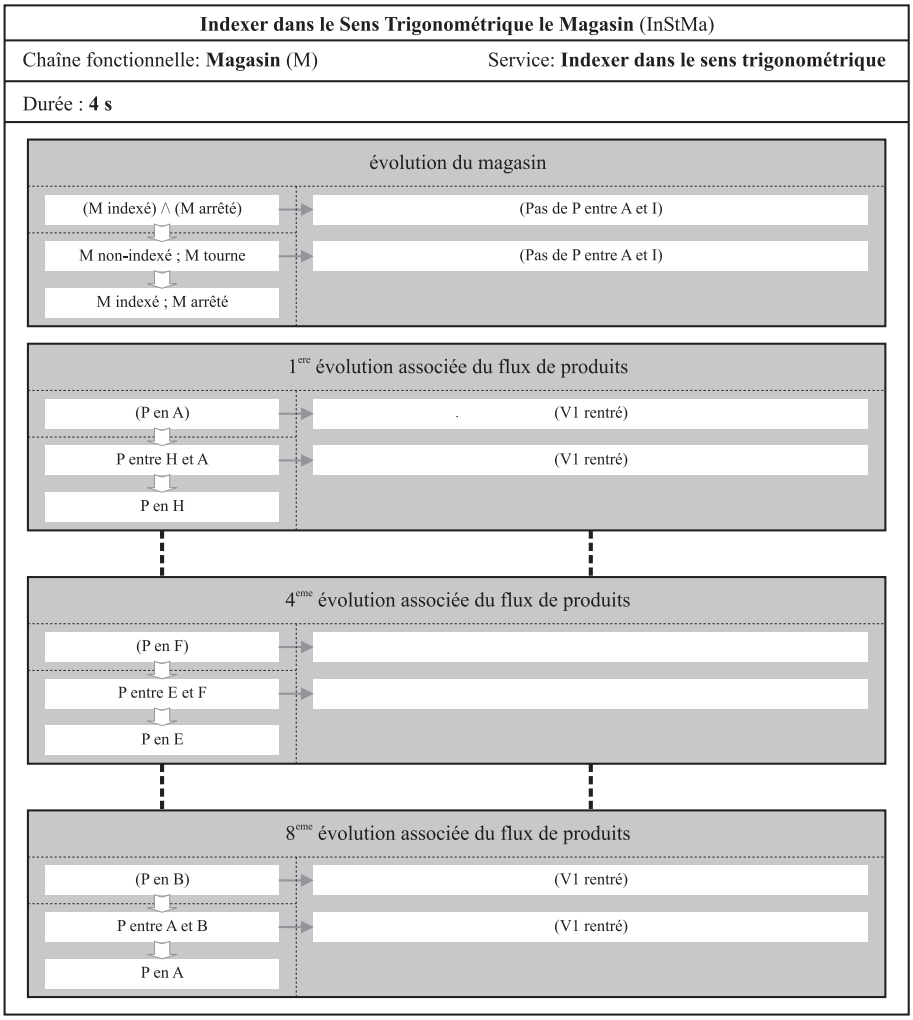


FIG. C.4 – Opération d'action Indexer dans le sens trigonométrique le magasin.

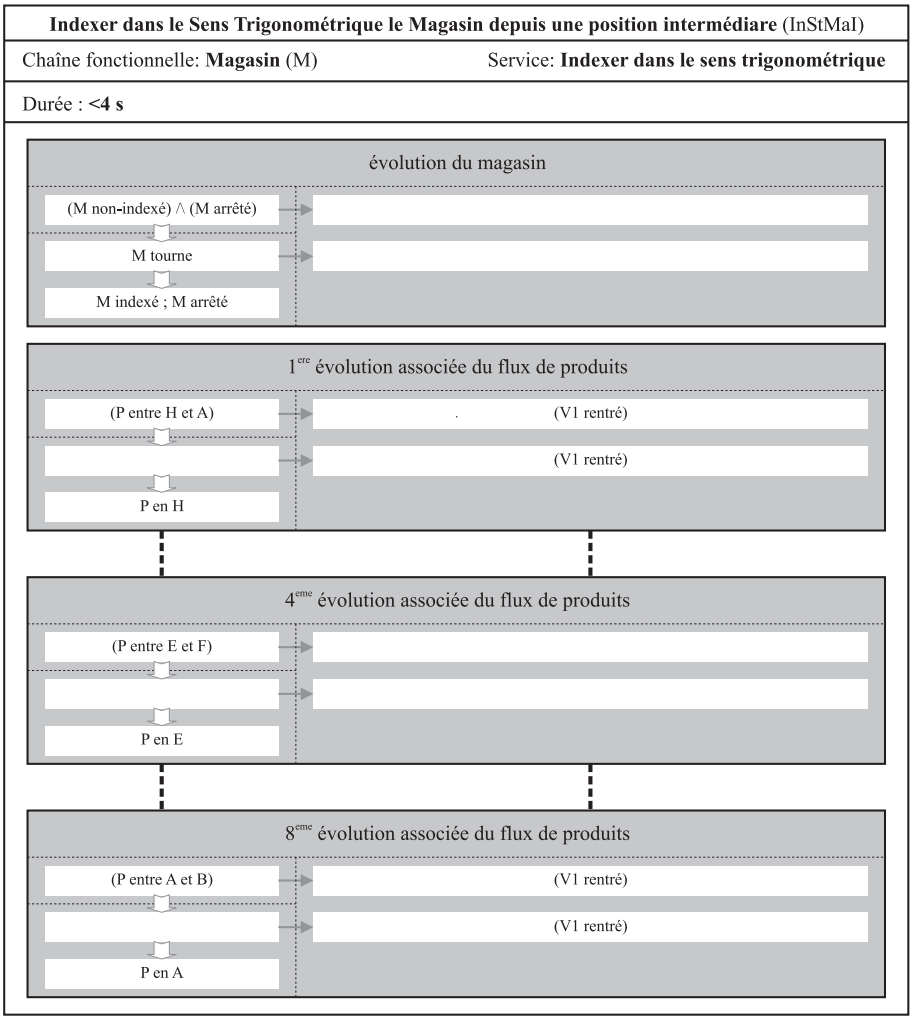


FIG. C.5 – Opération d'action Indexer dans le sens trigonométrique le magasin depuis une position non-indexée.

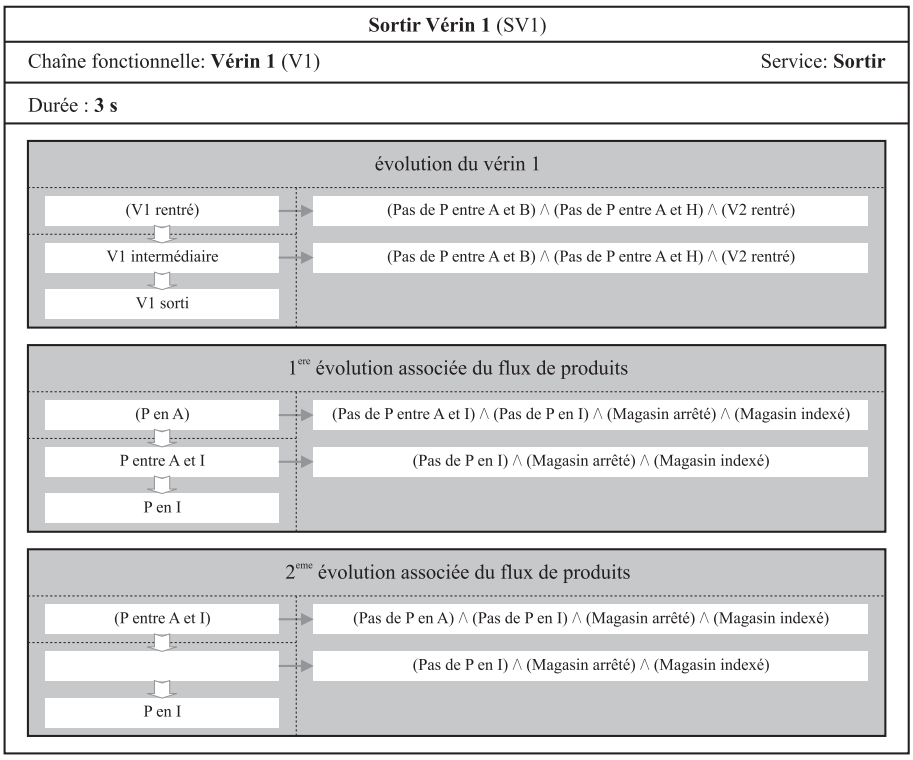


FIG. C.6 – Opération d'action Sortir vérin 1.

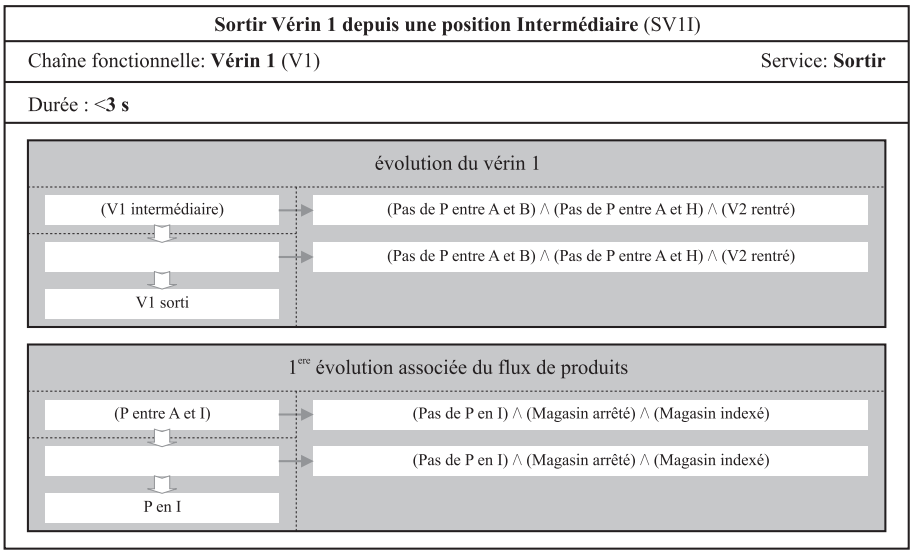


FIG. C.7 – Opération d'action Sortir vérin 1 depuis une position intermédiaire.

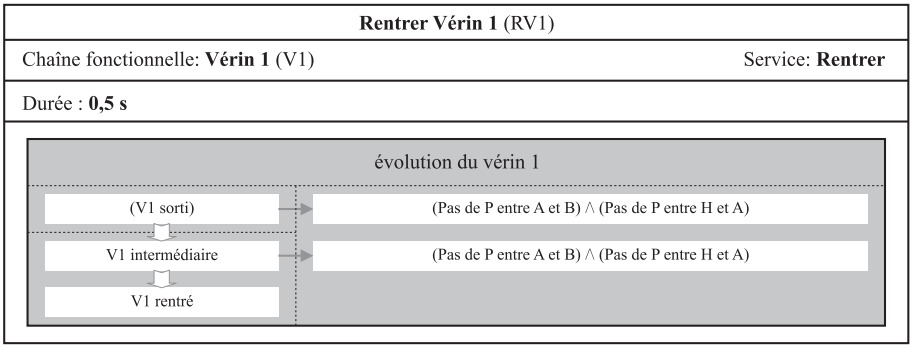


FIG. C.8 – Opération d'action Rentrer vérin 1.

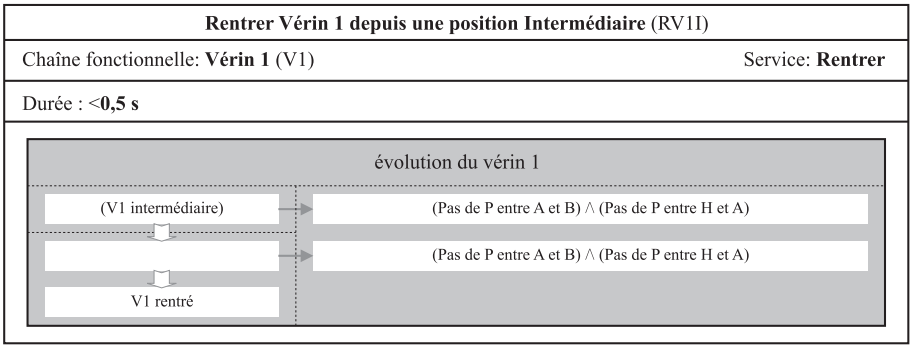


FIG. C.9 – Opération d'action Rentrer vérin 1 depuis une position intermédiaire.

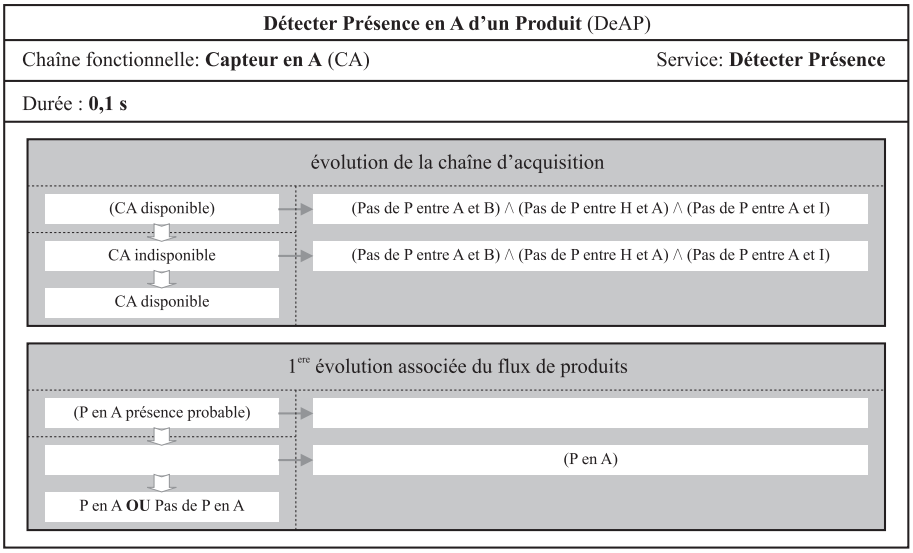


FIG. C.10 – Opération d'information Détecter présence d'un produit en A.

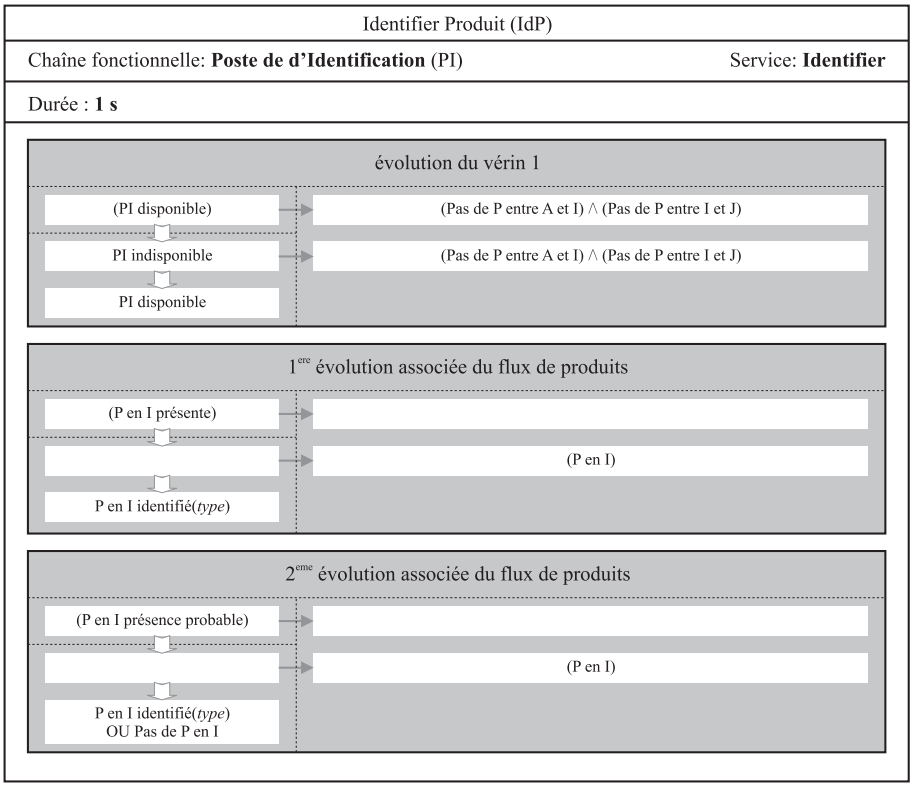


FIG. C.11 – Opération d'information Identifier un produit.

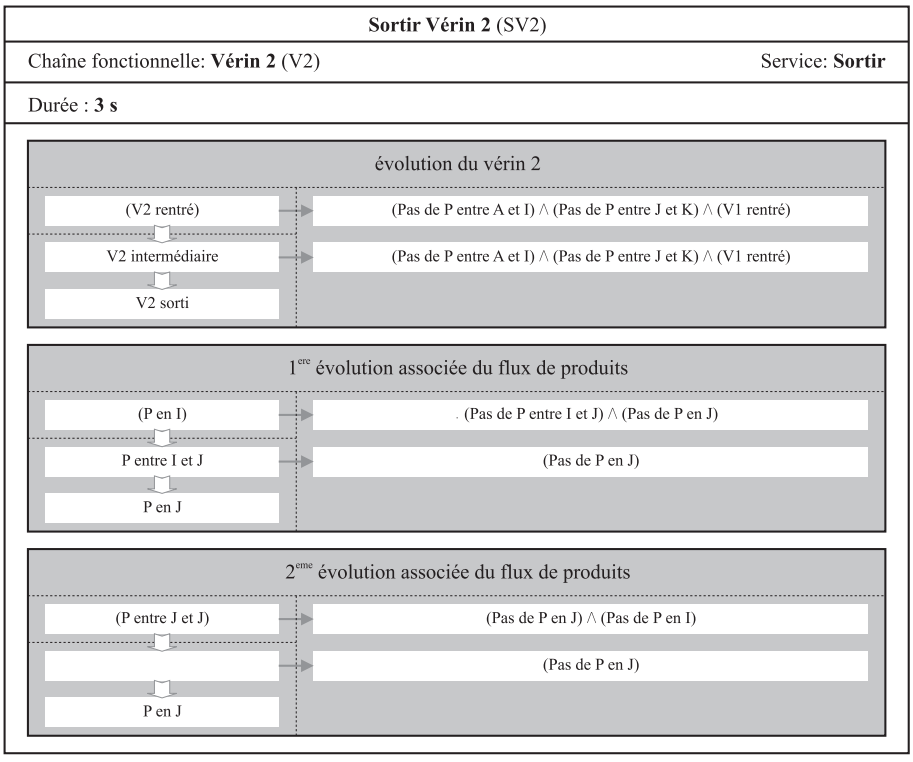


FIG. C.12 – Opération d'action Sortir vérin 2.

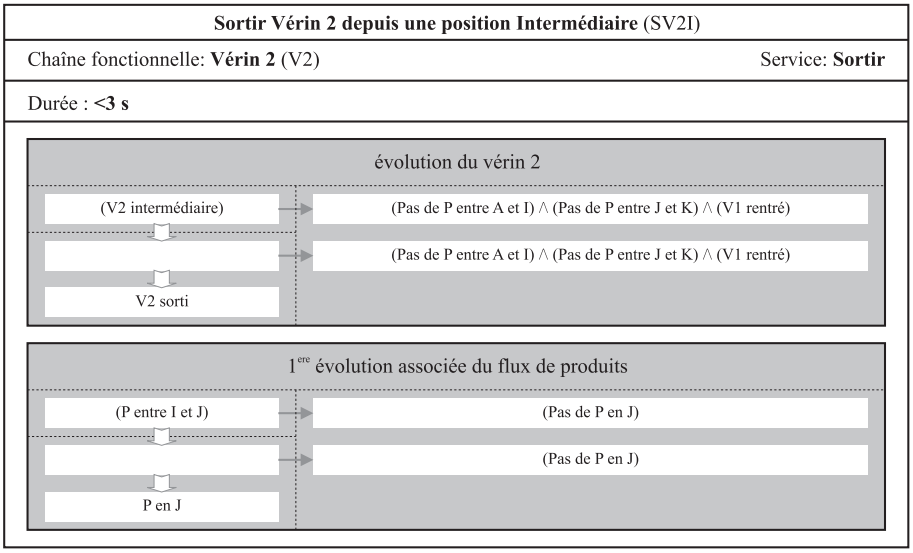


FIG. C.13 – Opération d'action Sortir vérin 2 depuis une position intermédiaire.

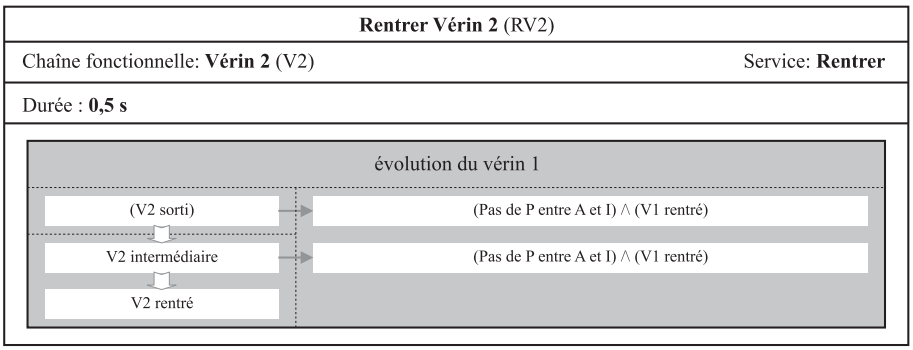


FIG. C.14 – Opération d'action Rentrer vérin 2.

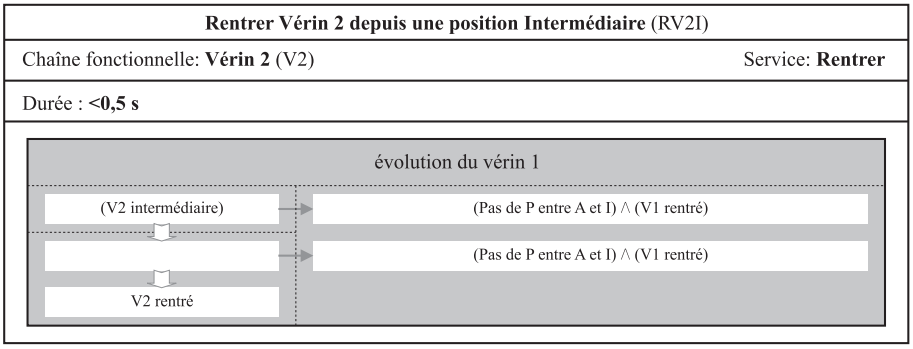


FIG. C.15 – Opération d'action Rentrer vérin 2 depuis une position intermédiaire.

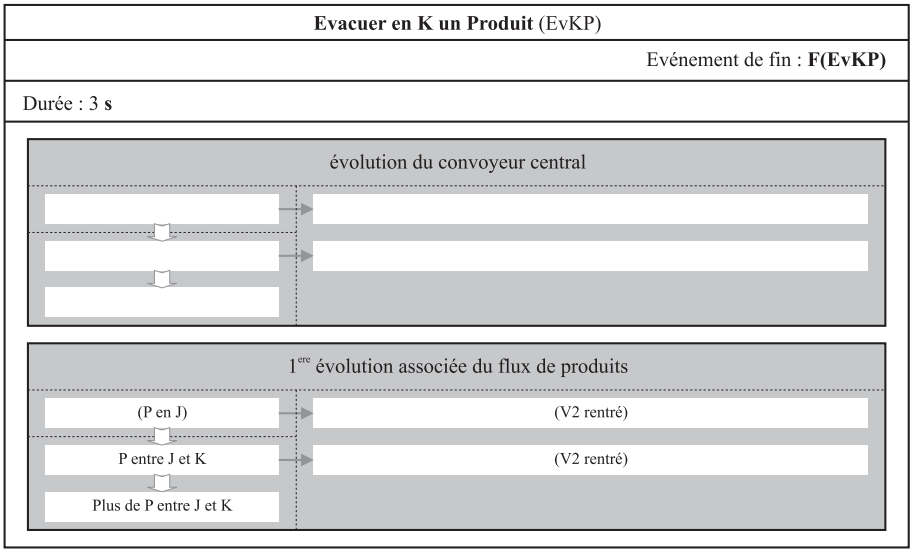


FIG. C.16 – Évolution requise Évacuer un produit de la position J vers la position k.

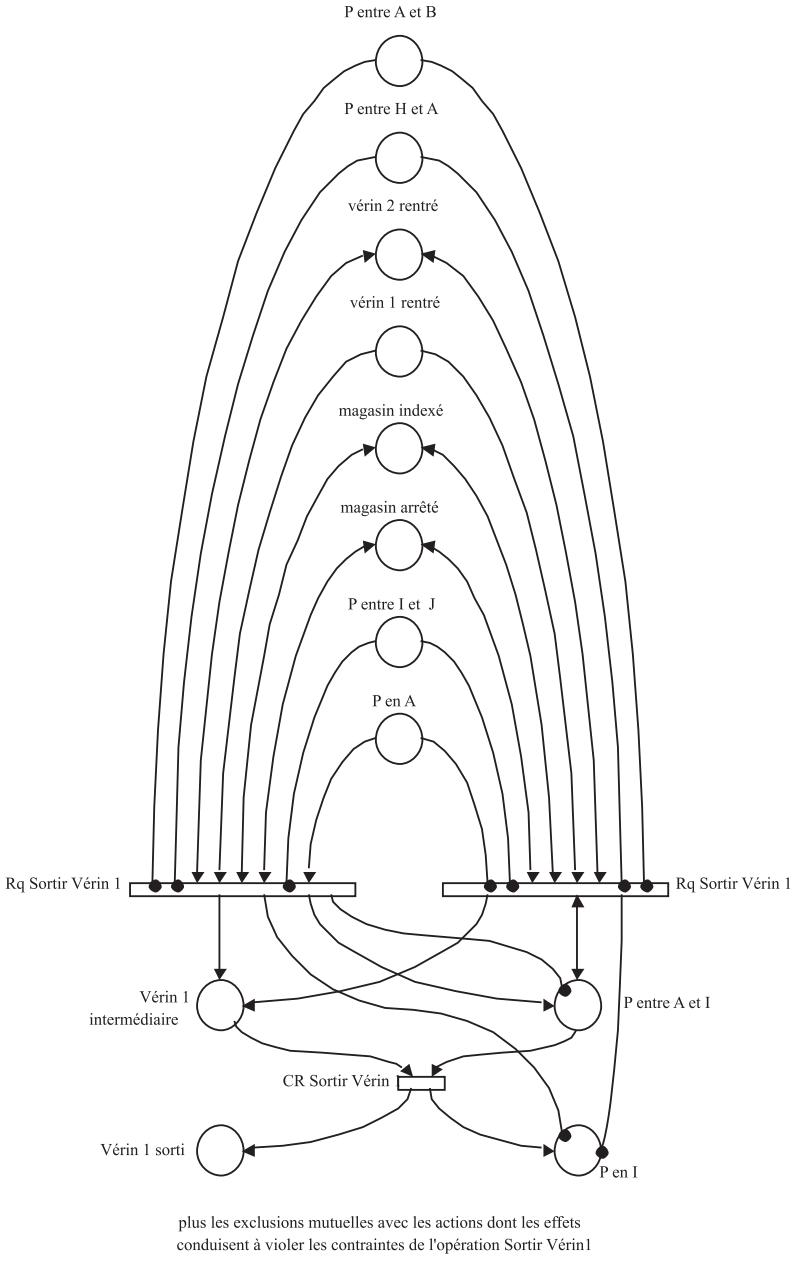


FIG. C.17 – Représentation avec l'outil réseaux de Petri des comportements 1 et 2 sans évolution associée du flux de produit de l'opération d'action Sortir vérin 1.

Annexe D

Résultats de l'Algorithme de Synthèse

Le lecteur trouvera dans cette annexe, les résultats de chacune des étapes de l'algorithme de synthèse appliqué, dans le chapitre 12, au système d'approvisionnement de la plate forme SAPHIR.

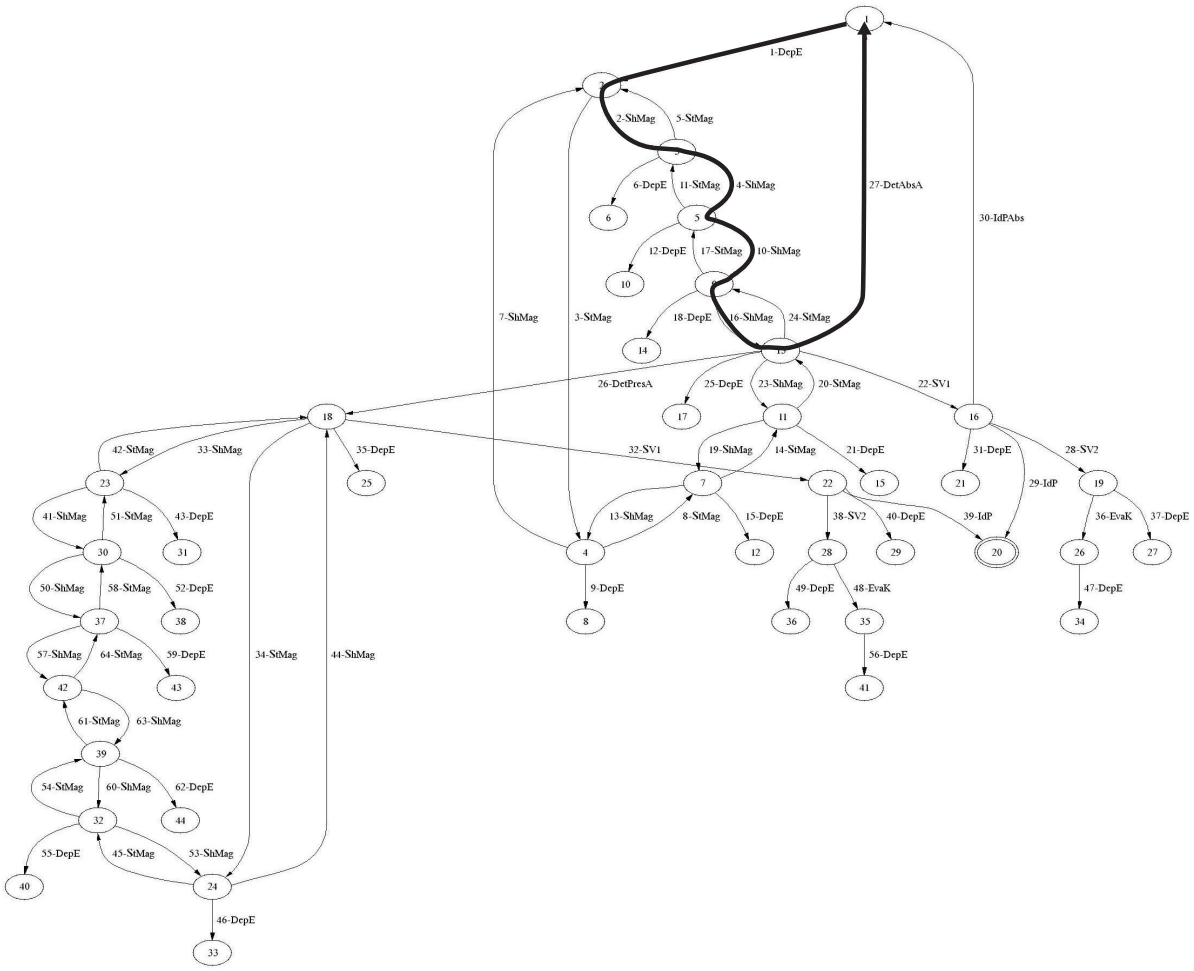


FIG. D.1 – Espace d'états atteignables par les opérations de transitique pour les produits absents.

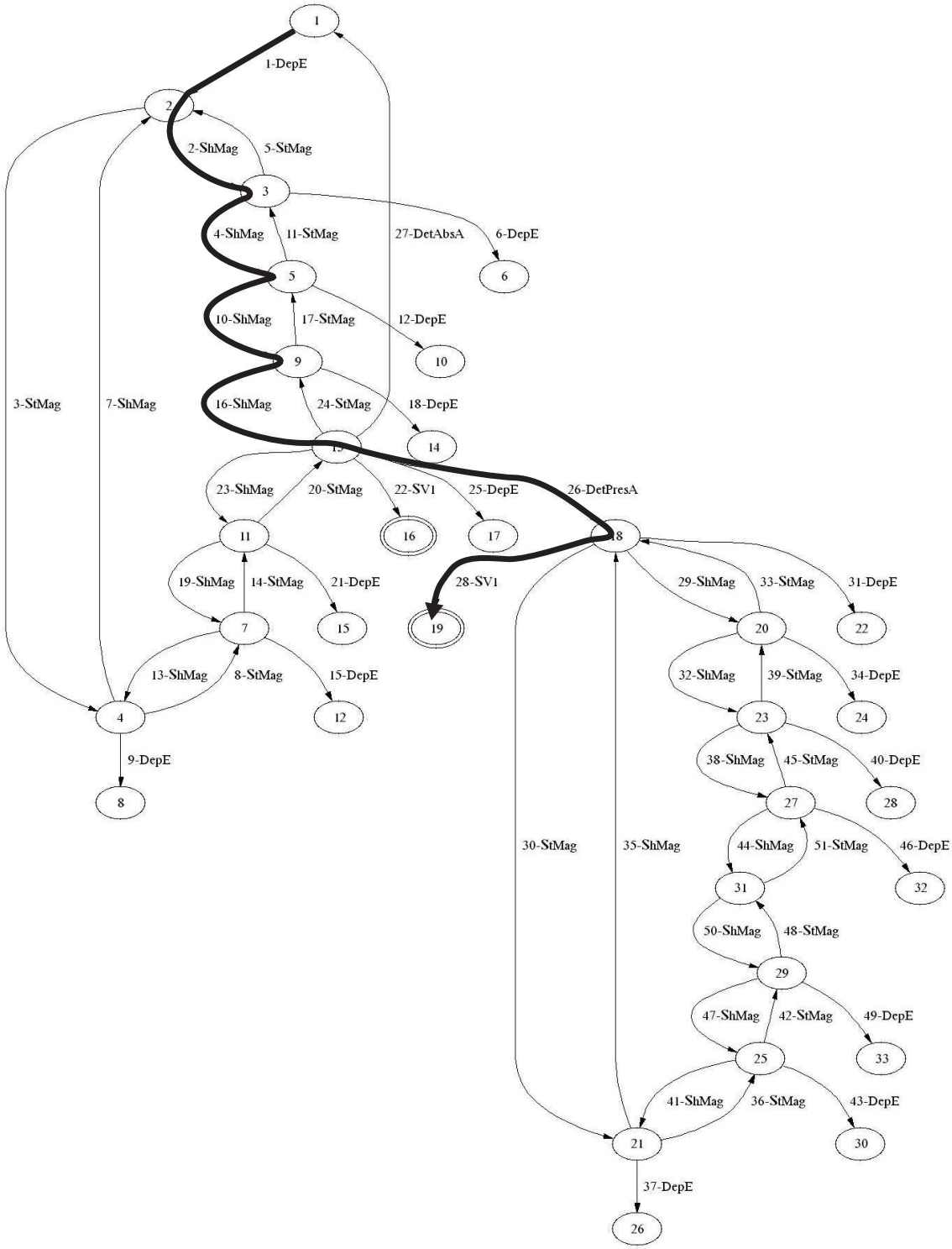


FIG. D.2 – Espace d'états atteignables par les opérations de transitique pour les produits présents avant l'identification.

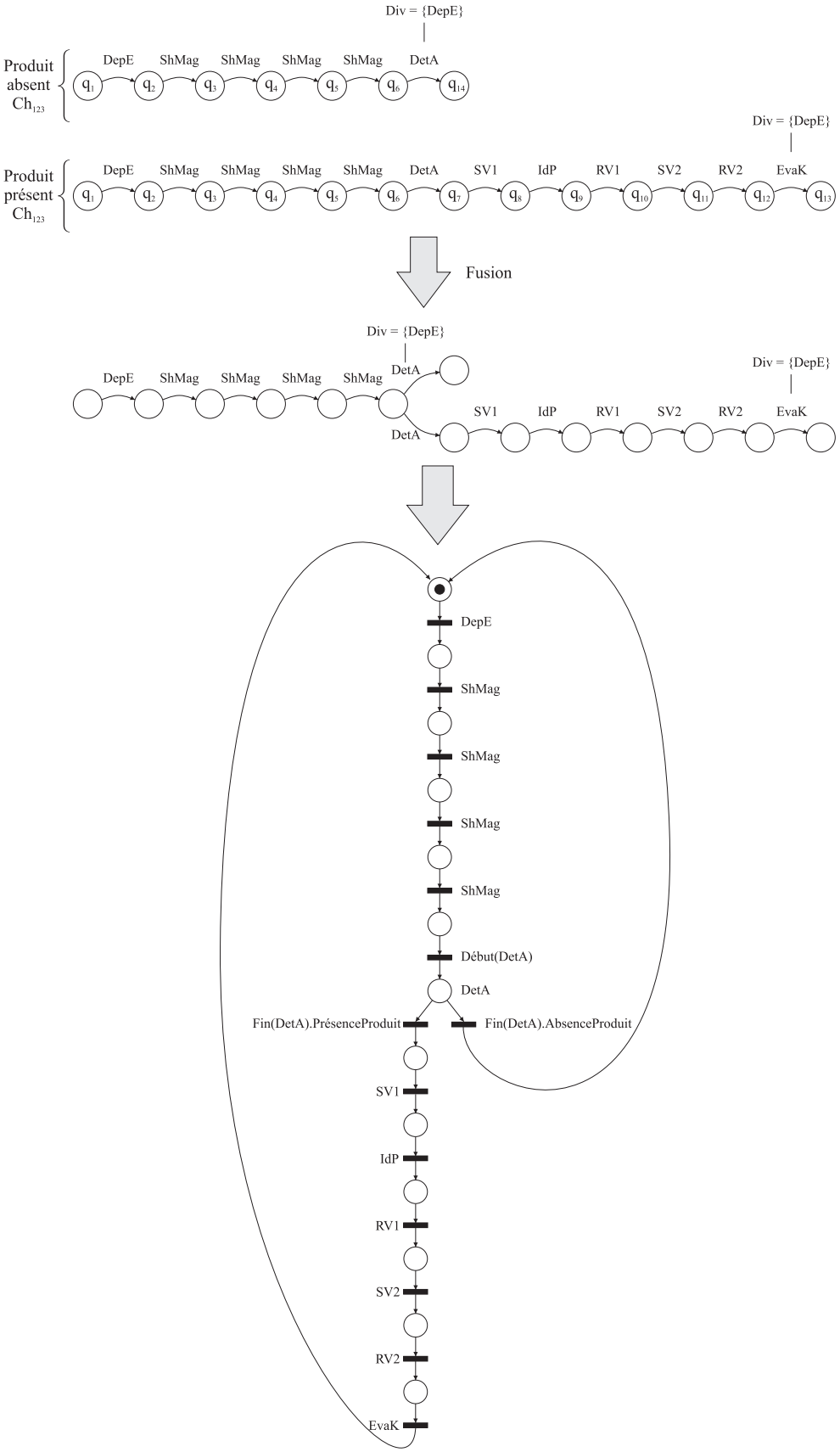


FIG. D.3 – Loi de commande sans parallélisme d'exécution ni insertion.

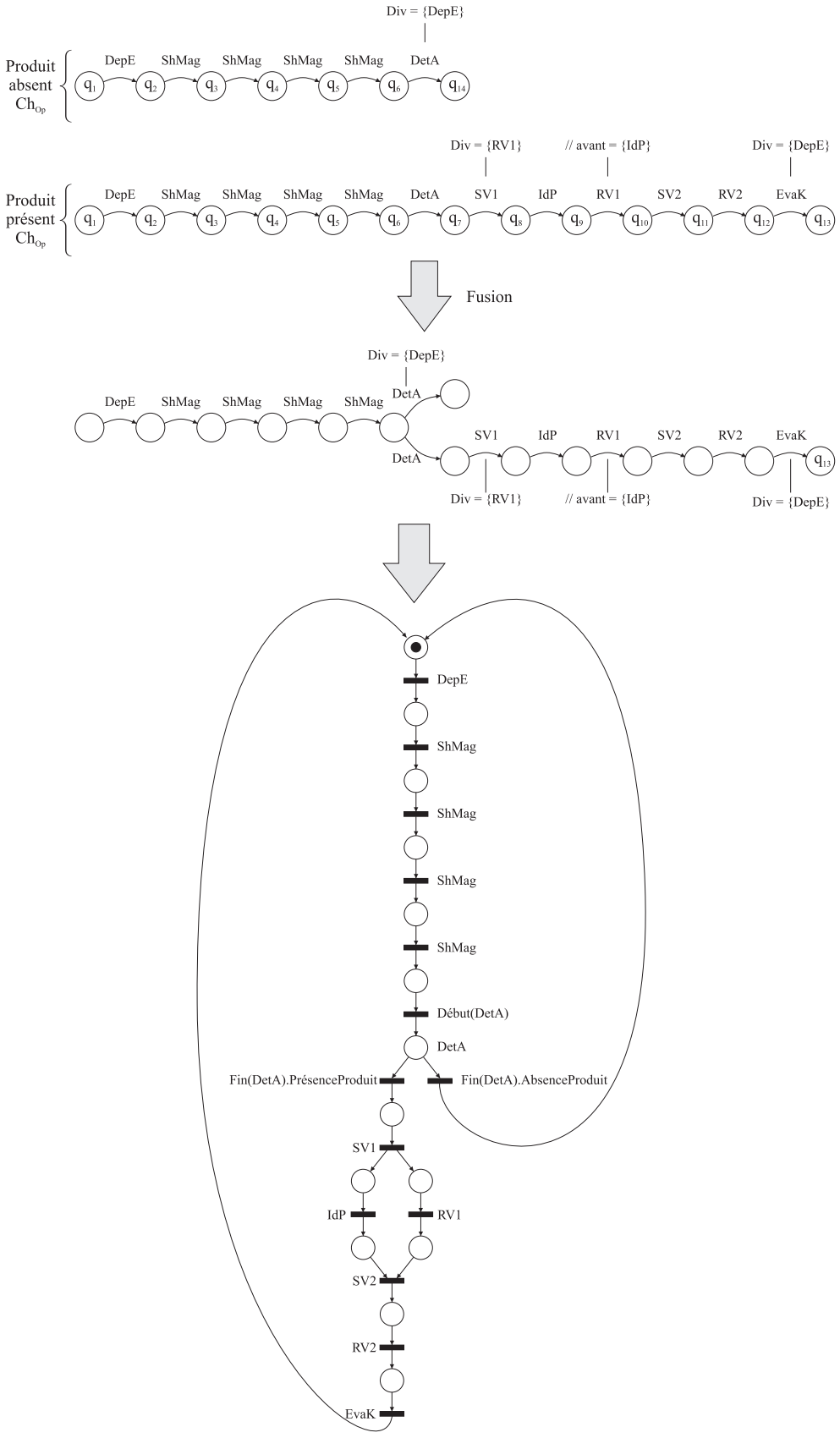


FIG. D.4 – Loi de commande avec parallélisme d'exécution entre les opérations liées à un même produit.

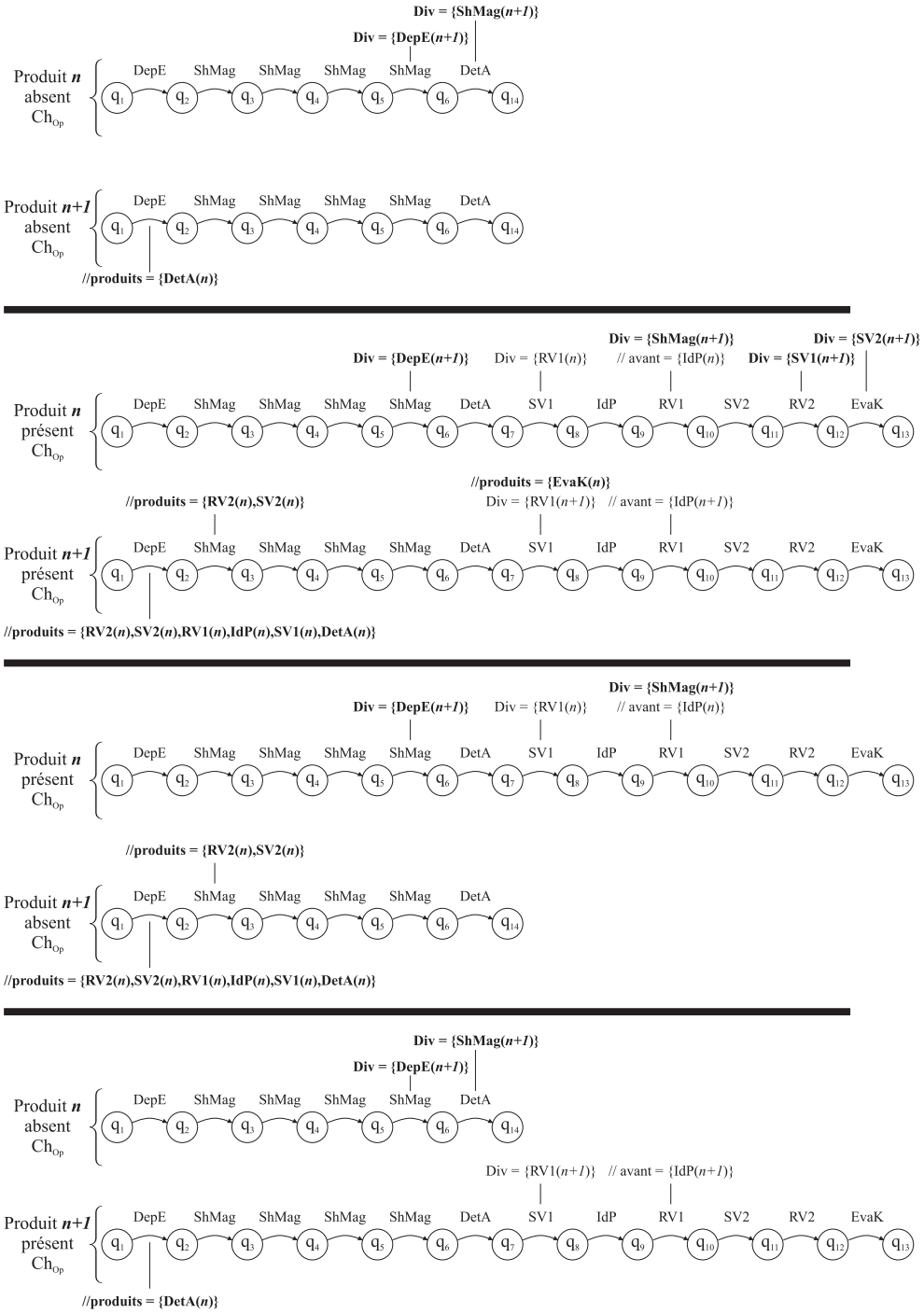


FIG. D.5 – Application quatre fois du mécanisme de parallélisation.

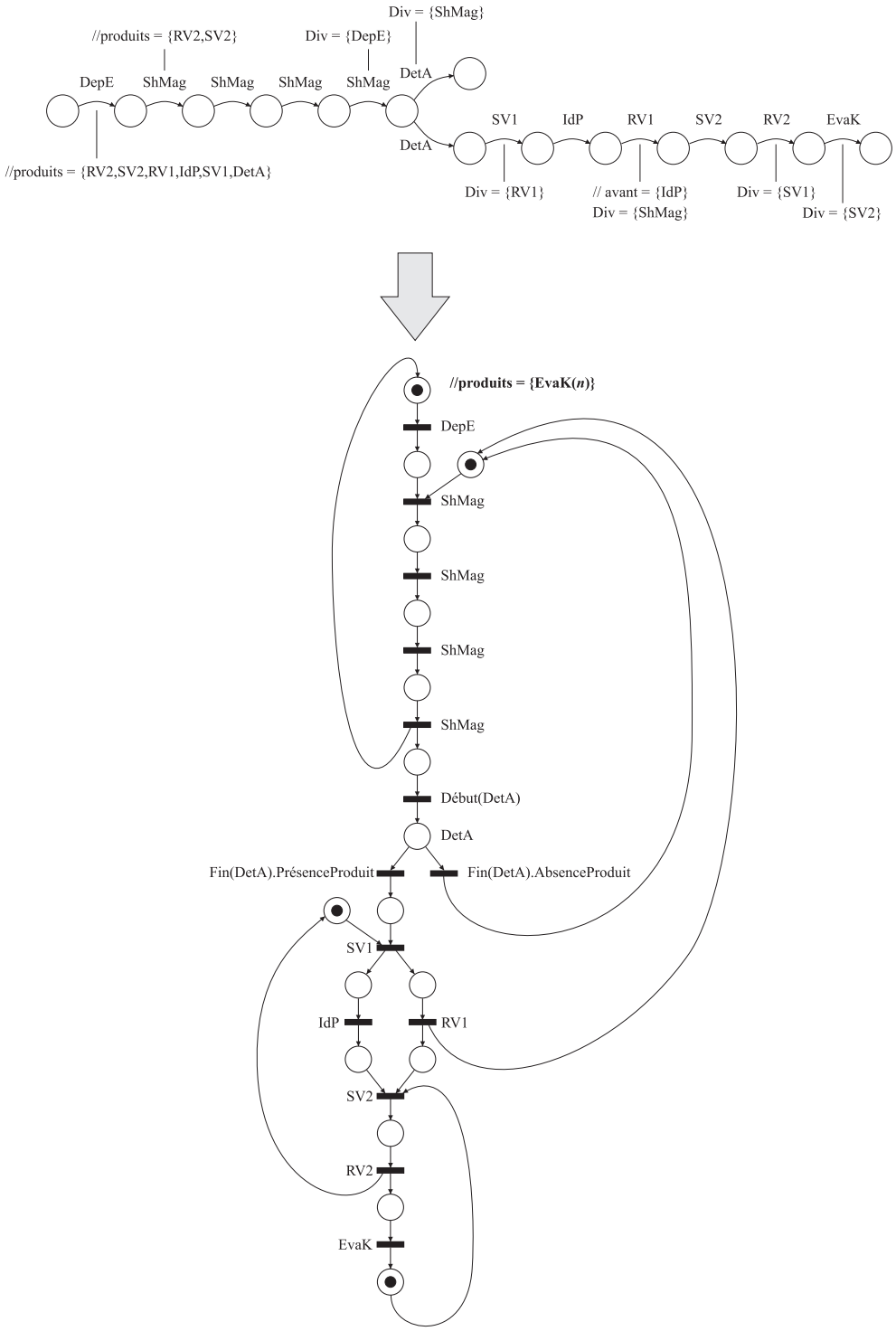


FIG. D.6 – Loi de commande avec l'ensemble des parallélismes d'exécution.

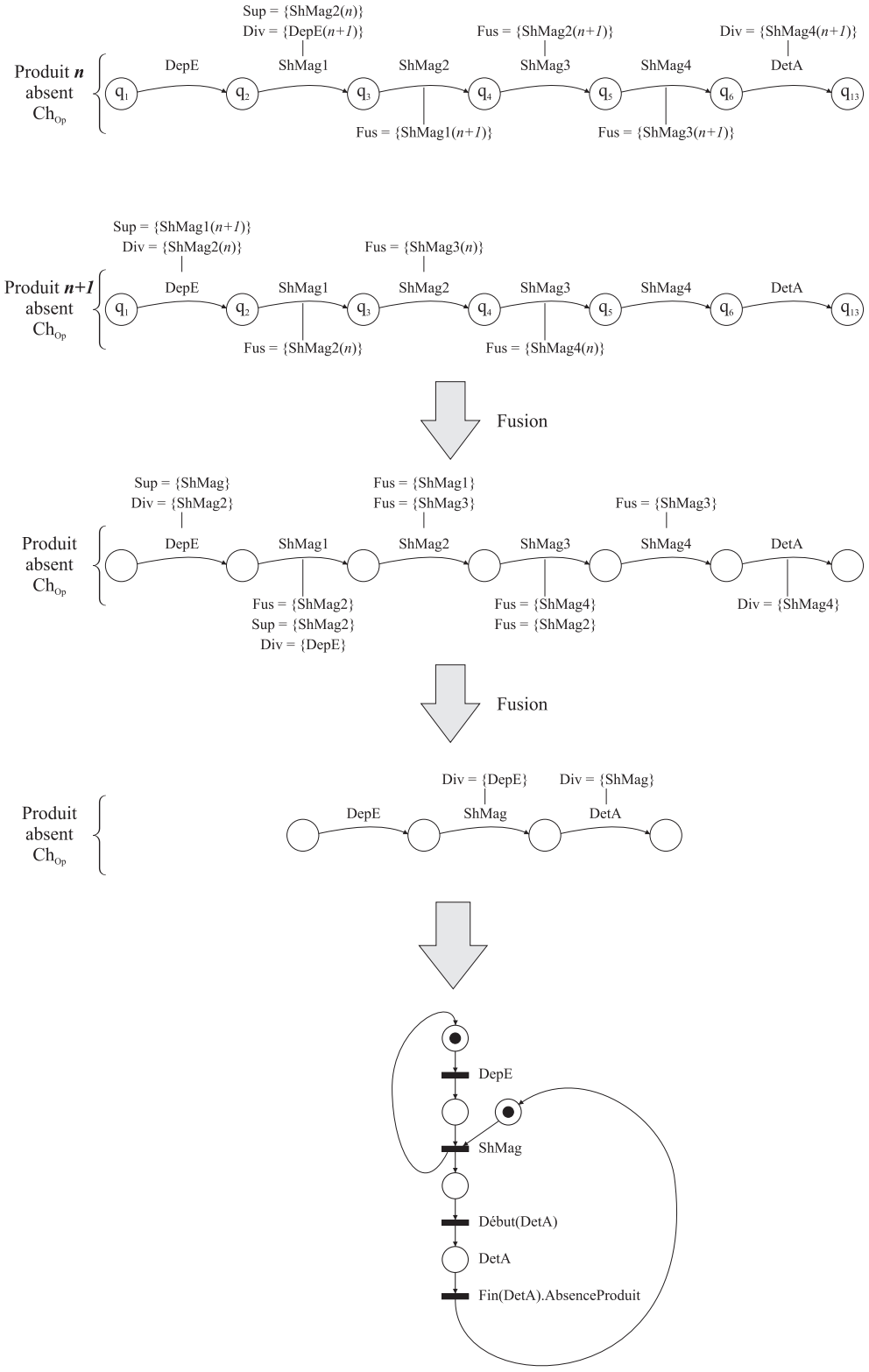


FIG. D.7 – Exemple d'application du mécanisme d'insertion sur un exemple simplifié avec uniquement les produits absents.

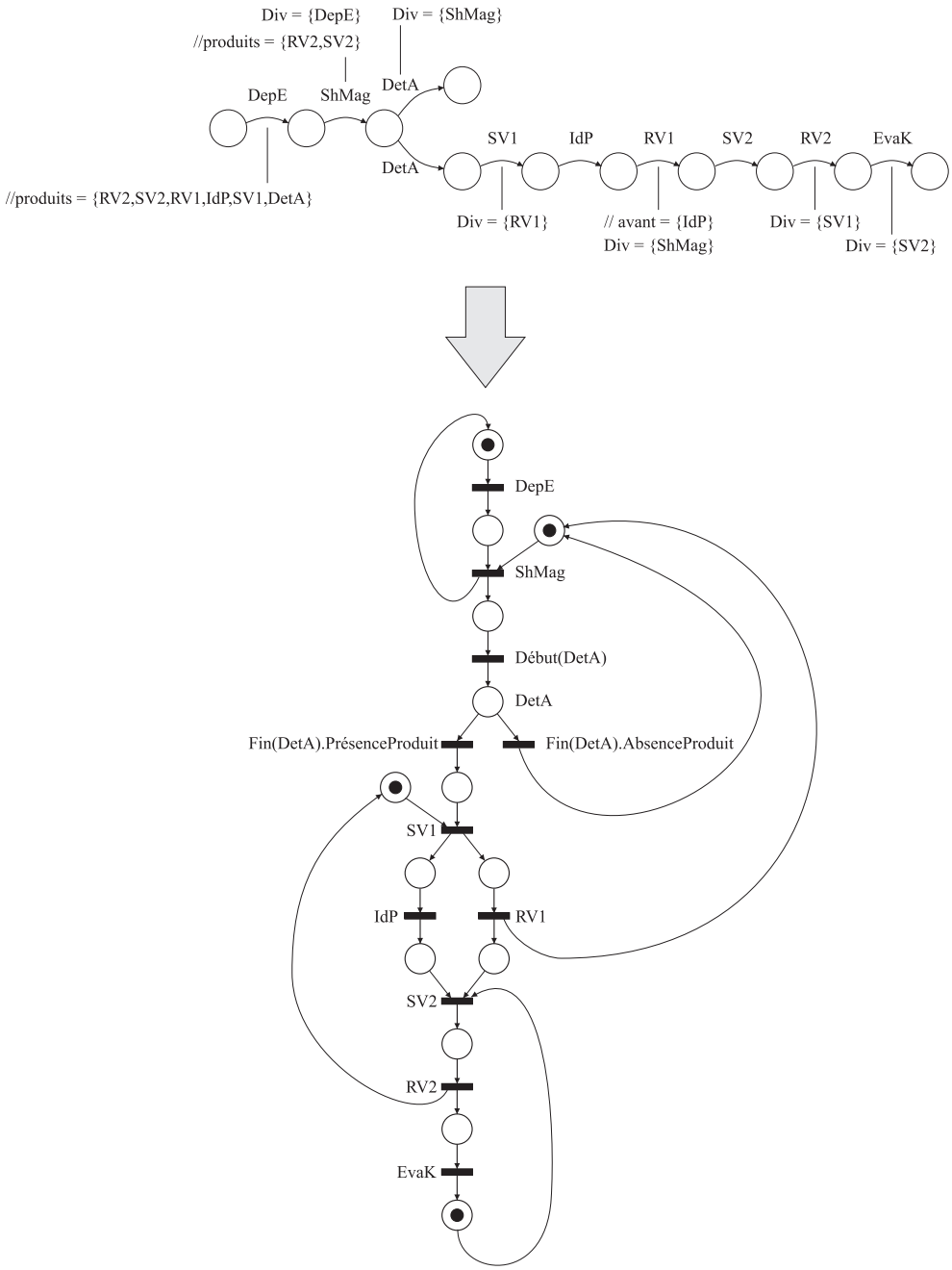


FIG. D.8 – Loi de commande résultat de l'algorithme complet de synthèse répondant à la demande du § 2 page 185.

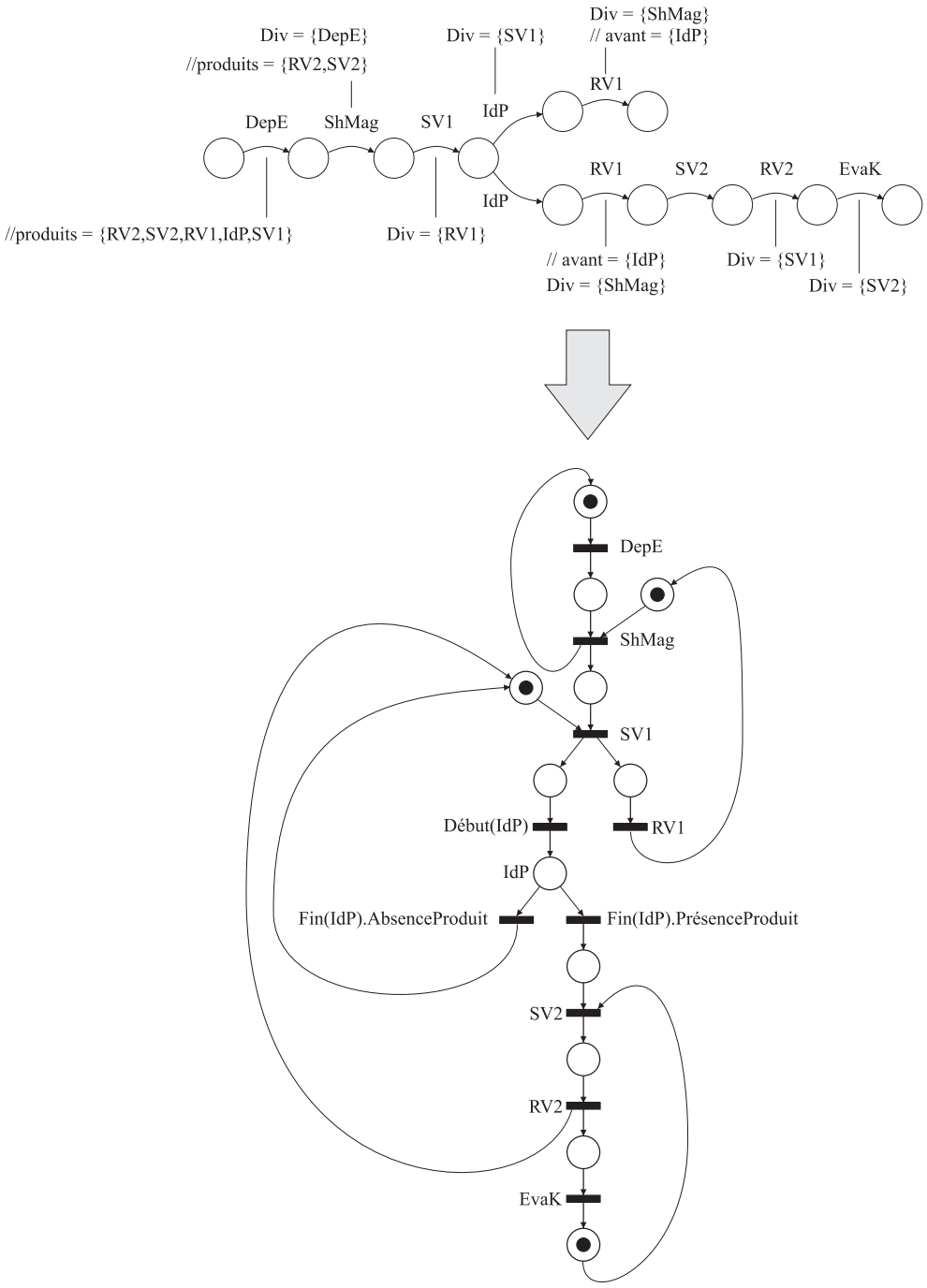


FIG. D.9 – Loi de commande synthétisée sans utilisation de l'opération de détection offerte par le capteur en A.

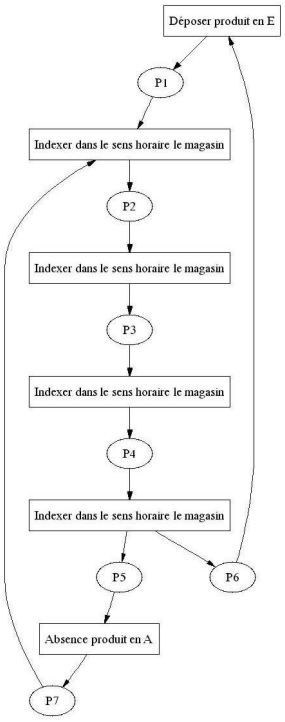


FIG. D.10 – Pour les produits absents uniquement, réseau de Petri généré automatiquement par l'algorithme mis en œuvre en Java.

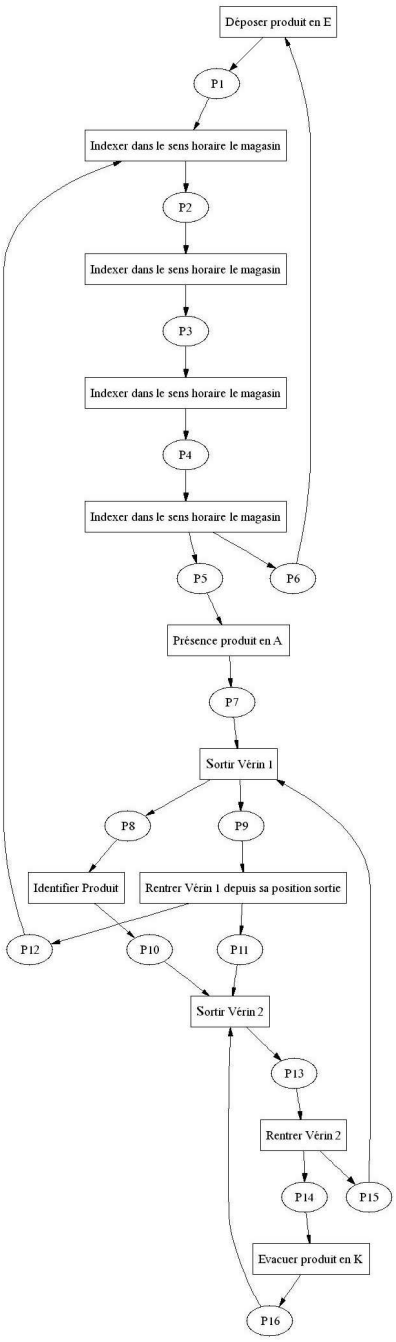


FIG. D.11 – Pour les produits présents uniquement, réseau de Petri généré automatiquement par l'algorithme mis en œuvre en Java.

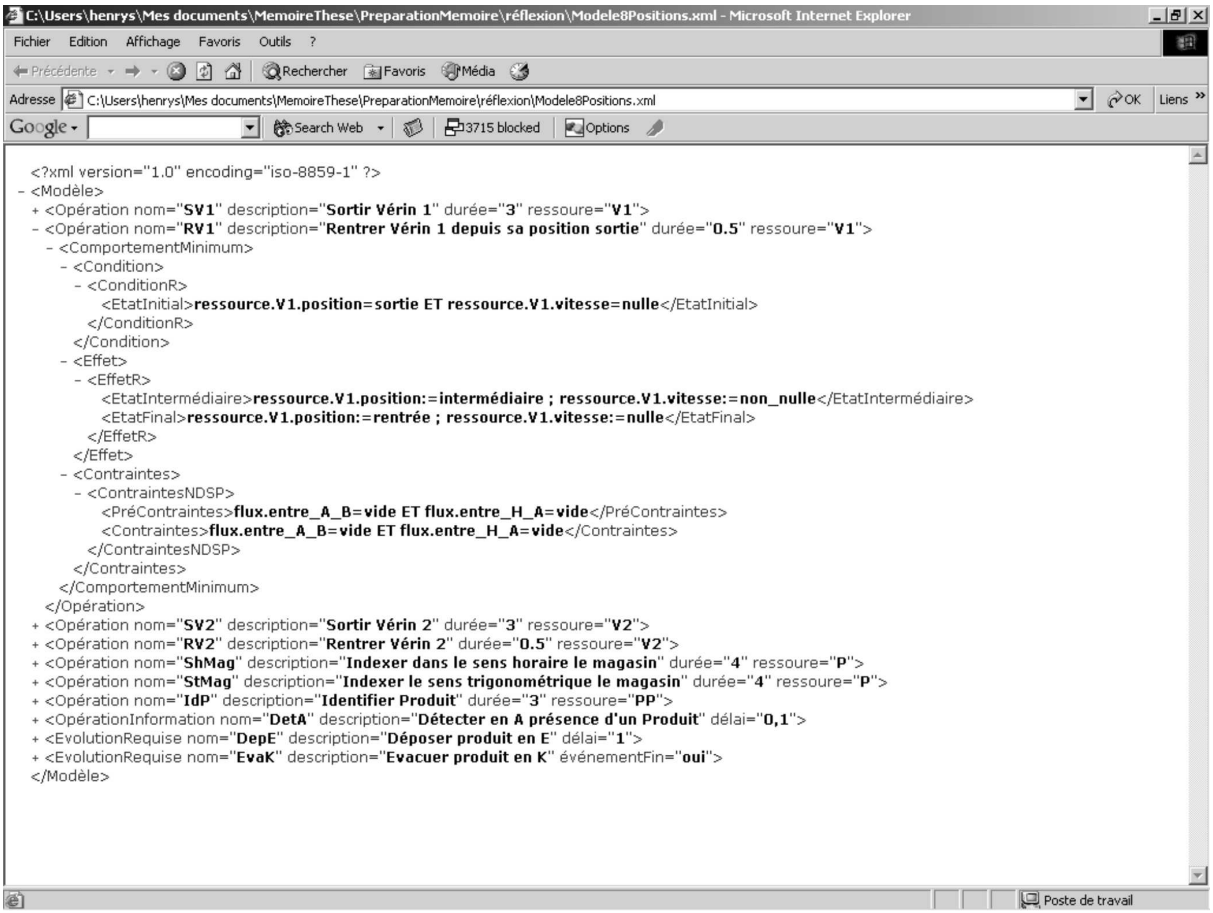


FIG. D.12 – Copie d'écran du fichier XML décrivant le modèle du système d'approvisionnement de SAPHIR.

Résumé : le travail présenté dans ce mémoire de thèse apporte sa contribution à la synthèse de lois de commande en contexte incertain des systèmes automatisés de production. L'incertain est ici caractérisé d'une part par les variations imprévues des demandes client, mais également par les aléas de fonctionnement déclarés au niveau de la partie opérative. L'approche, localisée au niveau 1 du CIM, et en particulier au sein des modules de coordination des chaînes fonctionnelles, s'intègre au sein d'un système plus général de Supervision, Surveillance et Commande. L'approche se distingue en considérant la globalité du processus qui mène à la reconfiguration des lois de commande. En effet, elle propose non seulement une méthode de modélisation de la partie opérative utilisant un formalisme particulièrement adapté à la complexité des procédés considérés mais aussi une technique de synthèse de lois de commande basée sur un mécanisme de recherche de chemins dans un graphe. La modélisation proposée, proche de celle utilisée en planification automatique, est basée sur un ensemble d'opérations qui décrivent la dynamique des chaînes fonctionnelles et leurs effets sur le flux de produits tout en prenant en considération les contraintes sécuritaires et environnementales associées. Toute l'originalité du mécanisme de synthèse proposé réside dans le compromis réalisé entre la complexité du graphe manipulé et les performances de la solution obtenue.

Un exemple d'application basé sur un processus manufacturier réel, la plate-forme SAPHIR du Laboratoire d'Automatique de Grenoble, et sur l'atelier logiciel développé sur la base du mécanisme de synthèse proposé illustre les apports de notre approche.

Mots clès : Supervision, Reconfiguration, Commande des systèmes à événements discrets, Synthèse, Optimisation, Réseaux de Petri, Contexte incertain, Aide à la décision.

Abstract: The presented work deals with the synthesis of control laws in uncertain context of the automated production systems. The uncertain one is characterized here on the one hand by the unpredicted variations of customer requests, but also by the failures of resources. The approach, localised into level 1 of the CIM, and in particular within the coordination modules of the functional chains, is integrated in a system of Supervision, Monitoring and Control. The approach considers globality from the process which leads to the reconfiguration of the control laws. Indeed, it proposes not only one method of modelling of an operative part using a formalism adapted to process complexity but also a technique of control law synthesis based on the research of paths in a graph. The suggested modelling, near that used in the automatic planning field, is based on a set of operations which describe the dynamics of functional chains and theirs effects on product flow while taking into account the security and environmental constraints. The originality of the synthesis algorithm is the compromise between the complexity of the used graph and the solution performances.

An application example based on a real manufacturing process, the platform SAPHIR of the Laboratoire d'Automatique de Grenoble, and on the software workshop developed on the basis of proposed synthesis algorithm illustrates the contributions of our approach.

Key Words: Supervision, Reconfiguration, Control of discret event systems, Synthesis, Optimization, Petri nets, Uncertain context, Decision-making aid.