Nancy-Université
Université Henri Poincaré

CRAN

Département de formation doctorale en automatique
UFR STMIA

École doctorale IAEM Lorraine

Proposition d'un environnement de modélisation et de test d'architectures de pilotage par le produit de systèmes de production

THÈSE

présentée et soutenue publiquement le 13 juillet 2007

pour l'obtention du titre de

Docteur de l'université Henri Poincaré

en Automatique, Génie Informatique

par

Rémi Pannequin

Composition du jury

Rapporteurs :	Pr. Pierre Castagna	Université de Nantes
	Pr. Damien Trentesaux	Université de Valenciennes et du Hainaut-Cambrésis
	Dr. Paul Valckenaers	Université Catholique de Louvain, Belgique
Examinateurs :	Pr. Isabel Demongodin	Université Paul Cézanne de Marseille
	Pr. Thierry Divoux	Université Henri Poincaré
	Pr. Gérard Morel	Université Henri Poincaré (directeur de thèse)
	Pr. André Thomas	Université Henri Poincaré (co-directeur de thèse)

Mis en page avec la classe thloria.

Remerciements

Les travaux présentés dans ce mémoire ont été réalisé au Centre de Recherche en Automatique de Nancy (CRAN - UMR 7039), au sein du groupe thématique Systèmes de Production Ambiants (SYMPA), et dans l'équipe-projet Systèmes Contrôlés par le Produit. Ils ont été financés grâce à une allocation de recherche et un contrat de moniteur par le ministère de l'Enseignement Supérieur, de la Recherche et de la Technologie. De plus, le financement de la dernière année de la thèse a été assuré par l'équipe de recherche technologique Tracilog du CRAN.

Je suis extrêmement reconnaissants à Monsieur Paul Valckenears, chercheur à l'université Catholique de Louvain, à Monsieur Pierre Castagna, professeur à l'université de Nantes, et à Monsieur Damien Trentesaux, professeur à l'université de Valenciennes et du Hainaut-Cambrésis d'avoir accepté la charge d'évaluer ce travail en qualité de rapporteurs.

Je remercie également Madame Isabelle Demongodin, professeur à l'université Paul Cézanne de Marseille et Monsieur Thierry Divoux, professeur à l'université Henri Poincaré pour l'intérêt qu'ils ont porté à ces travaux en tant qu'examinateurs.

Que Messieurs Gérard Morel et André Thomas, professeurs à l'université Henri Poincaré, trouvent ici l'expression de ma gratitude pour leur encadrement en tant que directeur et co-directeur de thèse. Sous leur encadrement, ces quelques années de *formation par la recherche* m'ont beaucoup apporté, tant au plan professionnel que personnel. Je les remercie aussi pour leur patience face à mes défauts.

Je tiens enfin à remercier l'ensemble des membres du laboratoire, dont l'attitude chaleureuse à rendu mon séjour au CRAN bien agréable. Si j'ai jamais eu une idée, c'est grâce aux dicussions informelles dans notre *brain-storming room* favorite. Je remercie aussi mes parents pour leur support de tous les instants, en particulier pour m'avoir assisté, quant à l'orthographe et l'expression, dans la rédaction de ce mémoire.

Résumé

Les exigences conjointes de productivité (cohérence et optimalité dans la chaîne logistique) et d'agilité (traçabilité, personnalisation en masse, tolérance aux fautes) amènent à considérer un pilotage de la production combinant les approches centralisées et distribuées de la prise de décision. Toutefois, à cause de leurs caractéristiques opposées, globale et prédictive dans un cas, locale et réactive dans l'autre, trouver un équilibre entre les fonctions centralisées et distribuées n'est pas évident.

Les technologies infotroniques, qui permettent de faire porter numériquement des données par le produit afin de lui conférer un rôle actif dans la boucle cybernétique, conduisent à remettre en question l'organisation conventionnelle des systèmes de pilotage, comme le prône l'initiative internationale IMS (Intelligent Manufacturing Systems). Il existe actuellement un large consensus quant à l'intérêt de ce nouveau paradigme de modélisation HMS (Holonic Manufacturing Systems), que nous déclinons en *pilotage par le produit*, quant à son apport potentiel pour la prise de décision tant centralisée que distribuée.

Toutefois, peu de travaux portent sur la preuve de ce concept quant à son efficience pour combiner de façon plus flexible des modes de pilotage centralisés avec des modes de pilotage distribués en tenant compte des capacités du produit pour jouer un rôle actif de synchronisation des échanges entre différent système d'entreprise de niveau *business* (ERP : Entreprise Resource Planing) et *process* (MES : Manufacturing Execution System).

Notre contribution porte sur un outil de modélisation et de simulation de systèmes de pilotage contrôlés par le produit afin d'évaluer différentes topologies d'organisation combinant décisions centralisées et/ou distribuées en comparant certains critères de productivité.

Notre contribution scientifique porte principalement sur la définition et le développement d'un environnement d'évaluation dans l'objectif d'un déploiement de systèmes de pilotage par le produit sur des cas industriels, notamment dans le contexte de l'ERT Tracilog (TRACabilité, Identification et contrôle par le produit pour les chaînes LOGistiques des filières bois et fibres).

La première partie de la thèse présente en deux chapitres la définition, le développement et la validation de cet environnement d'évaluation composé de :

- un logiciel d'émulation supportant la création et l'exécution de modèles d'atelier virtuels, d'échelle réaliste.
- un ensemble de composants logiciels orientés-agent facilitant le développement de systèmes contrôlés par le produit.

Ces deux éléments, utilisés conjointement ou séparément constituent le prototype d'un banc d'essais, permettant l'analyse des performances d'un système de pilotage par le produit et sa comparaison avec des approches classiques.

La seconde partie présente en deux chapitres l'application du pilotage par le produit à partir d'une série d'expériences sur un cas industriel ainsi que sur la plateforme d'expérimentation de l'ERT. Ces expériences permettent d'éprouver et de valider la faisabilité du concept de pilotage par le produit en terme d'impact décisionnel (réordonancement local, cohérence entre l'état physique et informationnel du produit, ...) ainsi qu'en terme de contraintes techniques (fiabilité de l'identification produits, défauts du réseau de communication, ...).

Table des matières

Table des figures

Introduction

Nos travaux adressent la problématique de pilotage des systèmes d'exécution de la production (*Manufacturing Execution Systems*) afin d'évaluer la pertinence offerte par les technologies infotroniques pour rendre *actif* le produit lui-même dans le processus de décision.

Les exigences de productivité sont renforcées par une concurrence exacerbée, entraînée par la mondialisation des échanges. Cette concurrence conduit aussi les entreprises à développer de nouveaux services comme la personnalisation ou la traçabilité des produits, qui augmentent leur valeur ajoutée. Pour répondre à ces deux exigences, le pilotage de la production doit combiner des qualités d'*efficience*, permettant de maintenir une haute productivité, et d'*agilité*, permettant d'appréhender la forte variété des produits et l'incertitude de l'environnement [Bousbia, 2006].

D'une part, l'efficience se doit d'être globale, l'objet considéré n'étant plus le site de production et l'entreprise, mais la chaîne logistique et le groupe multinational. En effet, les décisions prises lors de l'exécution de la production peuvent influer sur la chaîne logistique dans son ensemble. De ce point de vue, le problème est donc le maintien de la cohérence des informations et décisions de pilotage dans la chaîne logistique.

D'autre part, l'agilité est requise pour satisfaire aux besoins de personnalisation (fabrication à la commande, *mass-customisation*) et de traçabilité des produits, et aussi pour faire face à la versatilité des conditions de fonctionnement. Cette versatilité est causée, à moyen terme par le raccourcissement du cycle de vie des produits (reconfiguration), et à court terme par les perturbations opérationnelles (pannes, ruptures d'approvisionnement, absentéisme) ou commerciales (commande urgente, modification d'une commande, etc).

L'ensemble de ces exigences parfois contradictoires conduit à l'apparition d'une complexité difficile à gérer. En effet, la recherche de l'efficience et de la cohérence de la chaîne logistique conduit à centraliser la prise de décision dans des progiciels de gestion intégrés. Les décisions sont donc prises par des systèmes globaux difficilement maîtrisables (car complexes et opérant à distance), confinant les hommes sur le terrain à un rôle d'exécutants. À l'inverse, le caractère perturbé et incertain de l'environnement de production, ainsi que la complexité opérationnelle causée par la forte variété des produits, favorise une prise de décision distribuée au plus près des flux de matière, locale et réactive. Industriellement, on peut donc observer une relative incompatibilité entre ces deux pratiques. Les conséquences en sont soit la perte d'autonomie et d'initiative de la part des opérateurs (ce qui peut être vu comme le gaspillage d'un savoir-faire), soit la perte de visibilité de l'atelier par les système d'information centraux. Ce conflit s'accompagne d'une multiplication des données relatives au produit, éparpillées dans autant de bases (s'échelonnant du système de gestion de bases de données relationnel à la feuille Excel) qu'il y a de points de vue sur la production.

Une solution pour assurer la cohérence des informations tout en offrant un équilibre entres les approches centralisées et distribuées du pilotage est de faire porter par le produit lui-même les informations qui lui sont relatives. C'est le postulat de l'équipe *système contrôlé par le produit* (SCP) du CRAN, qui choisit de faire du produit le centre du système d'information de l'entreprise. En effet, le produit est l'objet « pivot » de l'entreprise, commun à une majorité de ses départements (conception, production, marketing, vente, service après-vente, etc...).

Ce choix est aussi motivé par l'apparition de technologies infotroniques, permettant d'adjoindre au produit matériel des capacités informatiques plus ou moins importantes. Actuellement, ces technologies se concrétisent principalement par les étiquettes électroniques et sans contacts (*tags RFID – Radio Frequency Identification*). Mais ce secteur technologique se développe rapidement, par exemple dans le domaines des réseaux de capteurs.

Ces nouvelles capacités de communication, de stockage et de traitement de l'information allouées au produit peuvent être exploitées pour supporter différentes phases de son cycle de vie (production, maintenance, recyclage) [Bajic et Chaxel, 2002], en particulier en ce qui concerne les problématiques de la personnalisation et de la traçabilité.

De plus, la possibilité d'observation automatique du flux de produits a un impact sur la prise de décision en modifiant la boucle cybernétique (figure 1) [Mcfarlane *et al.*, 2003]. Mcfarlane montre ainsi que l'identification automatique constitue un apport à la fois :

– pour les systèmes classiques de gestion de production, comme le système MRP2 ;
– pour les nouvelles approches, comme les systèmes holoniques.

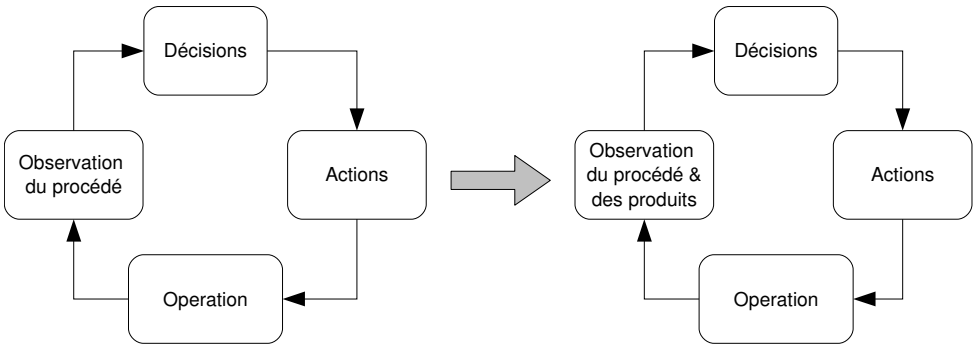


Fig. 1 – Introduction de capacité de perception du produit dans le contrôle de la production.

Intégrée à un système comme MRP2, l'identification automatique permet de résoudre le problème de la synchronisation entre flux de matière et flux d'information, reconnu majeur [Plossl, 1985]. En fournissant des données fiables et précises sur la situation d'état du système de production, l'identification automatique permet de mettre en œuvre des modèles avancés de recherche opérationnelle, répondant aux besoins de cohérence et d'efficience dans la chaîne logistique.

Par ailleurs, l'utilisation de l'infotronique dans un système holonique ou multi-agents permet de réaliser l'association entre la matière et l'information. Grâce à cette association, la décision peut être distribuée au plus près du flux de matière, en utilisant des méthodes d'intelligence artificielle, répondant aux besoins de gestion de la variété des produits et d'adaptation à la variabilité des conditions de fonctionnement.

Ces deux manières d'impliquer le produit aux processus de décision nous pousse à nous

interroger sur l'opportunité d'allouer au produit un rôle actif dans le système de décision. Cela amène à remettre en question la vision intégrée et hiérarchique du contrôle de l'entreprise, pour une approche axée sur l'interopérabilité et l'intelligence introduisant plus d'hétérarchie dans le pilotage de la production (figure 2) [Morel *et al.*, 2007].

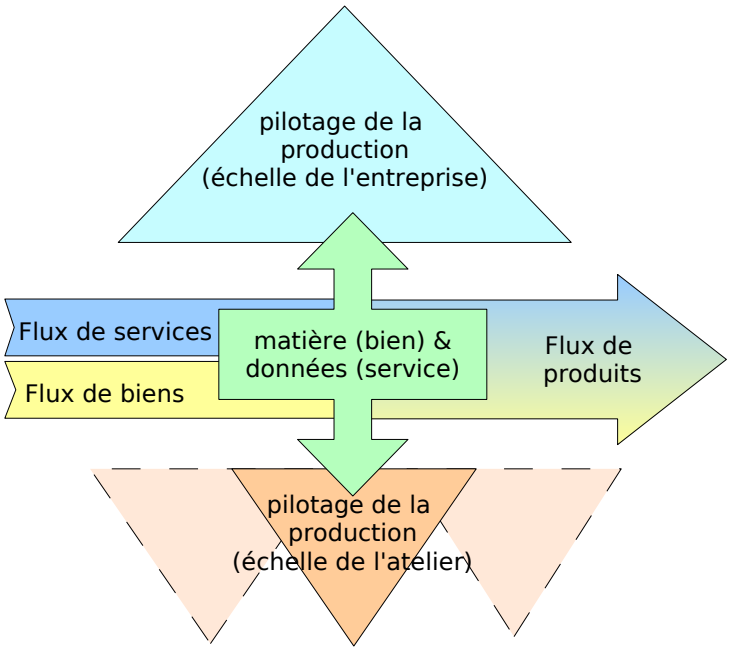


FIG. 2 – Architecture de décision centrée sur le produit

Cette remise en cause de l'organisation conventionnelle du pilotage, prônée par l'initiative internationale IMS (*Intelligent Manufacturing Systems*) repose sur le paradigme de modélisation HMS (*Holonic Manufacturing Systems*) [Valckenaers, 2001] qui est supporté par une communauté de recherche en plein essor.

Il existe donc un large consensus sur l'intérêt du concept de pilotage par le produit. Toutefois, il n'y a encore que peu d'outils et de travaux permettant de faire la preuve de ce concept. En particulier, peu de travaux portent sur la pertinence de combiner les approches centralisées et distribuées du pilotage de la production en tenant compte des capacités du produit à servir de vecteur d'échange entre les différents systèmes d'entreprise de niveau *business* (ERP : Entreprise Resource Planing) et *process* (MES).

Pour répondre à ce problème, nous proposons dans cette thèse un outil de modélisation d'une architecture décisionnelle, permettant d'évaluer des scénarios d'allocation des décisions (et des informations associées) à divers centres de décision. Cet outillage nous permet ensuite d'illustrer l'impact de ces scénarios sur les caractéristiques de traçabilité, d'adaptativité, et de cohérence de la décision dans la chaîne logistique.

Notre contribution se fait donc à deux niveaux :

- d'une part en illustrant le principe de contrôle par le produit, grâce à une mise en œuvre dans une situation concrète,
- d'autre part, en proposant des outils supportant le développement et la validation de systèmes contrôlés par le produit.

Le premier chapitre de la thèse s'appuie sur un état de l'art des solutions industrielles et académiques pour faire apparaître et expliciter les caractéristiques des deux approches, centralisée et distribuée, de la prise de décision.

Le mode centralisé est caractérisé par la décomposition d'un problème de pilotage global en un ensemble de sous-problèmes. La structure est donc hiérarchique, et le modèle du système à piloter doit être défini explicitement. L'approche est synchronique (ponctuellement dans le temps), la décision se faisant périodiquement dans la limite d'un horizon.

Le mode distribué est caractérisé par l'interaction entre des centres de décision locaux, le comportement global du système est émergent, sans nécessité d'une modélisation explicite. La structure est donc hétérarchique, et auto-organisée. L'approche est diachronique (à travers le temps), les centres locaux étant en interaction permanente avec leur environnement.

En conclusion de ce chapitre qui pose le problème de l'interaction entre décisions centralisées et décisions distribuées, nous postulons que ces deux modes, malgré leur relatif antagonisme, doivent coopérer en fonction de leurs apports respectifs complémentaires. Notre outil d'ingénierie doit donc permettre d'évaluer des solutions d'architecture mixant aussi bien des approches décisionnelles centralisées que des approches distribuées, y compris dans le produit lui-même.

Le deuxième chapitre définit l'architecture de l'outil d'ingénierie proposé. Cet outil a pour fonction de permettre la modélisation et l'évaluation par émulation de diverses architectures décisionnelles, en particulier le contrôle par le produit.

Pour atteindre cet objectif, les méthodes classiques de modélisation et de simulation doivent être ré-examinées. Ainsi, il faut assurer la distinction entre les flux de différents types (matières, données, événements), permettre une représentation suffisamment fine des équipements agissant sur le produit, et représenter les événements associés aux nouvelles capacités d'observation du produit.

Pour répondre à ces besoins, nous définirons une méthodologie de modélisation basée sur des primitives d'inspiration systémique permettant l'émulation du système opérant. Nous présenterons ensuite le développement d'un prototype, basé sur un logiciel de simulation, supportant cette méthodologie. Enfin, nous validerons le fonctionnement correct de ce prototype, et nous vérifierons l'expressivité des primitives de modélisation, en l'appliquant à des situations réalistes.

Le troisième chapitre décrit le système multi-agents que nous proposons pour mettre en œuvre un contrôle par le produit. La technologie multi-agents permet l'évolution en parallèle de plusieurs entités décisionnelles, et facilite d'autre part l'étude des interactions entre centres de décision.

Nous utiliserons les outils classiques d'ingénierie multi-agents, et en particulier les normes FIPA (*Foundation for Intelligent Physical Agents*), pour développer un ensemble de composants logiciels orientés agents, pouvant être utilisés pour mettre en place un système de contrôle. Ces composants comportent divers modèles d'agents (ressources, produits, centres de décision, ...) ainsi que les protocoles leur permettant de communiquer. Cet environnement de développement et d'étude facilite donc la représentation les produits et leurs interactions avec leur environnement décisionel et opérant.

Ce développement est validé techniquement par l'application du système multi-agents à quelques cas simples.

Le quatrième chapitre met en œuvre les outils et méthodes présentés précédemment dans une série d'expériences sur un cas industriel. Ces expériences ont pour but de montrer la faisabilité industrielle des procédures de décision de pilotage par le produit, en les comparant avec un

pilotage centralisé. Les erreurs d'implémentation du pilotage par le produit pouvant biaiser les résultats, notre modèle est validée par un « étalonnage » préliminaire.

Un plan d'expériences est mené en faisant varier divers facteurs perturbant du process (variabilité des temps de cycle et de réglage, probabilité de pannes, etc) et du business (variabilité des ordres de fabrication, présence de *rush orders*) en mesurant des indicateurs de productivité du système opérant.

Enfin, dans le cinquième chapitre, nous présenterons l'application de notre environnement de modélisation et de test dans le cadre des activités de transfert industriel de l'équipe de recherche technologique TRACILog. Nous analyserons d'abord les apports et les limites de l'environnement d'évaluation par rapport au problématiques industrielles posées par un projet de mise en oeuvre d'un pilotage par le produit, en l'illustrant par le cas d'un fabricant de meubles.

Ensuite, nous présenterons la plateforme d'essais de l'ERT, et la manière dont le pilotage par le produit pourra y être appliqué.

Nous montrerons que ces deux outils d'évaluation permettent d'offrir à l'indutries des services de benchmarking dans les phases de définition et de développement du projet de mise en place d'un pilotage par le produit (en utilisant l'environnement d'évaluation), ainsi que dans les phases de déploiement (en utilisant la plateforme d'essais).

Chapitre 1

Pilotage par le produit des systèmes de production : hybridation entre approche centralisée et distribuée de la prise de décision

Ce chapitre a pour but de construire la problématique de la thèse, en partant du problème général de l'interaction entre décisions centralisées et décisions distribuées, pour ensuite réduire le champ considéré au concept de pilotage par le produit, et finalement arriver à la formulation du problème précis auquel nous nous intéressons : faire la preuve expérimentale et valider la pertinence de ce concept. Il présente de plus l'état de l'art sur les systèmes de pilotage distribués, en particulier les systèmes contrôlés par le produit.
La première section traite du problème général de l'antagonisme entre les approches centralisée et distribuée du pilotage des systèmes de production. Nous définirons les différents critères caractérisant chaque approche, et nous énoncerons leurs apports et limites respectifs, à l'aide d'exemples issus de la pratique industrielle et académique.
La deuxième section montre les domaines d'application de ces deux types d'approche, ainsi que le problème posé par leur interaction. Nous analyserons les solutions proposées dans la littérature pour nous positionner par rapport à elles.
Enfin, la troisième section propose une solution potentielle à ce problème : l'approche de pilotage par le produit. Après avoir présenté l'état de l'art sur le pilotage par le produit, nous énoncerons le concept qui pourrait être utilisé, et nous montrerons enfin le besoin de faire la preuve de ce concept, qui constitue le verrou *que nous cherchons à résoudre dans cette thèse.*

1.1 Approche centralisée et distribuée du pilotage de la production

L'opposition entre une vision centralisée ou distribuée de la prise de décision est un concept largement utilisé, mais assez vaguement défini. L'antagonisme centralisé/distribué peut être trouvé dans de nombreuses disciplines, comme l'économie, la politique ou la sociologie. Dans le domaine de l'organisation des entreprises, plus proche du pilotage de la production, cet antagonisme s'incarne dans l'opposition entre le Taylorisme, fondé sur des structures hiérarchiques

rigides et un découpage du travail en tâches élémentaires, et les méthodes d'origine japonaise (issues des travaux de Ohno chez Toyota), fondées entre autre sur l'implication et la polyvalence des ouvriers.

Cette opposition s'exprime actuellement de manière renouvelée dans le domaine du pilotage de la production. En effet, l'accroissement des besoins industriels d'agilité et de traçabilité, ainsi que l'apparition de nouvelles possibilités techniques ou fondamentales, conduisent à la proposition de systèmes de pilotage *distribués*, qui s'opposent à des méthodes de gestion plus classiques, *centralisées*. Il est nécessaire de définir plus précisément ces termes, qui font en fait référence à tout un ensemble de critères.

1.1.1 Critères d'analyse

Le choix des critères utilisés pour distinguer les systèmes de décision centralisés et distribués s'appuie sur une analyse du processus de prise de décision. D'après H.A. Simon [Simon, 1997], le processus de décision peut être décomposé en quatre phases :

- *l'intelligence* : perception des problèmes,
- la *conception* : modélisation des solutions potentielles,
- la *sélection* : choix d'une solution
- *l'implémentation* : mise en œuvre et évaluation de la solution.

Ce processus de décision est exécuté par un *centre de décision*. Celui-ci reçoit donc des informations, provenant soit de son champ de perception, soit d'un échange d'informations avec d'autres centres. Ces informations sont intégrées à la représentation de l'état de l'environnement (*world model*) afin de construire des solutions potentielles. Celle qui correspond le mieux à un critère de *jugement de valeur* est ensuite sélectionnée [Albus et Barbera, 2005]. De plus, ce processus s'exécute avec une certaine dynamique.

Nous retenons donc les critères d'analyse suivants :

- la structure du système de décision, qui dépend de la représentation par les centres de décision de leur environnement (modèle du monde et champ de perception)
- les mécanismes de coordination utilisés,
- la formulation des objectifs, qui permet de doter chaque centre d'un critère de jugement de valeur,
- la dynamique des centres de décision.

1.1.2 Topologie des systèmes de décision

La structure, ou topologie, des centres de décision est le critère le plus couramment utilisé pour distinguer un système centralisé d'un système distribué. Ainsi, on parle de forme *pyramidale* ou de forme *plate* [Bongaerts *et al.*, 2000]. La différence entre ces deux structures relève de l'existence d'une échelle, permettant de distinguer des centres de décision « haut » ou « bas ». Cette échelle est fondée sur la notion de *limite cognitive*.

La limite cognitive [Simon, 1996] borne la quantité d'information pouvant être gérée par un individu ou par un système artificiel. Elle va ainsi restreindre le rapport entre la globalité des perceptions et leurs précisions (figure 1.1). On aura donc d'une part des centres de décision locaux, traitant des informations détaillées mais relatives à une petite zone (par exemple une cellule de fabrication), et d'autre part des centres globaux, ayant un point de vue bien plus large (un site, un ensemble d'usine) mais ne connaissant que des informations agrégées.

Du point de vue temporel, on observe le même phénomène : l'horizon temporel, c'est à dire la durée dans le futur qui est prise en compte, limitera la *maille de temps* avec laquelle on

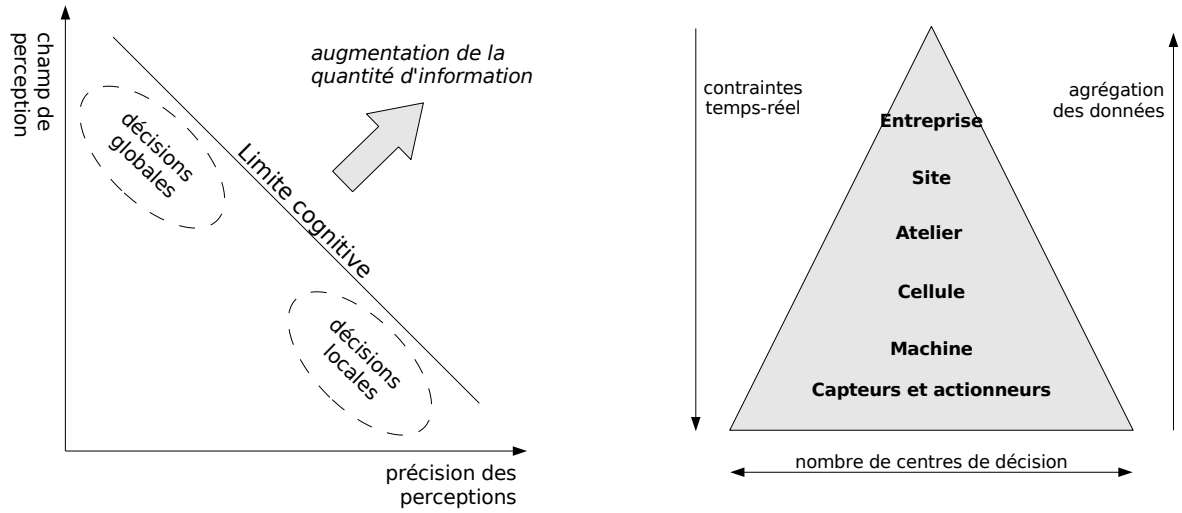


FIG. 1.1 – La limite cognitive permet de distinguer des centres de décision locaux ou globaux

travaillera.

L'impossibilité de prendre en compte des informations détaillées sur une grande échelle rend nécessaire leur agrégation pour les rendre plus synthétiques. Chaque centre de décision peut donc être situé dans une pyramide telle que la pyramide CIM, en fonction du degré d'agrégation des informations qu'il perçoit et emmagasine.

Ainsi, une structure pyramidale des centres de décision se distingue par l'utilisation *d'éléments centraux*, c'est à dire de centres de décision ayant une représentation plus globale et agrégée du système à contrôler. À l'inverse, une forme plate est caractérisée par l'absence d'éléments centraux. On pourrait donc la qualifier de structure décentralisée.

Cette topologie influe de manière importante sur le mécanisme utilisé pour la coordination des centres. Ainsi, la hiérarchie est usuellement associée à une forme pyramidale, la coordination entre les centres reposant alors uniquement sur les éléments centraux, tandis qu'une structure décentralisée est associée à des mécanismes hétérachiques de coordination, dans lesquelles la prise de décision se fait conjointement entre des centres de décision égaux.

Mais il est difficile de tracer une frontière nette dans ce qui apparaît comme « un spectre d'architectures de contrôles allant de la hiérarchie à l'hétérarchie » [Brennan et Norrie, 2001]. En effet, forme plate et pyramide tendent à converger vers une forme mixte, ou forme holonique générale [Bongaerts *et al.*, 2000] dans laquelle il existe des éléments centraux, ainsi que des relations entre centres de même niveau (figure 1.2). De même, il est possible, partant d'une structure pyramidale, de créer des liens de coopération entre centres de même niveaux, ce processus de ré-ingénierie constituant en quelque sorte une « holonification » du système de pilotage.

Le critère de topologie du système de décision, bien qu'il soit important, n'est donc pas suffisant. En effet d'autres paramètres sont déterminants, comme par exemple la nature des relations entre centres de décision. Si par exemple le comportement des centres locaux est strictement contraint par les décisions prises par les éléments centraux, on pourrait dire que la prise de décision globale a plus de poids que la prise de décision locale. À l'inverse, si les éléments centraux sont limités à un rôle consultatif, c'est la prise de décision locale qui domine.

Les liens entre les centres de décision ne sont donc pas équivalent, et leur nature doit être prise en compte de manière plus explicite.

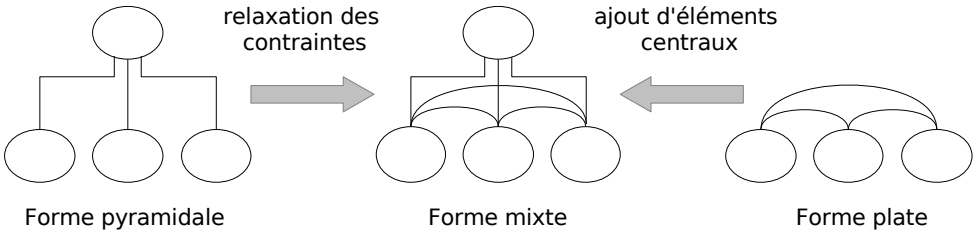


FIG. 1.2 – Convergence des formes plate et pyramidale vers une forme mixte

Ainsi, la manière dont les centres de décision sont coordonnés influe sur le caractère *centralisé* ou *distribué* d'un système de décision. Il est donc nécessaire de considérer les différents mécanismes de coordination.

1.1.3 Mécanismes de coordination

Coordination hiérarchique

Le mécanisme de coordination le plus simple et le plus répandu est la coordination hiérarchique. Celui-ci repose sur une structuration pyramidale des centres de décision, et des flux d'information verticaux. Ces flux correspondent d'une part à la transmission d'ordres des centres les plus globaux vers les centres les plus locaux, et d'autre part en la remontée d'information sur le déroulement de ces ordres, ou plus généralement sur l'état du système à contrôler (figure 1.3).

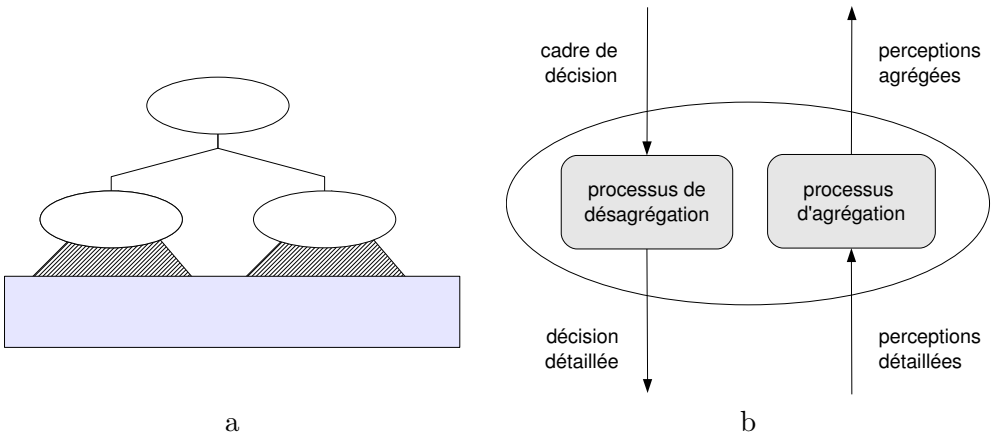


FIG. 1.3 – Structure pyramidale et coordination hiérarchique. Structure du réseau des centres de décision (a), détail d'un centre (b)

La coordination hiérarchique repose donc sur deux processus de traitement de l'information : l'un *d'agrégation*, l'autre de *désagrégation*. Le terme *d'agrégation* ne doit pas être pris ici au sens de l'approche orientée-objets (où plusieurs objets simples s'agrègent pour former un objet complexe, mais continuent à exister en tant que tels), mais plutôt au sens de la gestion de production, où un indicateur global est calculé à partir d'indicateurs locaux (un exemple simple étant de calculer une moyenne). L'agrégation entraîne donc une perte d'information, puisque les valeurs locales disparaissent dans le calcul de l'indicateur global.

Si on considère la hiérarchie comme un mécanisme de coordination, rien ne s'oppose a priori à ce que les processus d'agrégation et de désagrégation évoluent *dynamiquement* pour s'adap-

ter à des changements du système de production à contrôler. Cependant, ces processus sont intimement liés au mode de représentation et de résolution du problème de pilotage (présenté dans la sous-section suivante). Dans la plupart des cas, le pilotage se fait en suivant un schéma d'agrégation/désagrégation prédéterminé *statique* qu'il est très difficile de faire évoluer.

Ce mécanisme de coordination, simple et bien connu, peut être répété tout au long de l'arborescence des centres de décision. Ainsi, en fonction des ordres transmis aux centres inférieurs, et en particulier du degré de liberté qui est conféré pour leurs exécutions, la prise de décision se trouve déplacée vers le centre de niveau supérieur, conduisant globalement à concentrer le pouvoir de décision au sommet de la pyramide des centres de décision. Une coordination hiérarchique est donc caractéristique d'un système de décision centralisée.

D'autres mécanismes de coordination alternatifs ont été proposés. Ceux-ci reposent sur l'imitation de phénomènes de coordination que l'on peut observer dans les sociétés humaines ou animales.

Coordination hétérarchique d'inspiration sociale

Une source d'inspiration importante est l'étude des mécanismes qui sont à l'œuvre lorsqu'un groupe de personnes effectue une tâche collective sans coordination centralisée. Un exemple d'un tel type de coordination est le protocole par réseau contractuel (CNP – *Contract Net Protocol*), introduit au début des années 80 [Smith, 1980]. Ce mécanisme, qui s'inspire du fonctionnement d'un marché, est l'un des premier protocole d'interaction utilisé pour la coordination des systèmes multi-agents. Il a été normalisé par la *Foundation for Intelligent Physical Agents* (FIPA) à l'aide d'un diagramme AUML[1] (figure 1.4). Il peut être décomposé selon ces étapes :

1. l'initiateur sollicite m propositions par l'envoi de messages d'appel-à-proposition (call-for-proposals), spécifiant la tâche à exécuter et une date limite.
2. Parmi les m agents récipiendaires, n répondent effectivement, soit par une proposition (j), soit par un refus ($i = n - j$). La proposition inclut les conditions de l'offre, par exemple son prix.
3. à la date limite, l'initiateur évalue les propositions, et en choisit l (aucune, une, ou plusieurs). Les agents sont notifiés de l'acceptation ou du rejet de leur proposition.
4. Enfin, chaque agent choisi notifie l'initiateur de la réussite ou de l'échec de l'action demandée.

Une utilisation industrielle du CNP peut être trouvée pour la coordination des activités de production dans le système *production 2000+* [Bussmann et Schild, 2001]. Ce système de pilotage a été développé au sein du département de recherche et développement de Daimler-Chrysler. Le système met en œuvre le pilotage d'un système flexible de production grâce à une approche basée sur des agents logiciels. Celle-ci s'adapte à des conditions de production changeantes, et convient à des volumes importants de production. Dans ce système, chaque « agent-pièce » (workpiece-agent), offre aux enchères une tâche correspondant à sa phase de fabrication courante, tandis que les machines enchérissent pour obtenir ces tâches. Lors du processus de choix du meilleur enchérisseur, l'agent-pièce prend en compte non seulement la charge des machines, mais aussi son débit de sortie. Ceci permet de détecter une machine bloquée, et d'éviter de lui envoyer de nouvelles pièces. De plus, un goulot de capacité sera automatiquement propagé en amont du flux.

[1] AUML, ou Agent-UML est une adaptation d'UML (Unified Modeling Language) pour la représentation des système multi-agents.

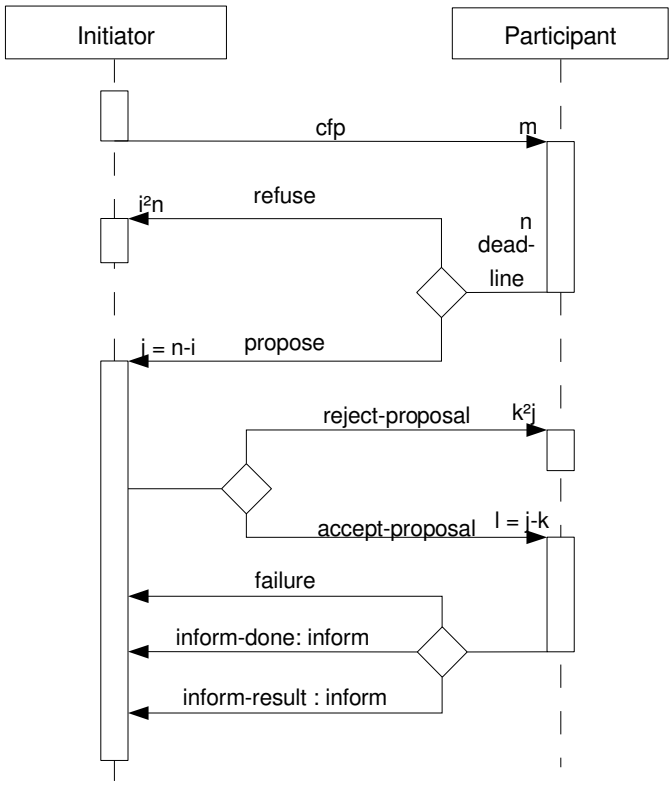


FIG. 1.4 – Le mécanisme de coordination par réseau contractuel (CNP) représenté sous la forme d'un diagramme d'interaction AUML (norme FIPA-SC00029) [FIPA, 2005]

Un mécanisme de coordination hétérarchique relativement semblable a été étudié par une industrie de fabrication de vitres de sécurité[Blanc *et al.*, 2006]. Un protocole supportant la négociation entre les ordres de production à effectuer et les machines, tous deux représentés par des agents logiciels, a été défini. Le dialogue entre les divers types d'agents permet aux ordres d'être ordonnancés sur les ressources.

D'autres protocoles d'interaction s'inspirant de phénomènes sociaux on été énoncés, comme par exemple ceux reprenant le mécanisme des divers types d'enchères. Tous ces mécanismes peuvent être caractérisés par l'échange direct de messages au contenu riche, et des procédures internes complexes, permettant de mémoriser et de réagir à ces messages.

Coordination hétérarchique stigmergique

D'autres types de mécanismes de coordination décentralisés peuvent être observés dans le monde animal. Contrairement aux mécanismes d'inspiration sociale qui reposent sur des échanges de messages explicites (figure 1.5b), les signaux échangés sont beaucoup plus simples. Un type particulier d'échange de signaux, la *stigmergie*, est basé sur un mode de transmission *indirect* (figure 1.5a).

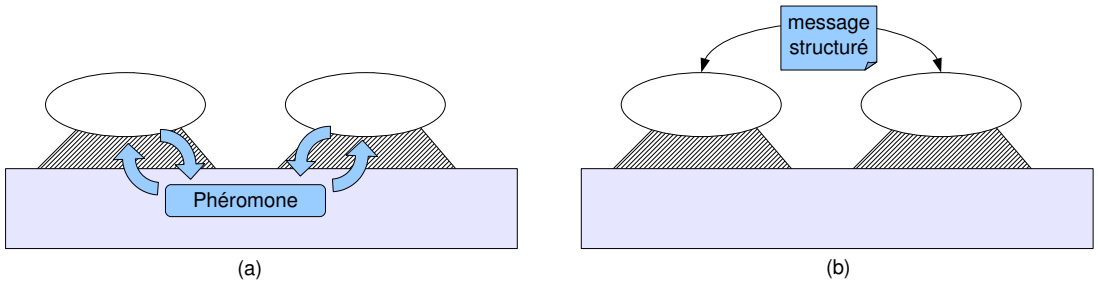


FIG. 1.5 – Coordination à travers l'environnement (a), ou par échange explicite de message (b)

La stigmergie est caractérisée par un échange d'information se faisant grâce à des modifications locales de l'environnement. Ce terme a été introduit par Grassé en 1959 [Grassé, 1959], d'après l'observation de termites construisant collectivement leurs nids. Les termites déposent en effet sur les boulettes de terre des marqueurs chimiques ou *phéromones*. Ces marqueurs déclenchent et conditionnent ensuite la manière dont les boulettes vont être assemblées, conduisant globalement à l'émergence de structures régulières telles que des voûtes ou des cheminées.

La stigmergie, dont les racines étymologiques sont les mots grecs *stigma* (signe) et *ergon* (action, travail), est donc un mécanisme de coordination dans lequel *l'œuvre collective coordonne les ouvriers*.

Une autre source majeure d'inspiration est l'observation du comportement des fourmis devant ramener de la nourriture au nid. Leur problème est la recherche du plus court chemin entre la fourmilière et les sources de nourriture. De plus, les fourmis doivent s'adapter aux changements de l'environnement (ajout d'un obstacle, épuisement d'une source de nourriture, etc...). Le mécanisme —stigmergique— qui est employé repose sur le marquage du sol avec des phéromones.

L'observation des fourmis a permis de formuler un modèle de comportement local comportant quelques règles simples :

- en phase de recherche, se déplacer vers la zone adjacente où la concentration de phéromone est la plus forte,
- si aucune phéromone n'est perçue, aller au hasard,

– lorsqu'une source de nourriture est trouvée, rentrer à la fourmilière, en déposant des phéromones ;

Le phénomène d'évaporation des phéromones, et de renforcement des traces existantes (les fourmis choisissent les pistes les plus marquées) permet la convergence des pistes des fourmis vers le chemin le plus court, ainsi que l'adaptation aux changements de l'environnement. En effet, plus une piste est courte, plus le rapport entre le renforcement et l'évaporation sera bon. De même, une piste devenant impraticable ne sera plus renforcée et finira par disparaître (figure 1.6 [Rennard et Mange, 2002]).

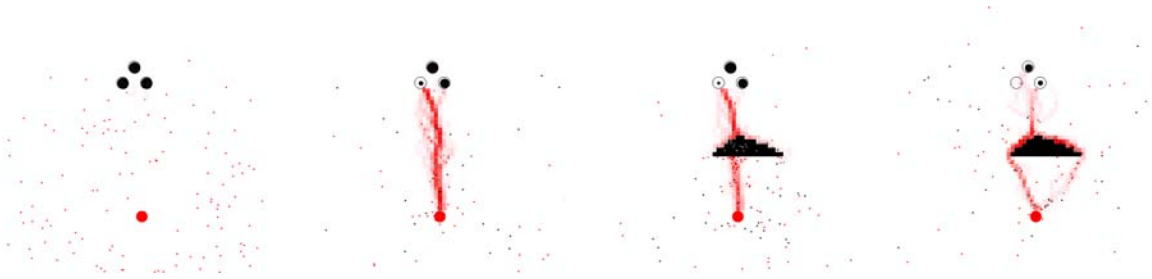


FIG. 1.6 – Simulation d'une colonie de fourmis. *a* : initialement, les fourmis se déplacent au hasard. *b* : lorsqu'une source de nourriture est trouvée, un chemin de phéromone se crée. *c* : un obstacle est ajouté *d* : les fourmis s'adaptent à la présence de cet obstacle en créant de nouveaux chemins. L'ancien chemin, devenu obsolète, disparaît par évaporation.

Ces règles définissent en fait une méta-heuristique, dénommée *optimisation par colonie de fourmis* (Ant Colony Optimisation – ACO). Celle-ci permet de résoudre la classe des problèmes apparentés au problème du voyageur de commerce[2].

L'optimisation par colonie de fourmis a donc été transposée à divers domaines d'application, comme par exemple le routage dynamique dans les réseaux de télécommunication, mais surtout au pilotage de la production [Valckenaers *et al.*, 2006b] [Hadeli *et al.*, 2005]. Il est supposé que chaque ressource de transformation est liée à d'autres ressources par des équipements de transport. Le routage des produits est donc flexible ; c'est ce paramètre qui va être optimisé. L'espace dans lequel on cherche un parcours optimal est donc constitué par le réseau des équipements de transformation ou de transport. Chaque produit (pouvant n'être qu'à l'état d'ordre de fabrication) génère des fourmis, qui représentent un parcours potentiel dans l'atelier.

Toutefois, le pilotage de la production n'appartient pas à la famille du problème du voyageur de commerce. En effet, deux tâches ne peuvent être accomplies simultanément sur une ressource. Cette exclusion mutuelle rend le problème beaucoup plus complexe, car en plus de la dimension spatiale, le temps doit être pris en compte.

De plus, il est nécessaire de prendre en considération les limitations techniques des équipements (opérations pouvant et ne pouvant pas être réalisées par cette ressource), les contraintes de précédence entre les opérations devant être effectuées sur le produit (gamme de fabrication). Cette difficulté supplémentaire est résolue en utilisant différents types de fourmis :

– *les fourmis de faisabilité* permettent de discerner les routages valides, en propageant vers l'amont les capacités de chaque ressource.
– *les fourmis d'exploration* sont générées périodiquement par les ordres de fabrication ; elles se déplacent sur le réseau virtuel des ressources, et servent à estimer la performance de

[2]Dans ce problème classique de recherche opérationnelle, il s'agit de déterminer l'ordre dans lequel on devra visiter n villes, de manière à minimiser la distance totale parcourue

chaque routage.
– *les fourmis d'intention* représentent le choix par un ordre de fabrication d'un certain routage ; les phéromones déposées jouent le rôle de réservation des ressources.

Coordination hétérarchique en essaim

Enfin, on peut observer des phénomènes de coordination dans le comportement animal, sans qu'aucun signal explicite ne soit échangé, la coordination résultant de l'observation par chaque individu des caractéristiques visibles de ses pairs. Les exemples les plus clairs d'une telle coordination sont le déplacement en groupe des animaux, par exemple la formation des bancs des poissons, le vol en V des oies sauvages, etc...

L'étude et la modélisation de ces mécanismes de coopération hautement décentralisés est le centre d'intérêt principal de deux communautés. D'une part, du point de vue l'intelligence artificielle, la communauté *swarm intelligence*, a formulé des résultats théoriques comme l'optimisation par essaim particulaire (*Particle Swarm Optimisation*) [Parsopoulos et Vrahatis, 2002]. D'autre part, la communauté *cooperative control* [Kumar *et al.*, 2004] considère ces mécanismes de coordination du point de vue de l'automatique théorique, et plus particulièrement leurs applications à la robotique mobile.

Les algorithmes de comportement des entités mobiles sont composés de règles élémentaires, par exemple l'attraction / répulsion ou la poursuite d'un leader. L'application de ces règles au contrôle de robots mobiles, met en évidence la convergence, sous certaines conditions, vers des structures régulières et stables [Marshall *et al.*, 2006].

L'application directe de ces mécanismes de coordination aux problèmes de pilotage de la production est tentante, mais la complexité et le caractère discontinu du milieu dans lequel les produits évoluent (l'usine, l'atelier) les rendent difficilement exploitables. Leur étude permet néanmoins de mettre en évidence des propriétés fondamentales des systèmes distribués telle les problématiques de convergence et de stabilité du système.

On constate au final qu'il existe de nombreux types de mécanismes d'interaction, plus ou moins hétérarchiques. En effet, les mécanismes d'inspiration sociale requièrent le partage d'une sémantique commune et des communications nombreuses, tandis que dans les mécanismes d'inspiration animale ces échanges sont plus réduits.

De même que ces mécanismes étaient fortement corrélés à la structure du système de décision (une coordination hiérarchique étant par exemple inapplicable à une forme plate), la manière de *représenter* et de *résoudre* le problème de décision dépend du mécanisme de coordination du système.

1.1.4 Formulation du problème de décision

La manière de représenter le système opérant et de formuler le problème de décision constituent un autre point de différenciation entre les modes centralisés et distribués. On a vu qu'une coordination hétérarchique conduisait à un comportement globalement cohérent à partir de l'ensemble des comportements locaux en interaction. Il y a donc *émergence* d'une résolution du problème de pilotage, sans qu'à aucun moment ce problème n'ait été formulé explicitement et globalement.

À l'inverse, la coordination hiérarchique repose sur la modélisation explicite du problème dans sa globalité, et sa décomposition en sous-problèmes devant être résolus par les centres hiérarchiquement inférieurs.

On peut donc parler d'une formulation et d'une résolution *émergente* du problème de pilotage, relative à l'approche distribuée, ou d'une formulation et d'une résolution *prescrite*, dans l'approche centralisée.

Un exemple de résolution prescrite d'un problème de décision peut être trouvé dans le système de planification hiérarchique (HPS *hierarchical planing and scheduling*) de la méthode MRP2[3] [Vollmann *et al.*, 1997]. En effet, la méthode MRP spécifie de décomposer le processus de décision en *fonctions* de la manière suivante :

- *La planification industrielle et commerciale (PIC)* correspond à un ensemble de décisions à moyen terme permettant de mettre en œuvre la stratégie de l'entreprise. En particulier, elle fournit les quantités à produire par mois, pour chaque famille industrielle de produits.
- *Le programme directeur de production (PDP)* précise pour chaque produit fini les quantités à produire, période par période.
- *Le calcul de besoins nets et la planification des capacités* permet d'atteindre les objectifs du plan directeur de production en générant les ordres de fabrication des composants et les ordres d'achat de matière première, tout en tenant compte des capacités des ressources.
- *L'ordonnancement et le pilotage d'atelier* traitent dans le détail des problèmes d'affectation et de contrôle des flux physiques sur le court terme.

Le point de départ de l'approche centralisée est donc un modèle global et agrégé du problème à résoudre. Ce modèle, qui doit explicitement donner la structure et de l'état du système opérant, peut être exploité grâce à des techniques de recherche opérationnelle (algorithmes, heuristiques ou méta-heuristiques) pour trouver une solution optimale, au sens où elle satisfait au mieux une fonction objectif, et respecte les contraintes du problème. Cette solution est ensuite utilisée comme base pour une résolution moins agrégée du problème, pouvant être basée sur une modélisation différente (par exemple sur une autre fonction objectif[4]) [Bitran et Hax, 1981] [Ortiz-Araya, 2005].

Cette technique est à comparer avec une approche émergente. Dans cette approche, le problème global n'est pas directement considéré. La modélisation se fonde sur les *objets* constituant le système opérant. Chaque entité du monde réel est donc dotée d'une contrepartie informationnelle, qui reflète son état et ses propriétés. L'architecture de référence PROSA [van Brussel *et al.*, 1998] spécifie la manière dont chaque composant du système à piloter est représenté. Après l'avoir présenté, nous verrons en quoi elle permet d'utiliser l'émergence.

PROSA est un acronyme pour *Product Resource Order Staff Architecture*. En effet, cette architecture est composée de trois types d'holons (figure 1.7), auxquels s'ajoute les holons staff :

les holons produits représentent le savoir-faire utile à la réalisation des produits. C'est en quelque sorte *la recette* de fabrication d'un type de produit ;

les holons ressources représentent les équipements permettant de réaliser la production ;

les holons ordres représentent les ordres de fabrication devant être exécutés

les holons *staff* dont la présence est optionnelle, permettent d'intégrer au système des centres de décision ayant un champ de vision plus large, par exemple un centre effectuant un processus de planification ou d'ordonnancement.

[3] La méthode de gestion de production MRP a été introduite dans les années 60. Initialement, elle était axée sur la nomenclature, pour effectuer le calcul des besoins en composants (Material Requirement Planning), mais elle a par la suite évolué pour englober des fonctions de planification à moyen terme, et de gestion de la capacité (Manufacturing Resource Planning)

[4] la décomposition d'une décision agrégée en sous-décisions (par exemple la décomposition du PIC en PDP) est l'objet du processus de *désagrégation*. Le problème posé n'est pas trivial, surtout si l'on cherche à conserver des propriétés de robustesse lors de la décomposition.

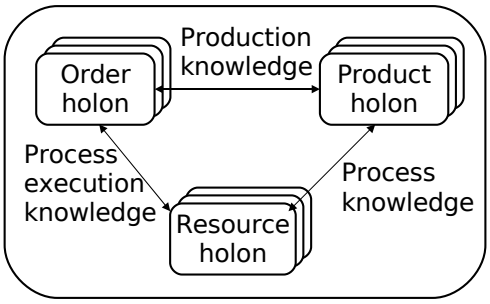


FIG. 1.7 – Les trois types principaux d'holons dans l'architecture PROSA

L'idée principale de PROSA est donc de *refléter* dans le système d'information les éléments existants dans le système physique. Cette approche de modélisation est voisine de l'approche orientée-objet, qui s'est progressivement imposée dans le génie logiciel ; l'architecture PROSA utilise donc les concepts orienté-objet, tels que les généralisations, ou les agrégations. En effet, la généralisation permet de représenter de manière générique les différents éléments composant le système de production. Ainsi, par spécialisations successives, on pourra passer d'une entité générique (par exemple un holon-ressource), à une entité partiellement générique (un robot), et enfin particulière (un robot FANUC six axes S-10). De même, les relations d'agrégation sont importantes, car elles permettent de représenter la hiérarchie des entités du système de production. Par exemple, la nomenclature d'un produit fini définit une hiérarchie d'holons-produits. Des hiérarchies semblablement basées sur les liens d'agrégation existent pour les holons-ordres (lots, ordres de fabrication, etc...) et les holons-ressources (atelier, cellule, machine, etc...).

En utilisant les éléments génériques de PROSA, il est donc possible de représenter un système de production sans jamais devoir le considérer globalement. En munissant de plus ce système d'un mécanisme de coordination hautement hétérarchique tel qu'un pilotage par fourmis (comme présenté précédemment), il est possible de résoudre un problème de pilotage sans le modéliser explicitement[5]. Par conséquent, les changements du système de production seront implicitement pris en compte. Un pilotage émergent de la production a donc des capacités d'adaptativité et d'auto-organisation. De plus, les tâches de collecte d'information et de maintenance du modèle sont ainsi grandement simplifiées.

La manière de formuler le problème est liée à la manière de découper le système de décision en centres. Ainsi, l'approche centralisée, basée sur une décomposition en sous-problèmes, sera fonctionnelle et descendante (*top-down*), tandis que l'approche distribuée, basée sur les interactions des objets du système de pilotage, est orientée-objet et montante (*bottom-up*).

1.1.5 Dynamique

Enfin, une dernière caractéristique à prendre en compte est la dynamique du système de décision. Ceci a d'abord été réalisé en spécifiant l'horizon et la périodicité de la prise de décision. Ainsi, la méthode GRAI [Doumeingts, 1984] de modélisation d'entreprise définit un niveau de décision par son horizon et sa période. Cette structuration correspond bien à une organisation hiérarchique (mode centralisé) des centres de décision, mais fait l'hypothèse d'une exécution périodique. Elle s'applique donc à une prise de décision *ponctuelle*.

[5]Les mécanismes basés sur des échanges explicites de messages nécessitent toutefois une modélisation des informations échangées entre les centres de décision.

En effet, une analyse plus fine montre qu'il existe deux types de problèmes [Wegner, 1997] [Valckenaers, 2001] : les problèmes ponctuels (*one-shot*) et les problèmes d'évolution (*on-going concerns*). Dans le problème one-shot, il s'agit de remplir une contrainte à un instant donné, sans que les états précédents et suivants du système ne soient d'importance. Un exemple d'un tel problème est de livrer une certaine quantité de produits, en respectant les délais et les contraintes de qualité. Dans le problème on-going, il s'agit de *maintenir pendant une durée* le système dans un état souhaité.

La résolution d'un problème one-shot peut être représentée mathématiquement par une fonction, dont les entrées sont les paramètres du problème, et dont la sortie en est la solution. Alternativement, la résolution d'un problème en évolution ne peut pas être faite par une fonction, car elle exige une interaction permanente avec le système à contrôler [Goldin et Wegner, 2005]. On peut pour s'en convaincre, considérer par exemple le problème posé par la conduite automatique d'une voiture. Une fonction devant résoudre ce problème devrait prendre en paramètre une description infiniment précise de la forme de la route et de la dynamique du véhicule... Paramètres impossibles à fournir avec certitude, surtout si d'autres véhicules sont présents (c'est à dire si l'environnement est perturbé).

Ces deux notions sont à mettre en parallèle avec celle de *synchronisme* et de *diachronisme* [Le Gallou et Bouchon-Meunier, 1992]. Dans l'axe synchronique, on observe le système dans son ensemble *à un instant donné*, tandis que dans l'axe diachronique, on observe l'évolution *à travers le temps* d'un élément du système. Le tableau 1.1 fait le parallèle entre la nature du problème, l'axe d'observation, et le déclenchement des activités de décision.

Problème	**Axe**	**Déclenchement**
one-shot	synchronique	périodique
ongoing	diachronique	événementiel

TAB. 1.1 – Parallèles entre la nature du problème, l'axe d'observation, et le déclenchement des activités de décision

Dans le domaine du pilotage de la production, l'approche centralisée utilise principalement des méthodes de résolution one-shot. Elles reposent sur la capacité des algorithmes de recherche opérationnelle à fournir un plan d'action optimal en fonction de l'état courant du système. Ainsi, les informations collectées sur le système opérant sont les paramètres d'entrée d'une fonction, dont le résultat constitue l'optimum à mettre en œuvre par la suite. Le déroulement de ce plan optimal est ensuite périodiquement vérifié, et une dérive trop importante conduit à une ré-exécution des procédures de décision (ré-ordonnancement). C'est le fonctionnement classique d'un logiciel d'ordonnancement.

Alternativement, les approches de contrôle distribué, en particulier les systèmes holoniques de production, adoptent un mode de résolution on-going. Les centres de décision sont en interaction permanente avec leur environnement (système opérant, pairs, acteurs externes, ...). L'arrivée d'un événement déclenche les procédures de décision, produisant si besoin une décision correctrice. Les systèmes d'inspiration biologique ou économique se comportent de cette manière. Un système d'ordonnancement temps-réel peut aussi être classé dans cette catégorie, dans la mesure où il maintient un ordonnancement, et le met à jour à l'arrivée de chaque nouvel OF.

1.1.6 Classification

Les critères que nous avons définis sont corrélés. En effet, l'organisation hétérarchique ou holonique est liée à une représentation du domaine d'intérêt par objets, et à une résolution en évolution des problèmes. Inversement, la résolution au coup par coup est facilitée par une décomposition fonctionnelle, et par une organisation hiérarchique.

Mode de décision	**Centralisé**	**Distribué**
Structure	pyramidale ou mixte	plate ou mixte
Coordination	hiérarchique	hétérarchique (bio– ou socio–inspirée)
Représentation et mode de résolution	fonctionnelle et descendante prescrite	orientée-objet et montante émergente
Dynamique	périodique	continue

TAB. 1.2 – Caractéristique des modes centralisé et distribué de décision

On définit donc deux manières de concevoir un système de contrôle de la production, le *mode centralisé* et le *mode distribué* de décision. Ces deux modes se définissent par rapport aux caractéristiques des systèmes de décision (tableau 1.2).

1.2 Problème du couplage entre décisions centralisées et décisions distribuées

Les caractéristiques des approches centralisées et distribuées du pilotage de la production déterminent en partie leurs domaines d'application préférentiels. Les apports et limites des deux modes de prise de décision, ainsi que les pratiques industrielles, nous conduisent donc à des systèmes de niveau *business* étant *de fait* centralisés, et des systèmes de niveau *process* pouvant être distribués. La nécessaire coexistence de ces deux types de systèmes pose la problématique de leur interaction.

1.2.1 Apports et limites respectifs

Approche centralisée

En se basant sur des techniques matures de recherche opérationnelle, l'approche centralisée du pilotage de la production permet de prendre par anticipation des décisions a priori *globalement optimales*. Cette formulation globale du problème permet une cohérence et une performance selon deux axes :

- dans l'axe temporel, la prise en compte d'un large horizon assure de pouvoir réagir de manière optimale, par exemple en répartissant la charge pour faire face aux variations de la demande (lorsqu'elles peuvent être anticipées).
- dans l'axe spatial, le système opérant étant considéré comme un tout, les interactions entre équipements peuvent être prises en compte, conduisant à un fonctionnement globalement optimal.

Du point de vue de l'ingénierie du système de décision, l'approche centralisée offre un modèle simple et clair, dans lequel le rôle de chaque centre est bien défini. La planification MRP2, qui utilise cette approche, est mature et très largement utilisée industriellement. Elle souffre néanmoins de certaines limites.

La principale limite de l'approche centralisée est la désynchronisation entre les flux de matière de l'atelier et la représentation informatique de ces flux [Plossl, 1985]. Cette limite est causée par la manière de formuler le problème de pilotage et de représenter l'atelier, ainsi que par la structure du système de décision, qui multiplie les niveaux hiérarchiques.

En effet, la prise de décision centralisée requiert un *modèle explicite*, suffisamment précis, fidèle et complet, dont il faut maintenir la correspondance avec le terrain. Les désynchronisations entre ce modèle et l'état réel du système de production s'expriment à court terme dans les décisions d'exécution de la production. Les procédures centralisées de décision prennent ainsi en compte de manière prédictive une certaine portion du futur, et produisent des décisions optimales par rapport à l'état prédit du système de production. L'apparition d'aléas invalide ces prédictions, et remettent en cause les décision prises. Ces aléas peuvent porter sur les capacités des ressources (e.g. pannes, absence d'opérateur), sur les tâches à effectuer (temps de cycle variables, rebuts), ou même sur le besoin à satisfaire (modification d'une commande).

Les évolutions à plus long terme du système de production, comme les re-configurations, peuvent elles aussi conduire à une désynchronisation. Pour maintenir à jour le modèle, il est nécessaire de collecter des données sur le terrain, et éventuellement de changer le modèle de décision, tâche longue qui nécessite un haut niveau d'expertise.

De plus, la multiplication des niveaux hiérarchiques peut causer une déformation de l'information. Ainsi, au plus haut niveau hiérarchique, la vision du terrain peut avoir été considérablement déformée par les agrégations successives d'information. De même, cette déformation peut se produire dans la mise en œuvre des décisions (flux descendants). En particulier, dans la logique MRP2, on raisonne d'abord à capacité infinie (planification), avant de descendre vers des modules fonctionnant à capacité finie (ordonnancement). Cette simplification pouvant conduire à la génération de plans non réalisables. D'autre simplifications (délais statiques d'obtention, définition rigide des nomenclatures, des gammes, etc...) empêchent la prise en compte des capacités réelles de l'atelier.

Une coordination hiérarchique et une prise de décision périodique peut aussi être la cause d'une perte de réactivité. En effet, la différence entre les périodes de chaque niveau hiérarchique conduit à des délais pouvant être importants entre une prise de décision et sa mise en œuvre sur le terrain (figure 1.8). Ce délai de propagation de l'information entre niveaux hiérarchiques conduit donc à une perte de synchronisation entre les plans.

Ce problème de réactivité est en partie résolu grâce à l'intégration des centres de décision dans des APS. Les APS, ou systèmes de planification avancée (*Advanced Planing System*) résultent de l'évolution des systèmes MRP2. Ce sont des progiciels qui optimisent la planification et synchronisent les flux de la chaîne logistique en tenant compte simultanément d'un grand nombre de contraintes. Ils utilisent des technologies telles que les systèmes de gestion de bases de données relationnelles, les architectures client-serveur ou les techniques d'échange de données (EDI) pour centraliser l'ensemble des plans d'une chaîne logistique au sein d'un même outil.

Cette centralisation des données permet d'accélérer la propagation des contraintes entre les différents plans. D'une part, cela permet d'assurer la cohérence des plans entre les maillons de la chaîne logistique, et d'autre part, d'accélérer les flux d'informations verticaux. Ainsi, si un plan s'avère remis en question en un point de la chaîne logistique, il sera possible, grâce à la réactivité accrue de l'APS, de générer un plan tenant compte de l'état des systèmes situés en amont et en aval.

Cela pose néanmoins le problème de l'hyper-sensibilité des décisions globales aux aléas locaux. Connu sous le terme de *nervosité* , ce phénomène cause le changement incessant des plans

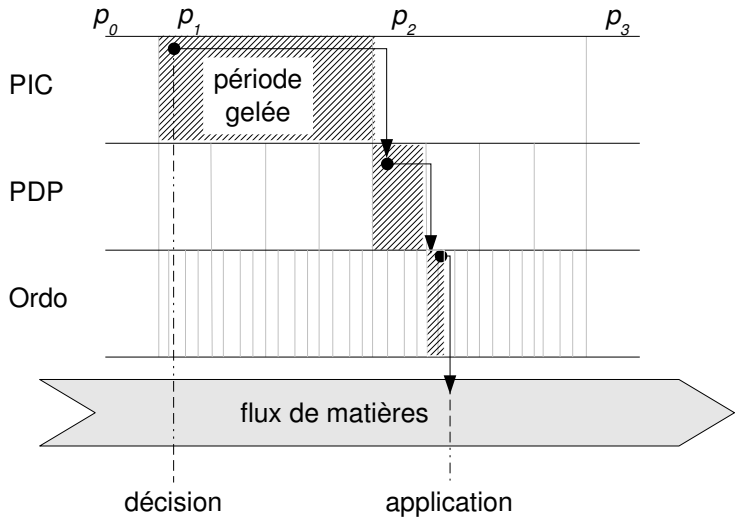


FIG. 1.8 – Désynchronisation des plans du MPR2

tactiques, ce qui est très gênant pour le fonctionnement de l'entreprise. Il est toutefois possible d'atténuer cette nervosité, en agissant au niveau du PIC [Génin, 2003], ainsi que lors de la désagrégation du PIC en PDP [Ortiz-Araya, 2005]. Ces progiciels relativement nouveaux ne résolvent pas, par contre, le problème de la modélisation du système de production.

Approche distribuée

L'utilisation de techniques d'intelligence artificielle dans l'approche distribuée lui confère des qualités d'adaptabilité à des conditions changeantes et incertaines. La localisation des prises de décision au plus près du terrain lui donne une grande réactivité aux aléas. De plus, l'utilisation d'une approche objet facilite les tâches de création et de maintenance d'un modèle global, puisque les propriétés globales du système *émergent* d'un ensemble de comportements simples. La simplicité même du fonctionnement interne des centres de décision leur confère une grande réutilisabilité.

La prise de décision distribuée est couramment critiquée par rapport à son comportement essentiellement réactif, pouvant conduire à des décisions qui ne sont pas globalement optimales (on peut parler de « systèmes myopes », dans la mesure ou seules les conséquences immédiates dans le temps ou l'espace sont prises en compte). L'approche distribuée parait de ce point de vue relativement inadaptée aux décisions ayant un impact à moyen ou long terme. Si cette critique est fondées dans de nombreux cas, elle est infirmée par des développements récents, visant à augmenter la proactivité de la prise de décision décentralisée. C'est par exemple le cas de l'approche par colonie de fourmis [Valckenaers *et al.*, 2006b]. Il est donc possible de doter les systèmes distribués d'une certaine dose de proactivité ; mais cela se fait au pris d'une augmentation de la complexité de ces systèmes.

Par conséquent, la principale limite de l'approche distribuée est la complexité et l'éventuelle instabilité des systèmes distribués, et la difficulté de conception qui en résulte.

Cette complexité est en grande partie due à l'existence de comportements émergents. En effet, le phénomène d'émergence est mal compris, et très peu formalisé. La connaissance des comportements locaux ne permet pas de prédire a priori les propriétés globales du système

comme la convergence vers le comportement global souhaité, et la stabilité du système. Il est ainsi difficile de prouver que va émerger un comportement global cohérent et stable ; de nombreux systèmes distribués peuvent conduire à l'émergence d'un comportement déviant (*emergent misbehaviour*), posent d'importants problèmes de convergence (lorsque les centres entrent en résonance *livelock*) ou de vivacité (inter-bloquages des centres de décision). Cela pose donc le problème de la validation et de l'évaluation des performances de tels systèmes avant leur mise en œuvre.

A fortiori, il est encore plus difficile de concevoir les comportements locaux à mettre en œuvre pour obtenir une propriété globale souhaitée. Le seul mode d'ingénierie serait donc de s'inspirer des systèmes biologiques, ou bien d'adapter des algorithmes existants, sans pouvoir s'affranchir de cette conception par mimétisme.

L'utilisation de mécanismes moins décentralisés facilite la maîtrise de l'émergence, mais en retour, l'emploi de messages riches contraint à spécifier explicitement les objets du domaine sous la forme d'une ontologie. La synchronisation de cette ontologie avec la réalité peut là encore, poser problème.

1.2.2 Nécessaire co-existence des approches centralisées et distribuées

D'après les apports respectifs des approches centralisées et distribuées de pilotage de la production, nous pouvons faire apparaître deux champs d'application préférentiels.

D'abord, l'approche centralisée, de par sa prise en compte globale du système, et sa capacité à programmer les activités futures, permet de remplir un ensemble de fonctions que l'on peut désigner comme la gestion logistique, ou la gestion de la chaîne logistique. C'est l'ensemble des activités de planification et de mise en œuvre, permettant de coordonner les activités de chaque acteur de la chaîne logistique. Ces activités incluent l'approvisionnement, la production, la distribution, la vente.

Les systèmes MRP2, très répandus et matures, y sont d'ailleurs très largement employés, puisqu'ils sont intégrés dans la plupart des ERP. La prise en compte de la complexité de la chaîne logistique, en particulier la planification à long terme des activités rendue nécessaire par les délais importants de transferts de matière et d'approvisionnement, ne semblent pouvoir être actuellement pris en compte que par un système centralisé.

Ensuite, l'approche distribuée, de par ses capacités de réactivité et d'adaptation à des conditions changeantes, correspond aux besoins du domaine de l'exécution de la production.

L'exécution (ou pilotage) de la production, est l'ensemble des activités de coordination des équipements industriels, des matières, de l'énergie et des ressources humaines, afin de convertir les matières premières en produits finis. Cela inclut par exemple des activités telles que :

- La traçabilité et la généalogie des produits
- L'affectation des ressources (matières, équipements, personnels) en fonction de leurs états
- La distribution des ordres de fabrication/lots
- La gestion de la qualité et du process (SQC/SPC)
- La gestion des indicateurs de maintenance
- La gestion de la performance des équipements

Les systèmes de pilotage intelligents sont les plus à même à répondre aux besoins croissants de gestion de la variété, de robustesse face à un environnement incertain et de reconfigurabilité, de par leurs caractéristiques d'auto-organisation et d'adaptativité. De plus, le pilotage de la

production est par nature un problème d'évolution, puisqu'il s'agit de maintenir le système en condition dans un environnement perturbé et incertain. Un système diachronique est donc requis.

Un système de production idéal comporterait donc les deux approches, centralisées et distribuées de prise de décision. Celles-ci, grâce à un système assurant un couplage et un bon équilibre, conjugueraient leurs avantages. Toutefois, comme le notent Bongaerts *et al.*, trouver le bon équilibre entre les fonctions centralisées et distribuées n'est pas trivial [Bongaerts *et al.*, 2000].

1.2.3 Adjonction d'éléments centralisés dans un système distribué

La réalisation de tels systèmes de pilotage *hybrides* (au sens où ils combinent l'approche centralisée et distribuée) a été l'objectif de la communauté HMS (*holonic manufacturing systems*) depuis une dizaine d'années.

Le concept de *holon* a été introduit par Koestler [Koestler, 1976], dans son livre *The Ghost in the Machine*. Ce terme est un néologisme combinant le mot grec *holos* (le tout) et le suffixe *–on* qui suggère une particule ou une partie (comme dans électron ou proton).

Koestler affirme que, bien qu'il soit facile de décomposer un tout en parties, ni les parties ni les « tout » n'existent au sens absolu dans la nature. Les organismes vivants, ou les sociétés humaines, peuvent ainsi être vus comme une hiérarchie d'entités semi-autonomes, décomposées elle-mêmes en sous-entités semi-autonomes. Ces entités, à tout niveau hiérarchique, sont appelées holons. Les holons sont donc des systèmes ouverts et auto-régulés, qui présentent les capacités d'autonomie d'un tout (tendance à l'affirmation de soi), mais aussi les capacités d'obéissance d'une partie (tendance à l'intégration dans le tout).

Les systèmes de pilotage holoniques (HMS – Holonic Manufacturing systems) se fondent donc sur cette dualité entre d'une part l'autonomie et la robustesse de l'holon, et d'autre part sa soumission au contrôle venant d'holons hiérarchiquement supérieurs [Valckenaers, 2001]. Un centre de décision holonique devra donc posséder ces deux composantes comportementales.

L'architecture de référence PROSA, qui a déja été évoquée, a été définie pour le développement de systèmes de contrôle holoniques. Dans cette architecture, l'intégration de l'approche centralisée de contrôle passe par des holons spécifiques, dont le pouvoir est limité à un rôle *consultatif*. Le problème qui se pose alors est le manque de visibilité sur l'exécution de la production par les fonctions de gestion de la logistique. En effet, la connaissance du système de pilotage centralisé est assujettie aux informations données par le système distribué. Puisque les décisions du système centralisé ne sont pas forcément respectées, il est impossible de prédire efficacement l'évolution future de l'atelier.

Les systèmes de pilotage distribués anthropocentriques, par exemple ceux basés sur un certain pouvoir de décision et d'autonomie conféré aux opérateurs, posent eux aussi ce problème d'invisibilité des décisions du terrain.

Ce manque de visibilité est d'autant plus dommageable qu'il va à l'encontre des exigences croissantes de traçabilité des produits et plus généralement de suivi au plus près de la production : comment informer un client de l'avancement de sa commande, comment choisir d'accepter ou de refuser une nouvelle commande (*available to promise*).

1.2.4 Encadrement des décisions distribuées par les décisions centralisées

Une autre manière de réaliser le couplage entre une approche centralisée et une approche distribuée du pilotage est de contraindre les prises de décisions dans le système distribué à respecter un cadre fixé par le système centralisé.

Pour générer un espace de choix suffisant pour le système distribué, on peut jouer sur la précision du cadre de décision. En effet, donner aux centres hiérarchiquement inférieurs un cadre plus vaste leur permet de traiter localement les problèmes rencontrés lors de l'exécution de la décision. On peut ainsi utiliser des *plans flexibles*. Par exemple, il est possible de fournir un ensemble d'ordonnancements admissibles, à la place d'un unique ordonnancement, afin de préserver la flexibilité [Artigues *et al.*, 2005]. Il est aussi possible de donner le cadre de décision sous la forme d'un *plan flou*. Ainsi, des travaux utilisant la logique floue, portent sur la prise en compte de l'imprécision et de l'incertitude dans la méthode MRP [Grabot *et al.*, 2005].

La prise de décision dans le système distribué cherchera à réaliser le pilotage en respectant le cadre fixé. À l'occurrence d'aléas, une solution correctrice (relevant éventuellement de mécanismes hétérarchiques de coordination, par exemple une négociation) sera recherchée. Si cette recherche collective d'une solution échoue, le système centralisé est notifié, et doit réagir en conséquence [Tranvouez *et al.*, 2006].

On peut néanmoins craindre, si les décisions centralisées encadrent la prise de décision distribuée, que l'espace de liberté conféré à ces dernières ne soit considérablement restreint. En effet, l'accroissement de l'intégration des décisions, lié à l'utilisation croissante des APS se traduit concrètement par une augmentation de la précision de ces décisions.

On constate donc que les tentatives de couplage entre les approches centralisées et distribuées du pilotage de la production reposent sur l'intégration directe entre les deux types de systèmes de décision, l'un des deux dominant l'autre. Cette organisation du pilotage est résumée par une vision « en trois couches » du pilotage de l'entreprise, basée sur la hiérarchie et l'intégration (figure 1.9).

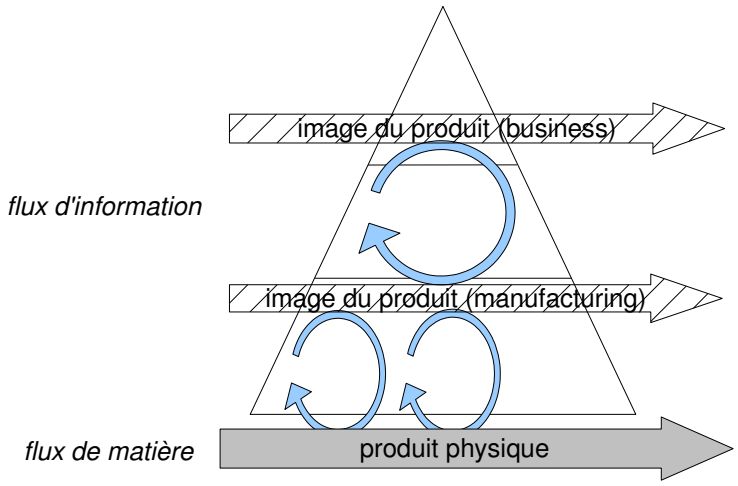


FIG. 1.9 – Problème de la synchronisation des flux de matières et des différents flux d'information

Dans cette vision classique de l'organisation du pilotage, trois flux distincts cohabitent :
- les produits physiques,
- la représentation des produits du point de vue de l'exécution de la production, obtenue en observant les flux physiques,
- la représentation des produits du point de vue de la gestion logistique, obtenue par synchronisation avec les fonctions d'exécution de la production.

Le problème du couplage entre gestion logistique et exécution de la production revient à syn-

chroniser ces flux.

Le principe alternatif de *contrôle par le produit* repose une vision différente de l'entreprise, qui conduit à envisager différemment le problème de l'interaction centralisé/distribué, en postulant que le produit est l'élément central du système d'information, et plus particulièrement du système de pilotage.

1.3 Contrôle par le produit

Cette section a pour but de faire l'état de l'art des recherches sur le contrôle par le produit. Nous évoquerons d'abord les systèmes kanban, que l'on peut considérer comme une forme primitive de contrôle par le produit, puis les recherches récentes motivées principalement par le développement des technologies d'identification par radio-fréquences, et enfin les projets internationaux et nationaux qui se créent autour du contrôle par le produit.

1.3.1 Une pratique nouvelle ?

L'association de données aux produits n'est pas, en temps que telle, une nouveauté. Associer à un lot de pièces une fiche suiveuse indiquant la nature du travail à réaliser est une pratique très courante. De même, il est courant de rendre compte d'une opération (d'un test qualité en particulier) par une marque sur une pièce. Prendre le produit comme pivot constitue en fait l'essence de la productique. Ainsi, dans le domaine de la conception, les environnements d'ingénierie collaborative[6] se sont naturellement organisés autour d'un maquette numérique du produit.

Cependant, dans le domaine du pilotage, ces pratiques centrées sur le produit souffrent de limites importantes. Ainsi, les lectures et écritures sont difficilement automatisables, la dissociation entre le produit et le support de l'information peut être génératrice de pertes de données, et l'intégration des données des produits dans le système d'information de l'entreprise est difficile.

Par ailleurs, le kanban peut être vu comme une forme très simplifiée de contrôle par le produit. C'est une méthode de gestion à court terme bien connue, qui a été appliquée dès 1953 dans les usines Toyota. Dans un système de pilotage par kanbans, le travail sur un poste est déterminé par la présence des étiquettes kanban, celles-ci constituant une sorte de signal envoyé par le produit, lorsqu'il est consommé sur le poste en aval. On peut donc voir le kanban comme un mécanisme de coordination centré sur le produit, entre centres de décision dédiés à l'exécution de la production (les opérateurs). Ce mécanisme permet de mettre en œuvre une production en flux tiré. Toutefois, le domaine d'application du kanban est limité aux productions stables et répétitives. Les productions de forte variété, en particulier réalisées à la commande, ne s'y prètent guère.

Il existe donc des pratiques industrielles pouvant s'apparenter à du contrôle par le produit, mais elles souffrent de limites causées en partie par l'impossibilité d'intégrer les données liées au produit au système d'information. Les évolutions récentes des techniques, qui remettent en cause ces limites, conduisent à proposer de nouvelles techniques de pilotage.

1.3.2 Apport des technologies infotroniques

Les technologies infotroniques, qui permettent d'intégrer les objets (et les personnes) dans un système d'information, rendent possible de nouvelles pratiques de décision, centrées sur le produit.

[6] par exemple la plateforme SmartTeam, édité par Dassault Systems

Ces technologies infotroniques reposent sur des techniques de perception et d'identification automatiques. L'identification par radio-fréquence (RFID) est la technologie qui connaît le développement le plus fort. En utilisant des technologies RFID, il est possible d'automatiser l'identification des produits, ainsi que d'automatiser l'écriture et la lecture de données sur le produit. Comme nous l'avons évoqué, ces pratiques d'identification et d'annotation ne sont pas nouvelles ; Il y a donc qu'une *différence de niveau* et non une *différence de nature*, entre ces technologies nouvelles et des pratiques plus anciennes. Cependant, cette différence de niveau conduit à un changement de nature dans la manière de concevoir le rôle du produit dans le système de production[7]. Le produit devient ainsi un *objet nomade de production*.

Un objet nomade de production est défini comme un objet capable de fournir des données aux clients, coopérer avec les autres entités participant à son évolution, adapter ses règles de décision en fonction de l'expérience acquise durant les phases précédentes de son cycle de production, coordonner des processus de décision et s'auto-finaliser ou imaginer.

Cette approche a été appliquée dans un projet avec un fabriquant automobile français. En effet, les technologies RFID permettent d'embarquer sur un véhicule une base de données technique qui l'accompagne tout au long de son cycle de vie, et offre des services principalement relatifs aux services d'après-vente [Bajic et Chaxel, 2002].

Des travaux plus récents [Parlikad et Mcfarlane, 2007] ont porté sur la prise de décision en « fin de vie » des produits. En effet, les fabricants sont maintenant légalement contraints de reprendre les produits usagés, dans le domaine du gros electro-ménager. Les alternatives sont le recyclage complet (destruction), la conservation de certaines pièces, ou encore le reconditionnement. Alors que les informations liées aux produits disparaissent en général dès sa vente, il a été montré qu'un système d'information lié au produit par RFID permettait de les conserver, et ainsi d'améliorer la pertinence de la décision de fin de vie.

Sahin [Sahin, 2005] évalue les retours sur investissement relatif à la précision accrue dans la gestion des stocks qui résulte de l'utilisation de RFID, en particulier dans un contexte d'utilisation du modèle du *newsboy*.

Selon [Karkkainen, 2003], la notion de produit intelligent est associée à la gestion individuelle de chaque produit au travers de son cycle de vie en intégrant le flux informationnel et matériel (physique) afin d'offrir des services en utilisant un réseau Internet.

Enfin, McFarlanne [Mcfarlane *et al.*, 2003] montre la manière dont un « produit intelligent » (un produit lié de manière permanente avec sa représentation informationnelle et capable d'être actif) influe sur le pilotage de la production, qu'il soit centralisé ou distribué.

L'utilisation des technologies RFID conduit donc à une modification des pratiques, en conférant aux produits des fonctions nouvelles. Des classifications plus générales de ces fonctions, ne dépendant pas d'une technologie particulière, ont été proposées.

Du point de vue des services offerts par l'objet dans le cadre d'une intelligence *ambiante*, on peut distinguer [Cea, 2006] :

- L'objet porteur de données,
- L'objet pointeur vers un système d'information,
- L'objet fournisseur et demandeur de services,
- L'objet sensitif enrichi avec des capteurs et des actionneurs.

[7] On peut faire ici l'analogie avec le développement d'internet. De la même manière, l'échange informatisé de données ne constitue pas en lui-même une rupture, puisque on peut le comparer avec d'autres pratiques plus anciennes, téléphonie, fax, poste, etc... qui offrent, avec de moindre performances, les mêmes fonctionnalités. Cependant, cette différence de niveau catalyse l'émergence de pratiques complètement nouvelles (différence de nature) telles que le développement collaboratif des logiciels libres.

Du point de vue des fonctions de communication et d'identification [Wong *et al.*, 2002], le produit intelligent peut :

- possèder une identification unique,
- être capable de communiquer efficacement avec son environnement,
- pouvoir mémoriser ou stocker des données au sujet de lui-même,
- disposer d'un langage de communication pour transmettre ses caractéristiques et ses besoins pendant son cycle de vie,
- être capable de participer ou de prendre des décisions appropriées à son propre destin de façon continue.

1.3.3 Une communauté de recherche émergente

L'association entre information et matière permise par les technologies infortroniques change donc profondément la manière d'envisager le pilotage de la production. En effet, l'association étroite entre la matière et l'information fait émerger un *produit holonique* [Gouyon, 2004]. La figure 1.10 est la représentation abstraite en UML de l'holon en tant qu'association d'un produit physique et d'un produit informationnel. Ce modèle initial de l'holon a été précisé et développé dans les travaux de Baïna [Baina, 2006].

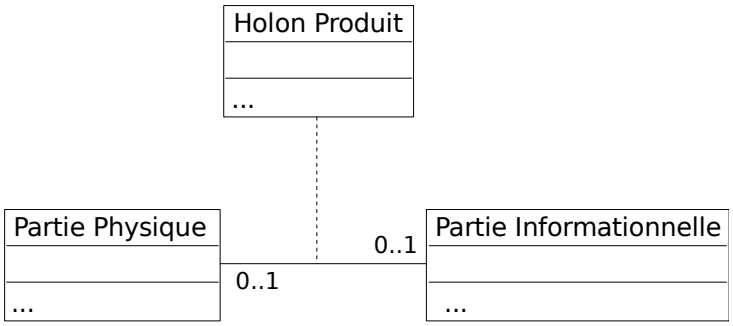


Fig. 1.10 – Modèle initial de l'holon en tant qu'agrégation physique / informationnelle

Les flux de matière et d'information, pour le moment synchronisés en un nombre fini de points (figure 1.11a), tend donc à devenir un flux de produits holoniques, à la fois matière et information (figure 1.11b).

L'émergence de ce flux de produits holoniques conduit à repenser l'organisation de la prise de décision. Ainsi, une problématique centrale est de passer d'une vision hiérarchique et intégrée du pilotage (figure 1.9)vers une vision axée sur l'interopérabilité et l'intelligence, en postulant un produit holonique vecteur de coordination et pilote des ressources (figure 1.12) [Morel *et al.*, 2007] [Pannequin et Thomas, 2006].

Une communauté de recherche est en train d'apparaître autour de ces considérations. Ainsi, au plan national, un groupe de travail *Systèmes contrôles par le produit* est en cours de création au sein du GDR MACS. Parmi les travaux récents dans ce domaine, on peut citer la thèse de Blanc [Blanc, 2006], dans laquelle est proposé un mécanisme de pilotage permettant de synchroniser les réalisations du point de vue des ordres de production. D'autre part, une architecture de pilotage basée sur des méthodes d'apprentissage est présentée dans la thèse de Bousbia [Bousbia, 2006] permettant au produit d'avoir un rôle majeur dans le pilotage de sa production.

Au plan international, le concept de contrôle par le produit est exploité par des projets

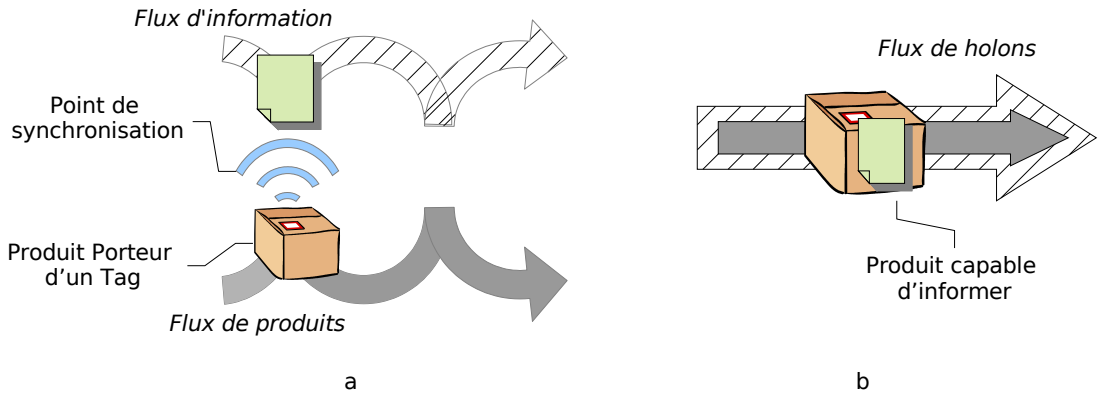


FIG. 1.11 – Synchronisation discrète entre matière et information (a), évolution vers un flux d'holon (b)

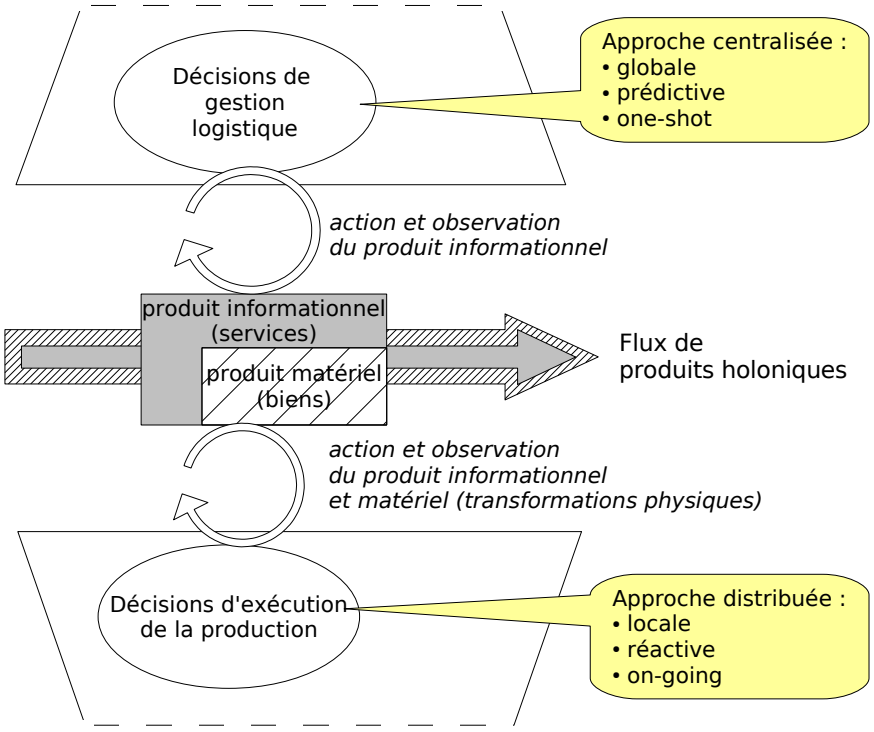


FIG. 1.12 – Architecture de décision centrée sur le produit

européens. En particulier, le projet PADABIS'PROMISE[8] étend l'idée de pilotage distribué à une architecture innovante qui incorpore ressources et produits. Le concept de contrôle par le produit y est formulé comme « L'ordre de fabrication est la fabrication ». La vision de l'architecture de pilotage de la production proposée dans ce projet est proche du paradigme présenté figure 1.12 : chaque produit serait équipé d'un agent logiciel supporté par un type nouveau d'étiquette RFID. Au niveau ERP, de nouveaux modules permettent l'accès direct de l'ERP au système de contrôle de terrain. Le niveau MES est entièrement décentralisé, les traitements du produit étant pris en charge par l'agent du produit.

Ce projet s'est ambitieusement fixé comme objectifs de développer :
- une nouvelle ontologie de système de production,
- une plateforme multi-agents temps réel,
- un nouveau type d'étiquette RFID,
- une nouvelle génération de contrôleurs programmables,
- des modules pour une nouvelle génération d'ERP.

1.3.4 Problématique : faire la preuve du concept

Le concept de pilotage par le produit, soutenu par une communauté en plein essor, s'affirme donc de plus en plus. On observe un certaine convergence des propositions autour d'architectures de référence, comme par exemple PROSA, sur lesquelles les développements récents s'appuient [Blanc, 2006].

Toutefois, assez peu de travaux portent sur la validation de ce concept, son application à des cas concrets, et sa comparaison avec des approches plus classiques. Ainsi, comme l'affirme Marik [Marik et Lazansky, 2007] dans le champ de recherche du pilotage par agents, « de nombreuses solutions expérimentales à base de systèmes multi-agents ont été proposées, développées, et – dans certains cas– mises en œuvre. Les applications industrielles à échelle réelle sont très rares, et les fonctionnalités des agents développés sont considérablement réduites. »

Parmi les divers obstacles identifiés pour expliquer cette faible adoption industrielle, les plus importants sonté :
- changer le mode de pensée lors de l'industrialisation, et de la conception du contrôle automatisé,
- supporter le processus de migration d'un contrôle centralisé vers un contrôle décentralisé,
- résoudre les problèmes techniques liés à l'interopérabilité, à la communication et à la négociation entre agents,
- rendre les systèmes multi-agents capables de travailler à l'échelle de problèmes industriels (*scalability*),
- travailler à des ontologies, structures et langages largement acceptés,
- évaluer et valider le comportement émergent d'un système multi-agents

C'est sur ce dernier point que nous nous focalisons : la problématique de cette thèse est donc la concrétisation et la validation du pilotage par le produit, et en particulier, son rôle dans l'interaction centralisé/distribué.

Notre contribution, qui est développée dans les chapitres suivants, se fait à deux niveaux :
- d'une part nous proposons d'outiller la validation et l'étude du contrôle par le produit, en proposant un environnement d'évaluation adapté (chapitre 2 et 3),
- d'autre part nous appliquons expérimentalement le contrôle par le produit à des cas d'étude, à l'aide des outils précédemment développés (chapitre 4 et 5).

[8] PABADIS based Product Oriented Manufacturing Systems for Re-Configurable Enterprises http ://www.pabadis-promise.org/

1.4 Conclusion

Nous avons énoncé dans ce premier chapitre les caractéristiques des approches centralisées et distribuées. La différenciation se fait sur la topologie du système de décision (forme pyramidale / forme plate), sur le mécanisme de coordination (hiérarchie / hétérarchie plus ou moins décentralisée), sur la manière de formuler et de résoudre le problème de pilotage (prescription par décomposition fonctionelle / émergence par approche objets) et enfin sur la dynamique (one-shot / on going).

L'approche centralisée est *de fait* relative aux systèmes d'entreprise de niveau *business* (progiciels intégrés et globaux), tandis que l'approche distribuée peut être mise en œuvre dans les systèmes de niveau *process* (systèmes « intelligents » et locaux). De par leurs différences mêmes, ces systèmes doivent coexister pour former un système de pilotage combinant leurs apports respectifs, l'efficience et l'agilité. Toutefois les solutions classiques d'intégration entre ces deux systèmes présentent une incapacité à trouver un équilibre entre les fonctions centralisées et distribuées.

Le développement des technologies infotroniques conduit à repenser la place du produit dans le système de décision, et à lui donner en particulier le rôle de vecteur de la communication entre les divers systèmes d'information d'entreprise. Nos travaux s'appuient sur ce concept, en cherchant à concrétiser et à évaluer la pertinence du pilotage par le produit pour résoudre l'interaction centralisé/distribué.

Cette étude empirique de l'impact du contrôle par le produit nécessite un environnement d'évaluation, dont le premier élément, dédié à l'émulation de systèmes opérants, est présenté au chapitre suivant.

Chapitre 2

Environnement à base de composants orientés-objets pour l'émulation de systèmes opérants contrôlés par le produit

Pour pouvoir étudier le concept de pilotage par le produit, valider l'implémentation qu'on pourrait en faire, discuter sur son applicabilité et de ses performances en regard des systèmes de pilotage existants, nous devons nous munir d'un outil d'évaluation. Ce chapitre présente la définition, le développement et le test d'un tel outil, basé sur l'émulation du système opérant, pour évaluer un système de pilotage.

2.1 Besoin d'un outil d'émulation

L'évaluation des performances d'un système de contrôle novateur est un point crucial pour son adoption ; Van Dyke Parunak a montré en 1993 avec la proposition d'un environement d'évaluation nommé MASCOT [Van Dyke Parunak, 1993], le besoin de bancs d'essais d'échelle industrielle pour combler le fossé entre la recherche et son application.

On trouve ensuite dans la littérature des articles qui relatent la comparaison de modes de contrôle antagonistes. Ainsi une coordination décentralisée « en marché » a été comparée à une coordination hiérarchique (basée sur un agent superviseur) [Cavalieri *et al.*, 2000], en utilisant le modèle de simulation d'un cas simple. Par ailleurs, des méthodes de pilotage prédictives et réactives ont aussi été comparées, à l'aide d'un dispositif expérimental modulaire[Brennan, 2000].

Mais ces premiers travaux se basaient sur des cas d'études particuliers ; les modèles d'essais sont difficilement réutilisables. De plus, les cas d'applications modélisés sont assez simples.

Sur la base de ces travaux, plusieurs développements scientifiques ont vu le jour. En premier lieu, ils ont porté sur le développement de métriques permettant de mesurer de manière homogène les performances des systèmes de contrôle distribués. En second lieu, ils portent sur le développement de dispositifs d'évaluation génériques. En particulier, le quatrième groupe d'intérêt spécifique du réseau d'excellence IMS (IMS-NoE SIG4) s'est consacré à la définition et au développement d'un service de benchmarking [Cavalieri *et al.*, 2003] [Valckenaers *et al.*, 2006a]. Ce service a pour but de fournir un environnement de modélisation ainsi qu'une bibliothèque de cas de tests accessible mondialement, par l'intermédiaire d'internet. Un tel service est hébergé, et

continue d'être développé, par l'université catholique de Louvain[9]. De plus, la problématique de la simulation des systèmes complexes a été abordée [Monch, 2006], ce qui a conduit à proposer une architecture logicielle assez similaire à celle évoquée par Brennan et les membres du SIG4.

On peut donc constater que de nombreux travaux portent sur la conception et le développement d'un dispositif d'évaluation adapté aux systèmes de contrôle distribués. On observe de plus une convergence des architectures de ces dispositifs vers une forme commune basée sur l'émulation du système opérant. Toutefois, il n'y a pas de consensus sur la manière de représenter génériquement les cas de tests émulés, ni sur la manière d'intégrer le système de contrôle à l'émulateur

Pour nous munir d'un outil d'évaluation, nous reprenons donc certains des concepts présentés dans la littérature. Cette démarche de développement nous permet aussi d'augmenter notre expertise en la matière. De plus, nous contribuons à l'obtention de modèles d'émulation génériques, en proposant une méthodologie de modélisation basée sur une vision systémique du système opérant.

La suite du chapitre situe et définit l'émulation. Ensuite le contexte de fonctionnement de l'émulateur est présenté, puis nous énoncerons notre méthodologie. Enfin, les deux dernières parties portent sur le développement d'un prototype et sur sa validation.

2.2 Évaluation par émulation

2.2.1 Méthodes d'évaluation

Évaluation abstraite

L'évaluation de la qualité de l'architecture de référence peut se faire analytiquement, l'architecture générique étant considérée *en tant que telle*, sans se fonder sur des applications concrètes. Ce type d'étude porte essentiellement sur les propriétés structurelles de l'architecture (par exemple la nature ou la quantité d'informations échangées, sa facilité de mise en œuvre, sa ré-utilisabilité ou sa capacité à évoluer) qui sont évaluées qualitativement, par comparaison avec une métrique telle que la métrique EICM (*Enterprise Integration Capability Model*). Des métriques plus spécifiquement adaptées aux technologies multi-agents sont aussi développées [Brennan et Norrie, 2003].

Évaluation empirique

Par ailleurs, l'appréciation de la qualité d'une architecture générique peut aussi résulter de *l'observation d'un ensemble d'applications concrètes* de cette architecture. Cette approche est centrée sur l'étude des propriétés opérationnelles (par exemple les taux d'utilisation des ressources, les temps de transfert des produits, etc) qui ne peuvent être observées que lors d'une application concrète.

Les propriétés générales sont obtenues par induction d'après les résultats de ces expériences particulières. La validité de ce raisonnement par induction repose sur le nombre et la représentativité des différents cas d'application testés. Pour chaque application individuelle, se pose donc le problème de l'évaluation des performances. On peut réaliser les expériences de deux manières différentes.

D'une part, elles peuvent être réalisées en déployant le système de contrôle sur un système physique, par exemple une plate-forme d'essais de laboratoire, ou bien un véritable atelier. Mais

[9] http ://www.mech.kuleuven.be/benchmarking/

cette approche pose par exemple certains problèmes :
- la disponibilité d'un atelier industriel est limitée et coûteuse,
- les plate-formes d'essais peuvent être trop simples,
- la construction et la maintenance de tels systèmes représentent un important coût matériel et humain.

D'autre part, les expériences peuvent être réalisées sur un *modèle*. À cette fin, on peut soit utiliser des méthodes analytiques, par exemple la théorie des files d'attentes, soit des méthodes de Monte Carlo[10], en utilisant des simulations.

Expérimentation par simulation

Pour l'étude du pilotage par le produit, dans laquelle l'échelle et la complexité fait partie intégrante du problème, les méthodes basées sur la simulation sont préférables aux méthodes analytiques. En effet, les méthodes analytiques, qui reposent sur la résolution d'équations représentant le système, sont limitées par la complexité de celui-ci. Utiliser une méthode analytique nous contraindrait donc à faire de nombreuses hypothèses simplificatrices ou bien à nous limiter à des systèmes de très petite taille. La simulation nous permet de représenter de manière réaliste les cas industriels que nous traitons. De plus, l'utilisation de modèles virtuels permet de multiplier les situations expérimentales, à un coût raisonnable.

Les techniques de simulation pourront donc être utilisées selon deux axes :
- D'une part, elle peuvent servir à mesurer les performances de multiples architectures de pilotage, toutes appliquées à un même cas, que l'on juge représentatif, et qui sert donc de banc d'essais (*benchmark*). Il est alors possible de comparer une architecture nouvelle à une architecture existante, ce qui permet de quantifier l'apport de cette nouvelle architecture (figure 2.1a).
- D'autre part, on peut mesurer les performances d'une architecture de contrôle dans de multiples situations d'application, ce qui permet de faire apparaître le champ d'application de cette architecture. Il est alors possible de quantifier les performances de l'architecture, en fonction du type de cas d'application (figure 2.1b).

Il nous faut séparer d'une part le cas d'application, correspondant en fait à un atelier, c'est à dire un système opérant, et d'autre part le système de pilotage, ou système de contrôle.

2.2.2 Définition de l'émulation

Il convient avant d'aller plus loin d'expliciter le concept d'émulation. Nous cherchons en particulier à distinguer émulation et simulation.

Ce concept n'est pas neuf ; dans le domaine du génie automatique, l'émulation de parties opératives permet de valider les programmes des automates programmables industriels avant leur implantation sur site. L'émulation représente donc une assistance à l'automaticien, qui pourra ainsi effectuer des tests alors que le système physique n'est pas encore réalisé, ou bien sans qu'il soit nécessaire de l'immobiliser pour les tests. Ce champ de recherche a été exploré au CRAN [Corbier, 1989] [Tixador, 1989], et a conduit à la spécification de l'outil Control Build, commercialisé actuellement par la société TNI. L'émulation est aussi utilisée dans d'autres domaines : par exemple, en électronique, l'émulation d'un microprocesseur permettra d'exécuter des programmes conçus pour ce composant, sans pour autant en disposer physiquement. Dans ce cas

[10] On appelle méthode de Monte-Carlo toute méthode visant à calculer une valeur numérique, et utilisant des procédés aléatoires, c'est-à-dire des techniques probabilistes. Le nom de ces méthodes fait allusion aux jeux de hasard pratiqués à Monte-Carlo.

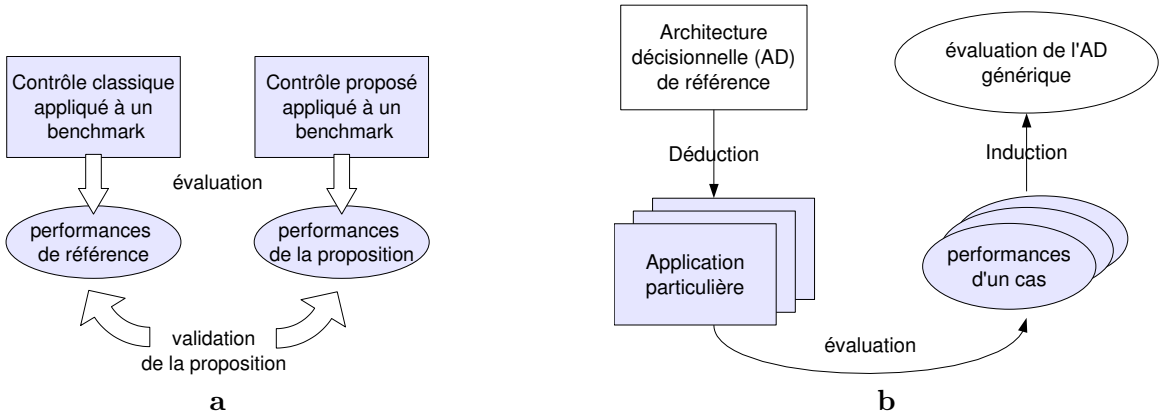


FIG. 2.1 – Deux utilisations de la simulation : (a) Validation de la l'architecture décisionnelle proposée par comparaison sur un benchmark. (b) Qualité d'une architecture décisionnelle générique, évaluée grâce à de multiples simulations

de figure, la validation n'est pas généralement l'objectif de l'émulation.

L'émulation se base donc sur un modèle de la réalité, que l'on anime. Par conséquent, simulation et émulation ont en commun les outils utilisés pour construire et pour animer les modèles, puisqu'il s'agit dans tous les cas de « singer » un comportement. On ne peut donc pas discriminer ces deux approches d'après les outils employés, ce qui contribue à une certaine confusion.

Le concept d'émulation doit donc être défini par rapport à sa finalité : *il s'agit de reproduire l'interaction du système à représenter avec son environnement. Le modèle d'émulation doit reproduire la réponse du système réel à des séquences d'entrées, afin d'être utilisé dans un système plus vaste.* En comparaison, la finalité de la simulation est d'observer l'évolution des états internes du modèle placé dans une situation prédéfinie.

2.2.3 Tentative de formalisation

L'évaluation est réalisée en mettant en contact d'une part le système de contrôle proposé, et d'autre part le système opérant sur lequel le contrôle doit s'appliquer. Si SO est le système opérant, SC le système de contrôle, et P les performances attendues, on peut écrire, en s'inspirant du prédicat de l'automatisation de Fusaoka [Fusaoka *et al.*, 1983][11].

$$SO_{connu} \wedge SC_{inconnu} \supset P_{connu}$$

L'expression de SO et P formalise donc un *besoin*. Par rapport à ce besoin, SC est une solution. L'expérimentation, c'est à dire l'exécution de $SO \wedge SC$ permet de valider cette solution.

Expérimenter par émulation revient à remplacer le système opérant réel par un modèle exécutable SO_m : le modèle d'émulation. Il est aussi possible de remplacer le système de contrôle par un modèle SC_m, par exemple un ensemble de règles de décision (figure 2.2).

Dans le domaine du développement de systèmes mécatroniques [Bishop, 2005], trois types de simulation ont été définis, dépendant de la nature réelle ou émulée du système de contrôle et du système opérant.

[11] Ce prédicat présente un principe bien connu, applicable pour tous les domaines de l'automatique. Il est particulièrement intéressant dans le domaine de l'ingénierie formelle de l'automatisation, où il peut être utilisé pour synthétiser la commande [Pétin *et al.*, 2006]. Dans notre cas, il sert lutôt à énoncer de manière claire et concise le principe de base de notre outil.

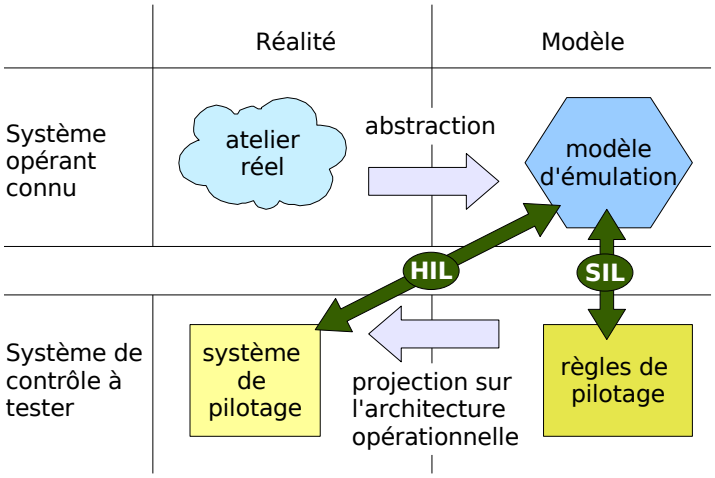


FIG. 2.2 – Système opérant et système de contrôle virtuels ou réels, et les approches de validation SIL (*software in the loop*) et HIL (*hardware in the loop*)

1. un modèle du système de contrôle peut être appliqué à un système opérant réel, mais plus simple que le système final. Ce mode de simulation permet le « prototypage du système de contrôle ».
2. un système de contrôle simulé peut être appliqué au véritable système opérant final. C'est la simulation *hardware in the loop* (HIL).
3. Le système opérant virtuel peut être opéré par le système de contrôle simulé. Ce dernier mode de simulation est désigné *software in the loop* (SIL).

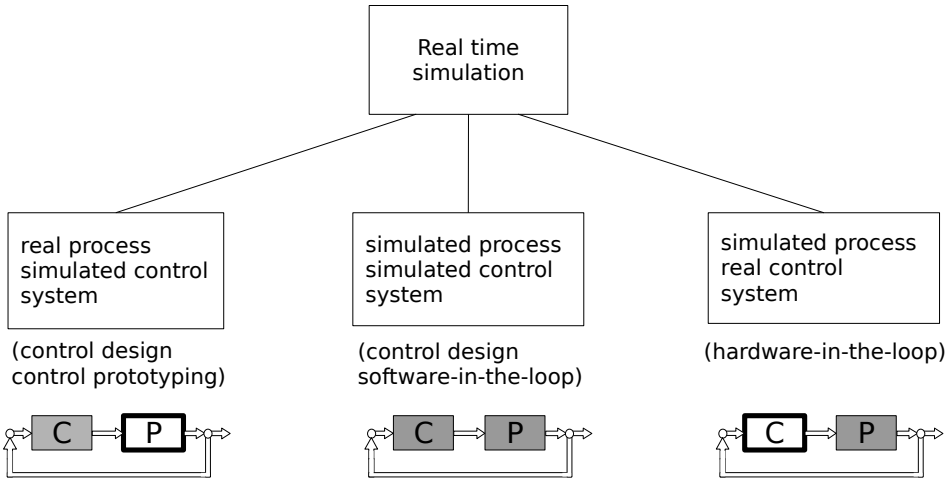


FIG. 2.3 – Classification des approches de simulation temps-réel, dans le domaine de la mécatronique [Bishop, 2005]

Dans le domaine du développement et de la validation des systèmes de pilotage de la production, les quatre modes d'expérimentations possibles (correspondant aux combinaisons possibles entre réalité et simulation) sont définis par [Pfeiffer *et al.*, 2003] :

1. la manière *traditionnelle*, dans laquelle système de contrôle et système opérant sont réels,

2. *l'émulation*, dans laquelle le système de contrôle réel est appliqué à un modèle du système opérant,
3. *reality-in-the-loop*, où un modèle du système de contrôle est appliqué au système opérant réel,
4. la *simulation hors-ligne*, dans laquelle le système de contrôle et de système opérant sont des modèles.

Nous choisissons pour notre part de parler d'émulation à partir de l'instant où le système opérant est mis sous la forme d'un modèle exécutable qui peut interagir avec un système de contrôle externe. Nous devrons donc considérer deux cas de figure, selon que le système de contrôle est modélisé ou réel. Ces deux situations sont dénommées par analogie avec les notations de la mécatronique :

SIL : $SO_m \wedge SC_m \supset P$; les règles de contrôle (prototype du système de contrôle) sont appliquées au modèle d'émulation. Cette première étape d'évaluation est focalisée sur les algorithmes de prise de décision. Cette situation expérimentale est proche de la pratique traditionnelle de simulation, mais insiste nettement sur la séparation entre modèle de contrôle et modèle du système opérant.

HIL : $SO_m \wedge SC \supset P$; le véritable système de contrôle est appliqué au modèle d'émulation. À la différence de la mécatronique, il est difficile de trouver une distinction nette entre un modèle informatique du système de contrôle (*software*) et le système de contrôle réel (*hardware*). En effet, le système de contrôle définitif est essentiellement de nature logicielle. Cependant, en déployant les éléments logiciels définitifs de l'architecture de contrôle, l'expérimentation est focalisée sur des problématiques proches du matériel, par exemple liées à la synchronisation des deux flux d'exécution parallèles, ou à des échanges de messages sur un réseau de bande-passante limitée, etc...

Nous utiliserons donc dans la suite de ce mémoire ces deux termes (HIL et SIL) pour désigner les deux approches d'expérimentation basées sur une émulation du système opérant.

2.2.4 Principe de modélisation

L'hypothèse fondamentale de notre démarche est donc d'affirmer que *si* les expériences effectuées sur le modèle d'émulation sont concluantes, *alors* elles le seraient dans la réalité. (On peut la formuler $(SO_m \wedge SC \supset P) \Rightarrow (SO \wedge SC \supset P)$). En bref, que le modèle est une représentation valable de la réalité.

Néanmoins, cette hypothèse peut être remise en cause, car la modélisation déforme la réalité :
- en agrégeant certains aspects (choix d'un niveau de granularité)
- en faisant abstraction de certains phénomènes (par exemple, il est courant de ne pas considérer l'approvisionnement en énergie des machines modélisées)
- en introduisant la subjectivité du modélisateur dans le modèle.

Le modèle est donc en quelque sorte une projection de la réalité sur un espace plus restreint, lié au point de vue utilisé pour modéliser.

Les méthodologies de simulation stipulent toutes une étape de validation du modèle, mais cette validation permet de répondre à la question « ce qui est représenté dans le modèle est-il correctement représenté ? » mais pas « les aspects importants de la réalité sont-ils représentés dans le modèle ? ».

Par exemple, il est courant pour les praticiens de la simulation de faire abstraction des problèmes de routage. En effet, les entités représentant les produits peuvent comporter un attribut

indiquant la route qu'elles doivent suivre. La route prescrite est alors nécessairement suivie, sauf erreur de programmation. Un tel modèle est naturellement plus simple, mais il ne permet pas d'étudier les problématiques liées au routage.

À cause de ce *biais* pouvant être introduit dans la création du modèle de validation, nous nous posons la question du caractère probant des expériences. Ce doute est particulièrement vif si le même acteur est à l'origine d'une proposition de contrôle et du modèle utilisé pour la valider. En effet, dans cette situation, il est à la fois celui qui formule un problème (le modèle d'émulation, et les performances attendues) et propose sa solution (le système de contrôle).

Il est donc important de spécifier un principe de modélisation permettant d'assurer que le modèle est une représentation de la réalité conduisant à des résultats expérimentaux probants.

Ce principe repose d'abord sur la séparation et la modularité entre le système à valider et le modèle de validation. Dans la mesure du possible, ces deux éléments doivent être créés séparément. de plus, le modèle de validation doit être élaboré en cherchant à *ne pas faire abstraction des aspects qui posent problème dans la réalité.*

Incidemment, l'application de ce principe nous donne une raison supplémentaire justifiant le choix de l'émulation par rapport à la simulation. En effet, de nombreux simulateurs (par exemple Arena) sont conçus de telle manière que les flux de natures différentes sont confondus (figure 2.4). Ainsi, une entité Arena représente à la fois de la matière (un produit), mais aussi un événement, puisque l'arrivée de l'entité sur un bloc déclenche un traitement. De plus, l'entité peut être porteuse d'information, comme dans l'exemple du routage, cité précédemment. Puisque le pilotage par le produit influe directement sur la synchronisation des différents flux, nous ne pouvons faire abstraction du fait qu'ils sont effectivement séparés dans la réalité. L'émulation permet de mieux représenter cette séparation.

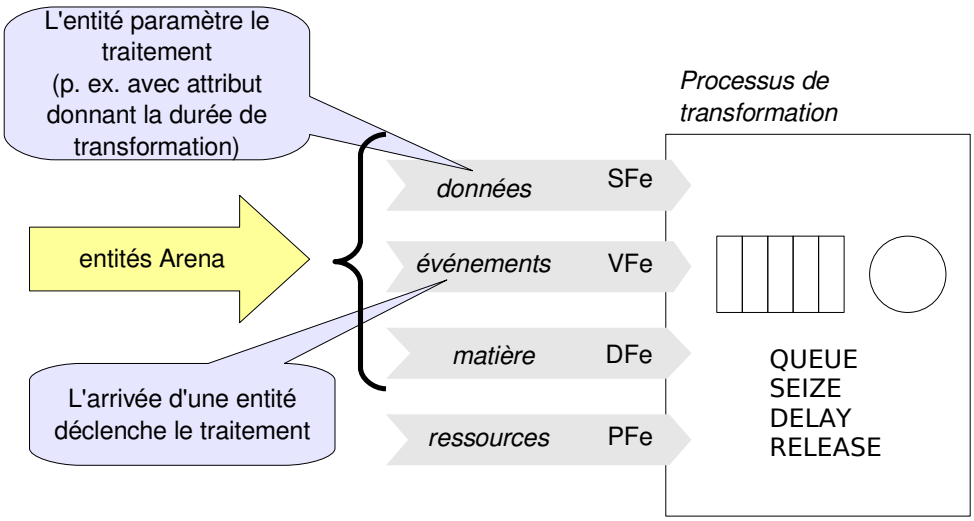


Fig. 2.4 – Confusion des flux de natures différentes sous Arena

Cette séparation des flux n'est qu'un aspect parmi d'autres ; le pilotage par le produit nécessite aussi de tenir compte de toutes les possibilités d'évolution des produits dans le modèle d'émulation. Cela inclut aussi bien le routage du produit que la flexibilité de son élaboration (gammes alternatives). La granularité avec laquelle le système opérant est modélisé doit elle aussi être adaptée au pilotage par le produit.

Pour répondre de manière générique à ces problèmes de modélisation, nous proposons une méthode de modélisation pour l'émulation, et un prototype qui la met en œuvre.

2.2.5 Démarche de développement

La démarche que nous avons adoptée pour définir et développer cette méthode et ce prototype s'appuie sur une vue systémique du système attendu. Ainsi, on distingue les points de vue fonctionnels (ce que le système fait), structurels (quelles entités le compose) et dynamiques (comment évolue le système dans le temps). D'autre part, la démarche repose sur les raffinements successifs des modèles. Nous commençons par l'étude des relations entre le système et son environnement, en nous focalisant sur les fonctions principales devant être assurées par le système. Ensuite, le système est défini par des modèles logiques. Ces derniers permettent enfin, d'obtenir des modèles d'implémentation, spécifiques au logiciel de simulation à événements discrets utilisé (figure 2.5). Pour chaque niveau, les aspects fonctionnels, structurels et dynamiques sont examinés.

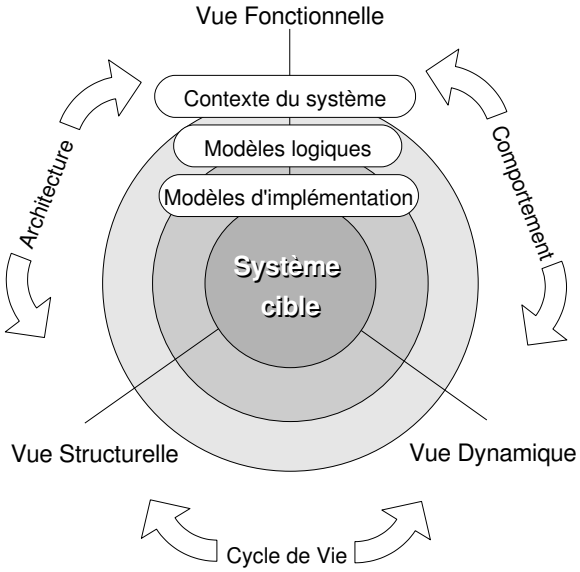


FIG. 2.5 – Vues fonctionnelle, structurelle et dynamique sur le système attendu [Morel, 2006]

La suite de ce chapitre est donc structurée de manière correspondante. La première section place l'outil d'évaluation dans son contexte, et énonce les fonctions principales qu'il doit remplir. La seconde introduit les primitives de modélisation, basées sur une décomposition systémique du système opérant. Ensuite, la troisième partie présente les modules d'émulation développés. Enfin, nous vérifions le bon fonctionnement de notre outil, en l'appliquant à la modélisation de différents cas.

2.3 Méthode de modélisation pour l'émulation

2.3.1 Contexte de l'émulation

Point de vue fonctionnel

Nous distinguons trois acteurs principaux dans l'environnement de l'émulateur. D'abord, le concepteur du système de contrôle, qui utilise l'environnement d'expérimentation pour évaluer un système de contrôle qu'il propose. La finalité de l'émulateur est de répondre à ce besoin.

Pour réaliser cette finalité d'évaluation, l'émulateur échange des flux d'informations (commandes et états) avec le système de contrôle. Enfin, pour représenter l'évolution de l'état du système opérant par rapport aux commandes du système de contrôle, le modèle doit prendre en compte les contraintes physiques qui dans la réalité influent sur l'évolution de l'état de l'atelier. Un troisième acteur est donc un expert du système à modéliser. Les relations du système avec les acteurs de son environnement sont représenté figure 2.6.

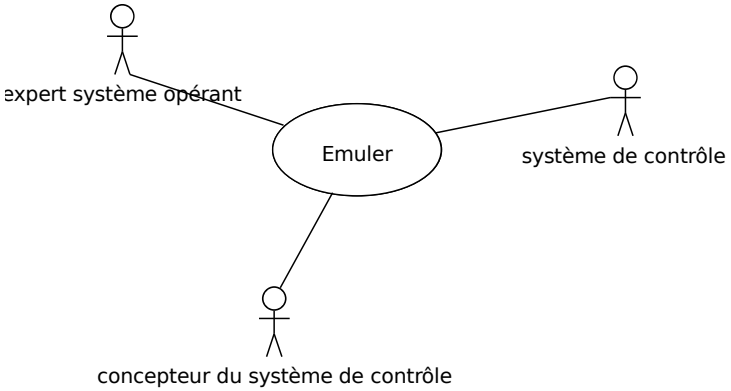


FIG. 2.6 – Le système d'émulation dans son environnement (Diagramme UML de cas d'utilisation)

La fonction globale d'émulation peut donc être affinée en diverses sous-fonctions, correspondant aux relations avec chaque acteur. Ainsi, on distingue les sous-fonctions de modélisation, d'exécution et d'expérimentation (figure 2.7).

Modélisation : L'expert du système opérant crée le modèle d'émulation. Cette modélisation comporte deux étapes :

- la construction de la structure du modèle. Elle comporte deux types d'éléments : les flux de produits et les équipements agissant sur ces flux.
- le paramétrage du modèle, d'après des données provenant du terrain (par exemple les temps de cycle, de changement de série, etc...).

Conformément à la démarche d'émulation, il s'agit de modéliser les éléments existant dans le système opérant, ainsi que les règles physiques qui contraignent leur fonctionnement. Par conséquent, le modèle d'émulation ne comprend aucun élément du système de contrôle. La création du modèle d'émulation s'avère donc plus simple que lors d'une démarche de simulation. En effet, la représentation de la décision dans les modèles de simulation est souvent délicate [van der Zee, 2006].

Exécution du modèle : Après la phase de modélisation, le modèle d'émulation est exécuté. Cette fonction correspond à :

- la réception d'ordres provenant du système de contrôle,
- l'application des ordres reçus dans le modèle et le calcul de l'évolution des états physiques qui en résulte,
- l'émission de messages d'état vers le système de contrôle. les messages émis peuvent correspondre à l'état des flux de produits, ou bien à celui des équipements.

Expérimentation : L'expérimentation comporte deux étapes :
- la définition des indicateurs de performance, et d'un scénario d'expérimentation, qui a lieu pendant la phase de modélisation,
- le mesure effective des indicateurs définis, et la mise en œuvre du scénario, en phase d'exécution.

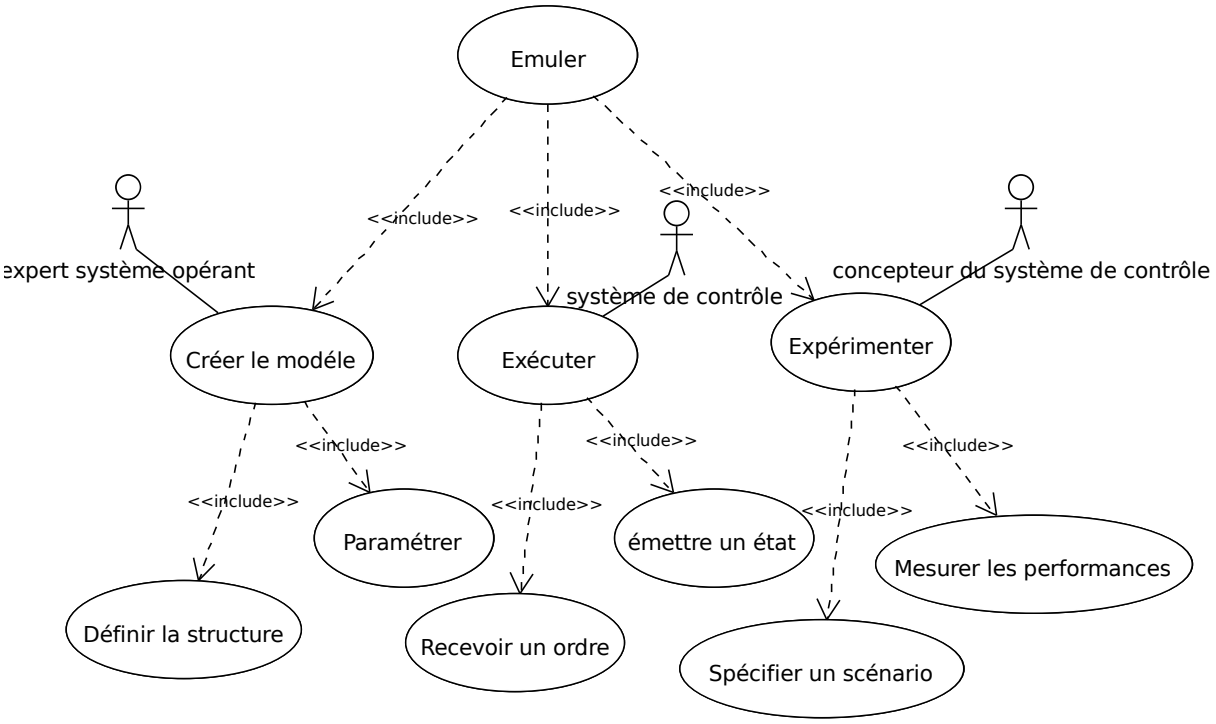


FIG. 2.7 – Détail des fonctions de l'émulateur (Diagramme UML de cas d'utilisation)

Point de vue dynamique

On peut donc définir, d'un point de vue très macroscopique, deux étapes du cycle de vie du modèle d'émulation :
- la phase de modélisation, incluant la création du modèle d'émulation, son paramétrage , et la validation du modèle obtenu ;
- la phase d'exécution, pendant laquelle le modèle d'émulation est en interaction avec le système de contrôle, et les performances mesurées.

Point de vue structurel

On isole les substantifs dans les énoncés qui précédent ; ils correspondent aux objets du domaine d'intérêt.
- des équipements
- des flux de produits
- des messages correspondant soit à des ordres, soit à des état
- une structuration des éléments du modèle
- des indicateurs de performance

– des paramètres, correspondant d'un équipement du système opérant
– un ou plusieurs scénarios, constitués essentiellement d'un jeu de paramètres et d'indicateurs de performance

Ces objets sont structurés de la manière indiquée dans le diagramme présenté en figure 2.8.

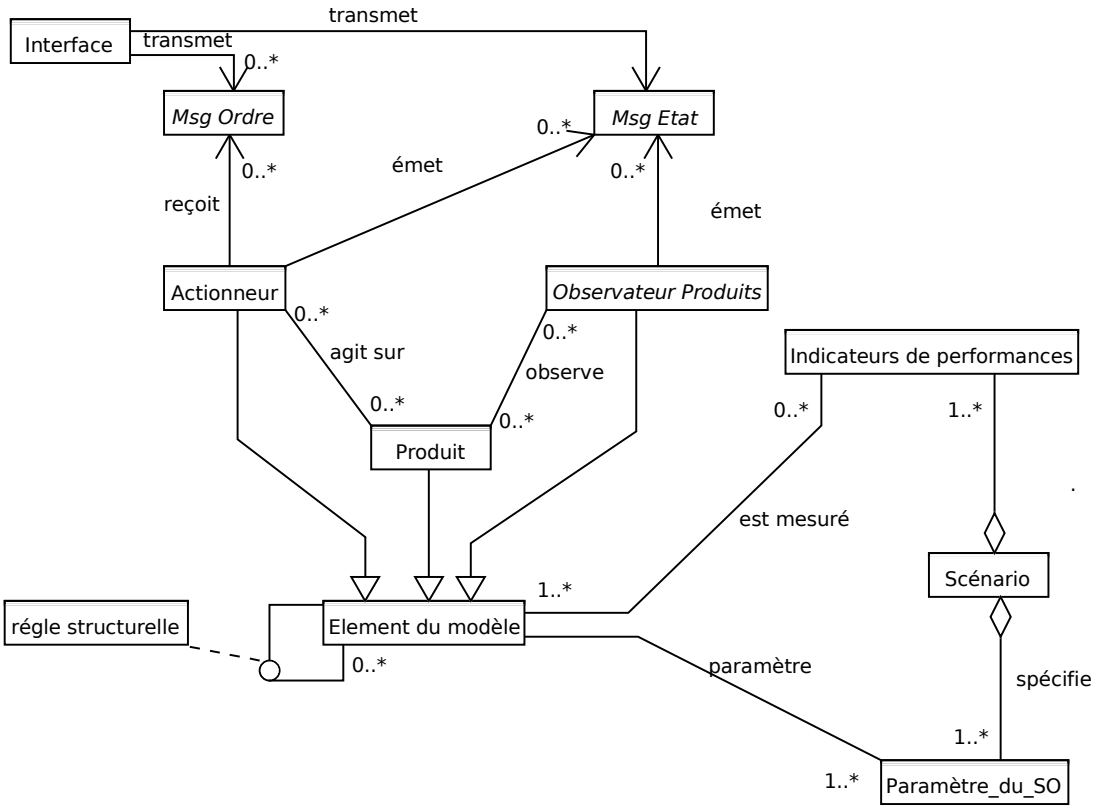


FIG. 2.8 – Modèle du domaine d'intérêt (Diagramme UML de classes, seuls les noms des classes sont présentés)

2.3.2 Modèles logiques : méthodologie de création de modèle d'émulation

Détermination de primitives d'émulation

On appellera *primitives d'émulation* les briques de base qui nous permettront de créer des modèles d'émulation. Déterminer ces primitives revient donc à choisir :

– la granularité de la représentation,
– les aspects de la réalité qui seront modélisés ;

Les travaux du SIG4 de l'IMS-NoE ont conduit à définir des primitives de modélisation inspirées de composants physiques :

– station de travail, se déclinant en station d'assemblage, de chargement/déchargement, d'usinage,
– zones de stockage, comme des tampons d'entrée ou de sortie,
– équipements de transport, comme des convoyeurs ou des véhicules autonomes.

De même, l'outil de simulation MAST[12] comporte lui aussi trois types d'éléments devant être représentés :

- des cellules de production, jouant le rôle d'une machine d'assemblage, ou d'usinage, mais ayant aussi une importance dans le routage des pièces.
- des convoyeurs, assurant le transport des pièces
- des aiguillages (diverter), assurant le routage entre différents convoyeurs.

Cet ensemble a été élargi, dans le cadre d'une application à la cellule expérimentale du centre pour l'automatique et le contrôle distribué de Cambridge : des éléments permettant de représenter les lecteurs RFID ont été introduits, et d'autres éléments génériques ont été modifiés pour correspondre aux spécificités du cas.

D'autres émulateurs et simulateurs adoptent une approche basée sur la représentation en trois dimensions des différents éléments modélisés (produits et actionneurs). Cette approche, pouvant se révéler indispensable lorsqu'il s'agit d'évaluer la trajectoire d'un robot ou l'ergonomie d'un poste de travail, n'apporte que peu d'avantage pour l'étude du pilotage. En revanche, elle rend la modélisation beaucoup plus difficile, puisque la constitution géométrique des équipements émulés doit être explicitement programmée.

On peut donc constater que la modélisation est fondée sur une représentation plus ou moins abstraite des équipements de l'atelier ; d'autre part, la granularité de la modélisation est le plus souvent la cellule de production.

L'application au contexte du pilotage par le produit de notre principe de modélisation spécifiant que le modèle d'émulation ne doit pas faire abstraction des aspects à étudier : l'émulateur doit séparer les flux de natures différentes, avoir une granularité à l'échelle du produit, et enfin admettre toutes les évolutions du produit (déplacements et opérations) possibles physiquement.

S'agissant de la granularité, elle doit permettre de représenter les opérations subies par un produit individuel. Il est donc exclu de piloter une ressource en lui soumettant des ordres de fabrication propre à un lot (typiquement : « produire telle quantité de tel type de produit »). La production de chaque produit doit donc être explicitement commandée.

Les autres contraintes, ainsi que le besoin de définir des primitives de modélisation génériques, conduisent à baser la représentation de l'atelier sur les évolutions logistiques des produits. L'application des principes de la systémique nous guide dans cette définition.

2.3.3 Vision systémique du système opérant

Nous utilisons la systémique pour développer nos primitives d'émulation. En effet, on peut caractériser l'état de chaque produit par sa morphologie, et ses coordonnées spatiales et temporelles [Le Moigne, 1977] [Feliot, 1997] [Penalva, 1997].

Chaque produit évolue donc continûment dans l'espace et le temps, en changeant de forme. Pour pouvoir représenter plus facilement ces évolutions, nous postulons que :

- le produit évolue dans un espace d'états discrets ;
- les évolutions morphologiques et spatiales sont indépendantes,

Le premier postulat affirme que l'on peut déterminer deux ensembles finis E et F correspondants respectivement aux états spatiaux et morphologiques des produits. L'état de tout produit pourra être représenté par un couple $(e, f) \in E \times F$ (figure 2.9).

Le second postulat affirme que toute évolution d'un produit $((e, f) \mapsto (e', f'))$ pourra être représentée comme une suite de transformations morphologiques à espace constant $((e, f) \mapsto (e, f'))$, et de transformations spatiales, à forme constante $((e', f) \mapsto (e', f))$.

[12] MAST (Multi-Agents Simulation Tool) est un outil de simulation de systèmes de production développé par Rockwell Automation, qui utilise des technologies de simulation à base d'agents.

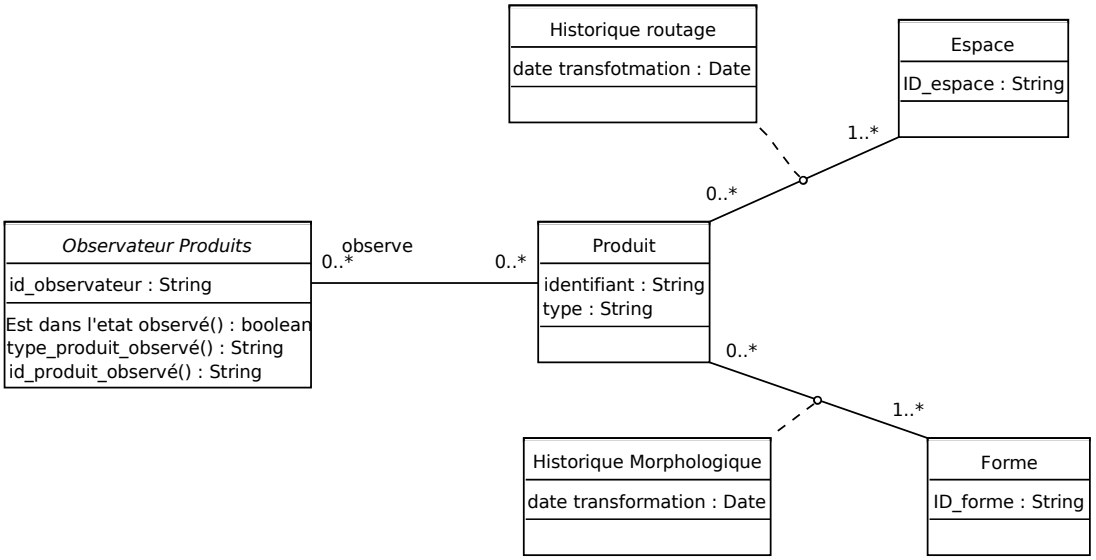


FIG. 2.9 – Représentation des évolutions spatiales et morphologiques des produits (Diagramme UML de classes)

2.3.4 Modélisation des équipements

Nous définissons les primitives d'émulation permettant de modéliser les équipements par rapport à ces transformations qui agissent sur les flux de produits. Deux types de transformateurs sont ainsi créés :

- les transformateurs morphologiques, $(e, f, t) \mapsto (e, f', t')$, qui agissent sur la forme des produits sans changer notablement sa position ;
- les transformateurs spatiaux, $(e, f, t) \mapsto (e', f, t')$, qui agissent sur la position des produits sans changer leur forme.

Les transformations ont une certaine durée $\Delta t = t - t'$. Ces durées sont données lors du paramétrage des transformateurs, afin de reproduire au mieux la dynamique du système modélisé. Il nous faut aussi représenter les transformations de temps à espace et forme constants, qui correspondent aux attentes des produits.

Les transformateurs de forme et d'espace sont contrôlables : il est possible de choisir la position ou la forme résultante de la transformation, ainsi que de déclencher leur action. Par contre, les transformations de temps ne sont pas directement contrôlables : elles dépendent du fonctionnement des autres transformations. Ainsi, c'est une transformation d'espace (déplacement hors d'un stock) qui met fin à l'attente d'un produit.

Le diagramme de classe UML de la figure 2.10 présente les primitives utilisées pour modéliser les équipements. La figure 2.11 présente quant à elle la manière de les paramétrer.

Au niveau de granularité d'un atelier, un centre d'usinage ou une cabine de peinture seraient par exemple modélisés par des transformateurs morphologiques. Par ailleurs, un chariot ou un aiguillage entre convoyeurs serait modélisé par des transformateurs spatiaux.

2.3.5 Observation des produits

Puisque l'on se place dans le contexte des systèmes pilotés par le produit, il est particulièrement important de représenter les moyens d'observation des flux de produits.

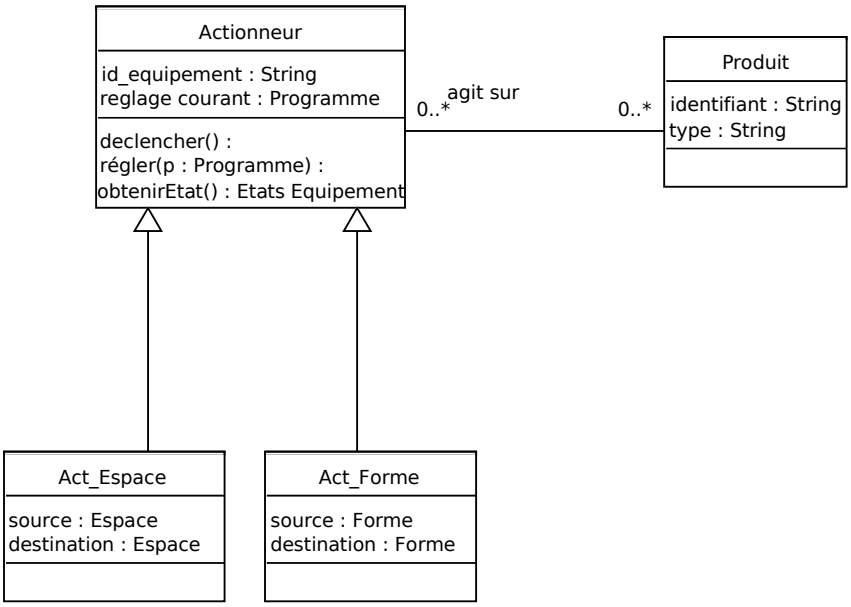


FIG. 2.10 – Les actions des équipements sur les produits sont représentées par des primitives Actionneurs, qui se déclinent en actionneurs spaciaux ou morphologiques.

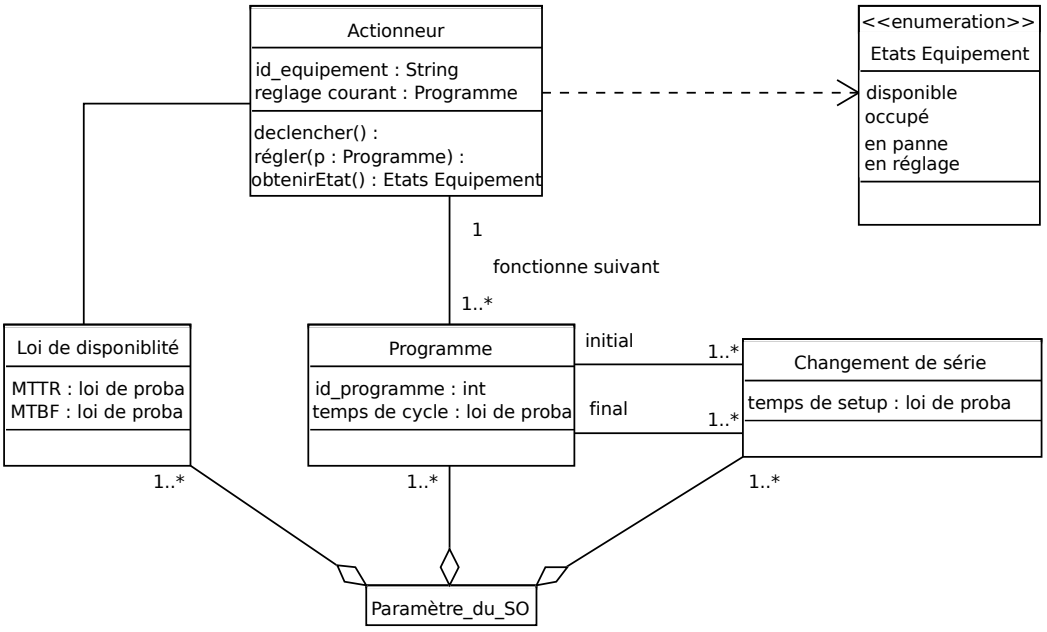


FIG. 2.11 – Paramétrage des actionneurs. Les actions pouvant être effectuées sont représentées par des programmes. On peut aussi modéliser la disponibilité de l'équipement, et prendre en compte les temps de changement de série.

Ces observateurs peuvent être classifiés de deux manières. D'une part on distinguera les observateurs de forme et les observateurs d'espace. L'observation de forme est en général plus complexe que l'observation d'espace. Elle peut être automatisée grâce à des systèmes de vision ou de palpage permettant de percevoir effectivement la morphologie d'un objet. L'observation d'espace se base sur des systèmes de détection de présence, ou de triangulation (par exemple, avec un GPS d'atelier). Accessoirement, on peut remarquer que l'observation de la forme sous-entend souvent une observation d'espace (puisqu'on connaît la position de l'observateur de forme).

D'autre part on pourra classer les primitives d'observation du produit en fonction de la précision avec laquelle le produit observé est identifié. Dans le cas le plus simple, un capteur renvoit une information de présence. Dans certains cas, il est possible de connaître le type du produit (utilisation classique des codes à barres). Enfin, les techniques les plus récentes permettent d'identifier individuellement le produit observé (code à barres, RFID, gps d'atelier, etc...).

Cette catégorisation est illustrée par la figure 2.12. Dans les développements que nous présentons, nous nous restreignons aux observations d'espace, en particulier, ceux qui permettent d'observer chaque instance de produit.

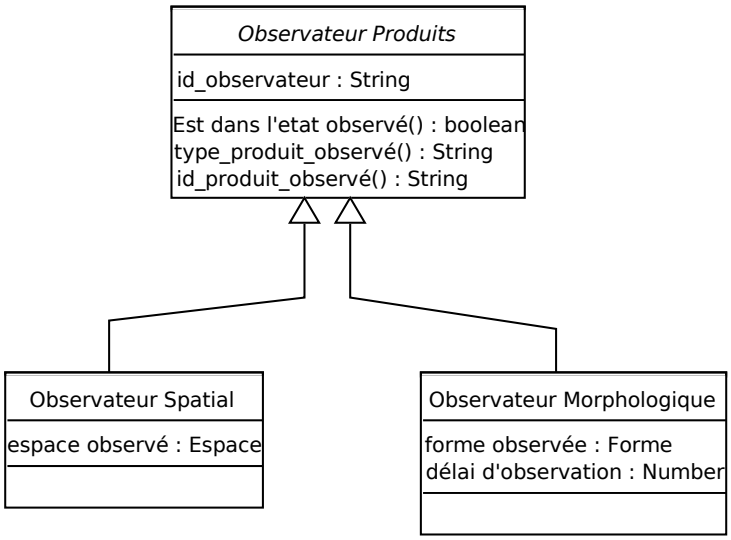


Fig. 2.12 – Les différentes manières d'observer le produit, dans son positionnement spatial, ou sa morphologie.

2.3.6 Étapes de la modélisation

On peut donc énoncer les étapes de la création d'un modèle d'émulation. Il s'agit d'abord d'identifier les différents états morphologiques et spatiaux des produits. Ces états peuvent être représentés à la manière d'un automate à état fini, comme on le voit dans [Gouyon, 2004].

Ensuite, nous associons les équipements de l'atelier aux changements d'états morphologiques ou spatiaux des produits. Le modèle d'émulation comprendra donc les états stables du produit, représentés par des transformations de temps, ainsi que les primitives de transformation de forme ou d'espace, qui s'intercaleront entre ces états stables pour représenter les changements d'état. Enfin, nous inclurons dans le modèle d'émulation les équipements d'observation présents dans l'atelier.

Enfin, la phase de modélisation sera clôturée par le paramétrage des primitives de transformation, en terme de lois de disponibilité, de temps de cycle, et de temps de changement de série. Il sera aussi nécessaire de définir les indicateurs de performances devant être mesurés.

En conclusion de cette phase de modélisation logique, on peut constater que l'émulateur comporte les éléments permettant de modéliser les actionneurs, la physique du procédé du point de vue du produit et les observateurs des produits. L'émulateur est donc en quelque sorte une demi-boucle cybernétique, sur laquelle pourra se connecter le système de contrôle (figure 2.13).

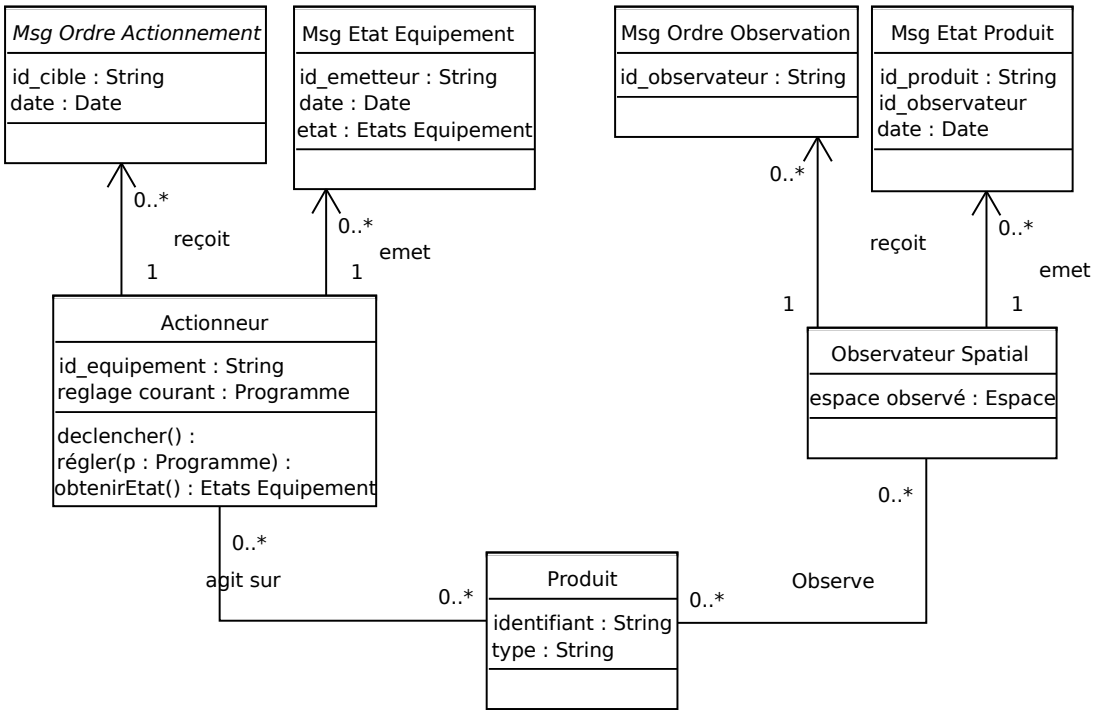


FIG. 2.13 – Diagramme de classe : L'émulateur est basé sur la représentation de l'évolution des produits, en tenant compte des capacités d'actionnement et d'observation de l'atelier modélisé.

2.4 Modèles d'implémentation : développement d'un prototype

2.4.1 Environnement de développement

Le logiciel Arena a été choisi pour programmer le prototype mettant en œuvre notre méthodologie de modélisation. Arena est un logiciel de simulation à événements discrets développé par la société Rockwell software. C'est un logiciel mature, qui est assez bien implanté dans le monde industriel et scientifique. La base historique d'Arena est le logiciel Siman. Siman est encore utilisé aujourd'hui comme moteur de simulation, mais il a été intégré dans une interface graphique conviviale, permettant la création visuelle des modèles de simulation, ainsi que leur animation. Ces caractéristiques rendent aisée la prise en main d'Arena : aucune connaissance en programmation n'est requise pour le développement d'un modèle.

Le choix d'Arena comme environnement de développement a donc été en partie influencé par cette facilité d'accès, ainsi que par l'expérience que nous avions de ce logiciel. On peut aussi noter que les offres concurrentes sont assez similaires à Arena, et que –à notre connaissance– il n'existe pas de logiciel open-source offrant le même niveau de fonctionnalité. De plus, Arena

permet (dans la version professionnelle) de développer ses propres modules de simulation. Enfin, il est possible d'interagir avec le logiciel par l'intermédiaire d'une API relativement complète.

Quelques inconvénients subsistent toutefois, en particulier les difficultés d'adaptation causées par la non-disponibilité du code-source du logiciel, ou les incompatibilités entre les différentes versions du logiciel. Arena a cependant les qualités requises pour développer un prototype, en utilisant cette capacité à développer nos propres modules de simulation.

2.4.2 Modules de simulation

Chaque primitive de modélisation a donc été développée sous la forme d'un module de simulation. Ces modules ont été groupés dans une bibliothèque de modules spécifiques. De plus, pour donner à ces modules des capacités d'interaction, une bibliothèque de fonctions (DLL[13]) leur a été adjointe. Celle-ci constitue une interface vers ces modules, et comporte aussi des fonctions permettant le contrôle à distance (en ligne) de l'émulateur.

La structuration des modules les uns par rapport aux autres est présentée figure 2.14, tandis que la liste des modules est donnée table 2.1.

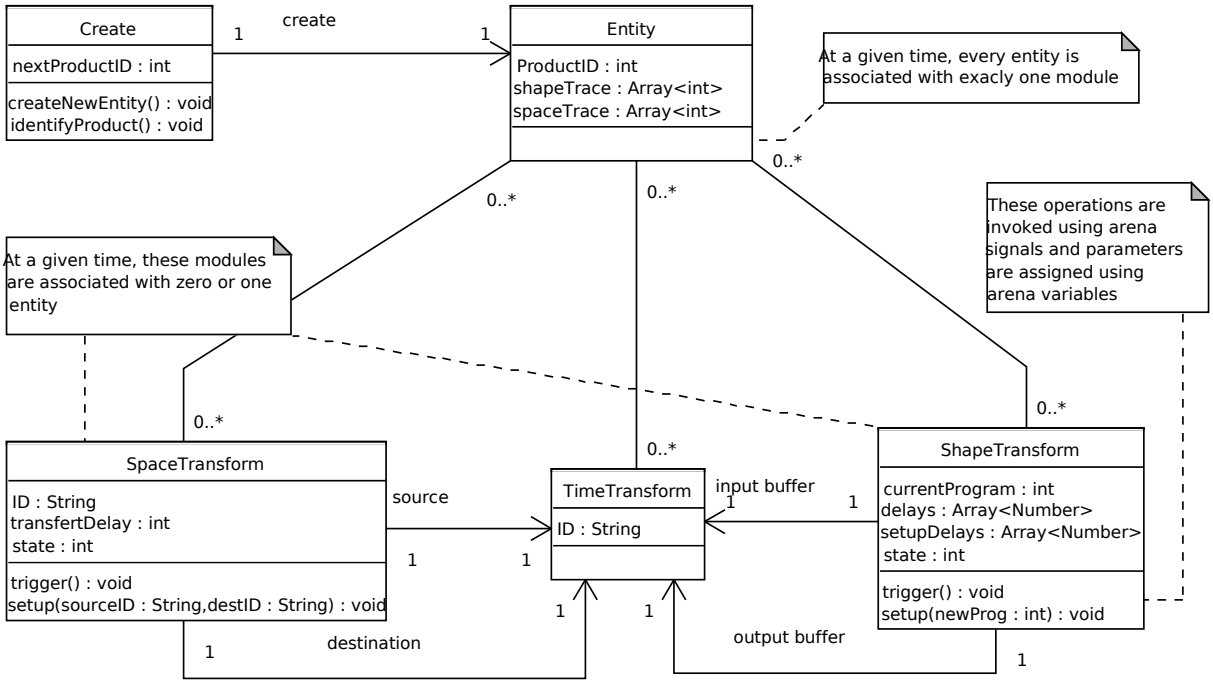
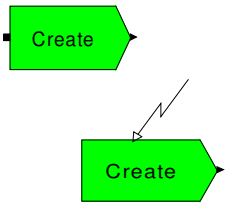
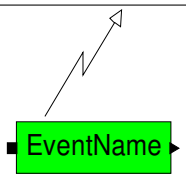
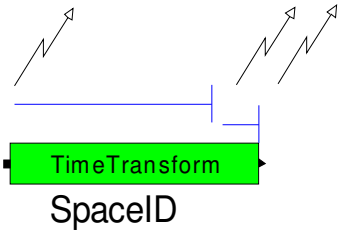
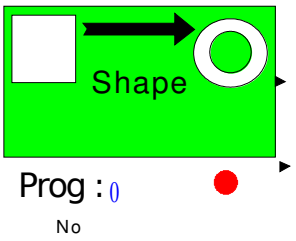
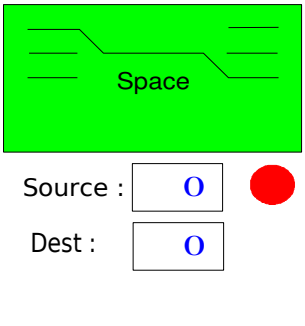
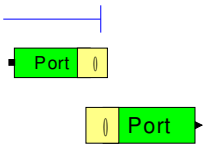


Fig. 2.14 – Structuration des modules d'émulation.

2.4.3 Contrôle à distance de l'émulateur

Pour qu'un système de contrôle externe puisse interagir avec l'émulation du système opérant (mode *hardware in the loop*), ce dernier doit pouvoir recevoir des commandes, et renvoyer les événements de l'émulation.

[13]DLL *Dynamically Linked Library* (Bibliothèque liée dynamiquement) : bibliothèque de fonctions chargées dynamiquement à l'exécution. Une bibliothèque logicielle se distingue d'un exécutable dans la mesure où elle est utilisée par des programmes plutôt que d'être exécutée directement elle-même ; elle fournit un code « assistant » un programme indépendant en lui fournissant des services

Création des Produits Ce module introduit dans le modèle des entités Arena représentant les produits, en leur associant un identifiant unique. Il fonctionne selon deux modalités : soit en créant des entités ; soit en traitant les entités qui le traversent.	
Événements-produits Ce module déclenche un événement lorsqu'un produit le traverse. Les paramètres de cet événement sont l'identifiant du produit déclencheur, l'identifiant de l'observateur (i.e. du module) et la date d'observation.	
Transformation de temps Cette transformation représente un état stable dans lequel le produit attend, sans changement notable de forme ou de position. Ce module incorpore des événements-produits, déclenchés par l'entrée ou la sortie du module ; un troisième événement est déclenché lorsqu'un produit est prêt à être libéré.	
Transformation de forme Ces modules permettent de représenter les évolutions de la morphologie des produits (usinage, peinture, etc...). Chaque transformation pouvant être appliquée est désignée par un entier. La transformation est paramétrée pour s'alimenter dans une transformation de temps. Elle répond à des demandes de début de cycle et de réglage, les temps de réglage pouvant être dépendants de la séquence.	
Transformation d'espace Ce module représente un changement de position notable du produit, par exemple le passage d'un stock à un autre. Ainsi, il libère un produit en attente dans un module de transformation de temps, et après un délai de traitement, l'envoi vers un autre module de transformation de temps. Comme le module de transformation de forme, il répond aux commandes de début de cycle, ainsi qu'aux demandes de réglages.	
Ports Les ports peuvent se substituer aux transformations de temps pour les procédures de prise et de pose des produits.	

TAB. 2.1 – Les modules Arena composant la bibliothèque d'émulation

Le format des messages échangés est déterminé d'après l'interface des blocs à commander. La figure 2.15 présente la structure de ces messages. Celle-ci est conçue pour englober dans une même structure les différents types d'événements échangés.

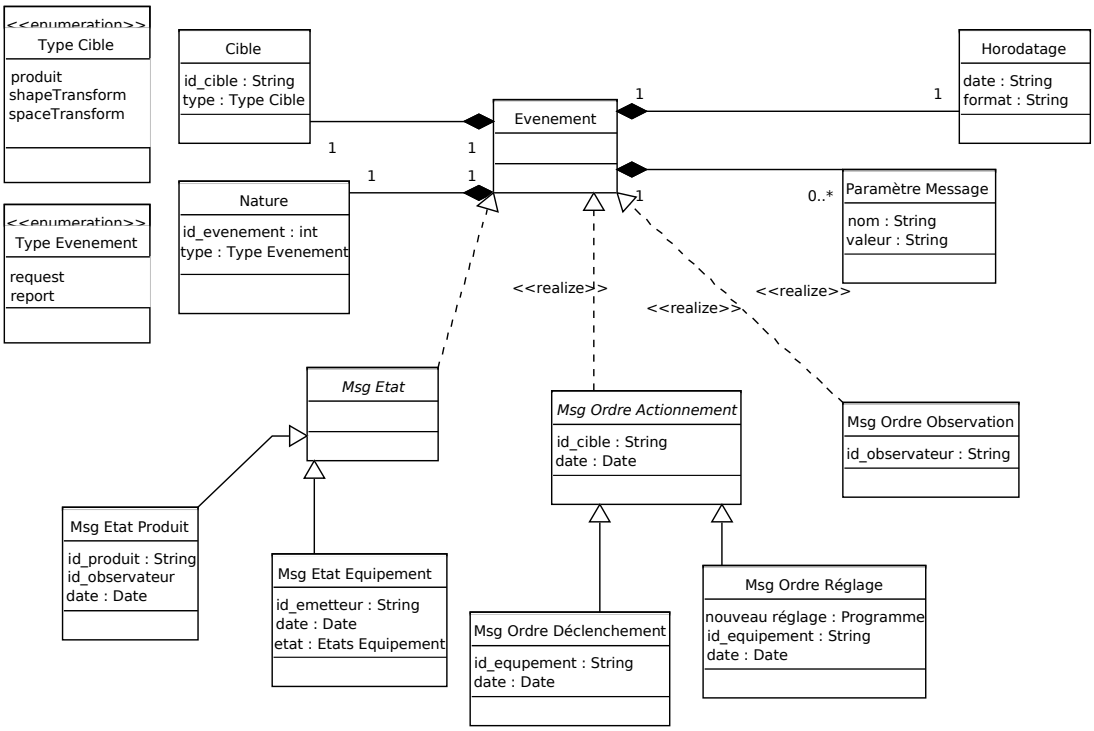


Fig. 2.15 – Messages échangés entre l'émulateur et le système de contrôle.

Le transfert de ces messages se fait grâce à un encodage en XML[14], et une transmission sur un socket TCP[15]. L'avantage de cette solution est sa simplicité. Toutefois, l'utilisation d'un standard comme par exemple SOAP[16] aurait certainement permis une meilleure gestion du transport des messages.

2.4.4 Architecture du dispositif d'émulation

L'architecture du dispositif d'émulation est basée sur les deux niveaux d'expérimentation *hardware-in-the-loop* et *software-in-the-loop* (figure 2.16). D'une part, il est possible d'intégrer le contrôle dans l'environnement de simulation, sous la forme de règles de contrôle programmées en Visual Basic (VB). Ces règles peuvent alors tirer parti de l'interface VB des modules d'émulation. De plus, puisque l'interprétation du code VB est gérée par Arena, la simulation peut s'exécuter en événements discrets.

[14]XML (*eXtended Markup Language* ou langage de balisage extensible) est un standard du World Wide Web Consortium qui sert de base pour créer des langages balisés spécialisés ; c'est un « méta langage ». Il est suffisamment général pour que les langages basés sur XML, appelés aussi dialectes XML, puissent être utilisés pour décrire toutes sortes de données et de textes.Il s'agit donc partiellement d'un format de données.

[15]une *socket* est une interface logicielle de connexion réseau permettant les communications basée sur les protocoles définis par TCP/IP. Une socket TCP/IP est caractérisée par une adresse IP identifiant un serveur, et un numéro de port TCP identifiant une application offrant un service.

[16]SOAP *Simple Object Access Protocol* (Recommandation W3C). SOAP est un protocole léger basé sur HTTP conçu pour échanger des données structurées au format XML entre deux machines distantes.

Ce mode de contrôle permet donc d'effectuer facilement des expériences dont l'exécution est rapide. Le mode SIL permet donc d'obtenir rapidement un prototype du système de contrôle.

D'autre part, il est possible de connecter l'émulateur à un système externe, par exemple un système multi-agents. Dans cette configuration, on est en HIL. Comme le système de contrôle ne partage pas l'ordonnanceur d'événements d'Arena, la simulation doit être exécutée en temps réel[17]. Par ailleurs, l'avancement du temps à événements discrets conduit à négliger les délais de prise de décision, puisque la simulation stoppe pendant la décision. Une solution basée sur l'hybridation entre temps réel et événements discrets a été proposée [Saint Germain *et al.*, 2003] : la simulation avance par événements discrets, sauf lorsque le contrôle est actif, auquel cas l'avancement est en temps réel. Une autre piste est l'utilisation de HLA[18] pour partager l'ordonnanceur d'événements de l'émulateur avec le système de contrôle externe. Une autre fonctionnalité intéressante de HLA est la possibilité de distribuer l'exécution de l'émulation. Le principal avantage que nous voyons dans cette distribution est la possibilité d'agréger en ensemble de modèles simples, créés et maintenus localement. Ces fonctionnalités de distribution de l'exécution de l'émulation ne sont donc pas actuellement requises, mais pourrai être utilisées dans des phases ultérieures du développement de l'outil d'émulation.

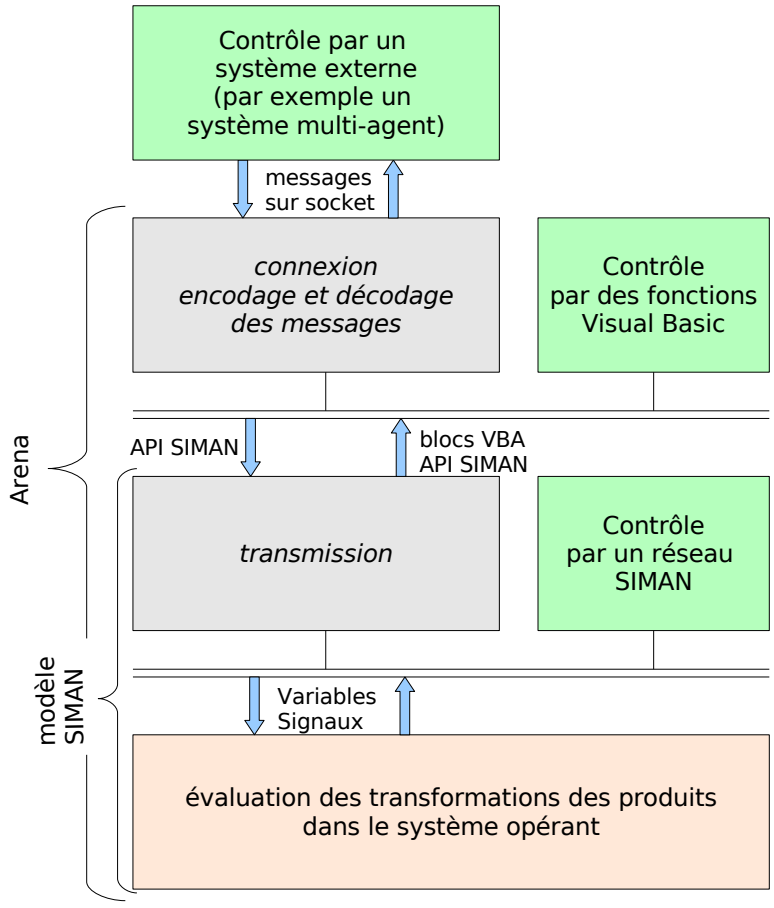


FIG. 2.16 – Schéma de l'architecture de l'émulateur

[17] C'est à dire que l'avancement du temps de la simulation se fait proportionnel au temps de l'ordinateur

[18] HLA, High Level Architecture (norme ISO 1516) permet de distribuer l'exécution d'une simulation sur plusieurs systèmes.

2.4.5 Mesure des performances opérationnelles

La mesure des indicateurs de performance est réalisée en utilisant les possibilités d'Arena, qui est conçu pour relever des mesures et faire des statistiques.

Par ailleurs, nous avons muni notre prototype du moyen de retracer les changements d'état des produits et des ressources. S'agissant des produits, la succession de transformations de forme et d'espace subies par le produit est enregistrée dans l'entité Arena le représentant. Ces traces peuvent facilement être exportées vers une base de données, bien que ce ne soit pas automatisé dans notre prototype.

L'activité des modules de transformation d'espace/temps et de forme/temps peut elle aussi être tracée. Chaque module génère des enregistrements à chaque changement d'état. Un script permettant de compiler ces traces individuelles sous la forme d'un diagramme de gantt a été développé (figure 2.17). Il est donc facile d'automatiser la création d'un tel diagramme à la fin de chaque simulation.

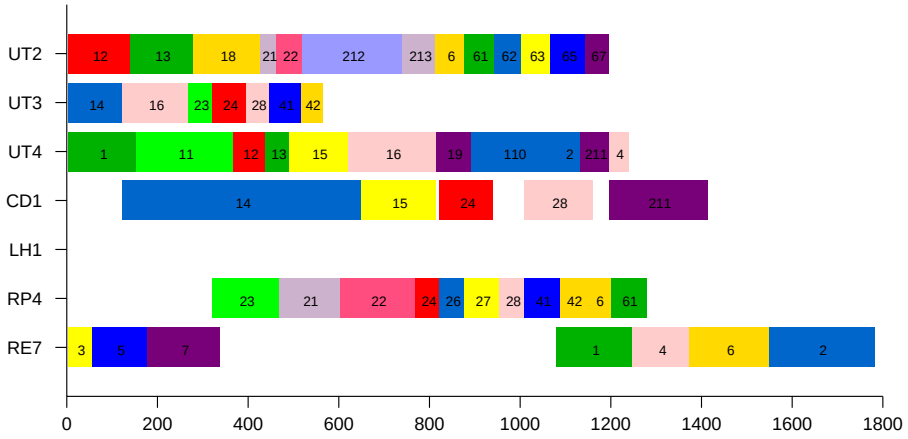


FIG. 2.17 – Exemple d'une trace d'une simulation, représentée sous la forme d'un diagramme de gantt.

2.5 Vérification du bon fonctionnement de l'outil

Dans cette section, nous cherchons à déterminer si l'infrastructure d'émulation que nous avons développée répond aux spécifications. Dans un premier temps, chaque module de simulation sera testé de manière unitaire. Les modèles utilisés sont très simples et ne correspondent pas à des cas réel. Ces tests unitaires servent d'abord à vérifier que le module a bien le comportement attendu, et ensuite qu'il s'interface correctement avec un système externe.

Ensuite, nous appliquerons notre méthodologie de modélisation à des cas réels, ce qui permet de valider l'expressivité des primitives d'émulation.

2.5.1 Tests unitaires

Validation manuelle

Nous illustrons dans cette partie notre méthode de test par le cas du module de transformation de forme, qui est d'ailleurs le plus compliqué parmi les modules développés.

Nous assemblons un modèle simple, dont la particularité est de comporter une interface utilisateur permettant de contrôler *manuellement* le module lors de l'exécution de l'émulation (figure 2.18).

Le modèle est centré autour du module d'émulation à tester. On y a adjoint d'autres modules, permettant de générer des produits et de les mettre en attente devant le module de transformation de forme.

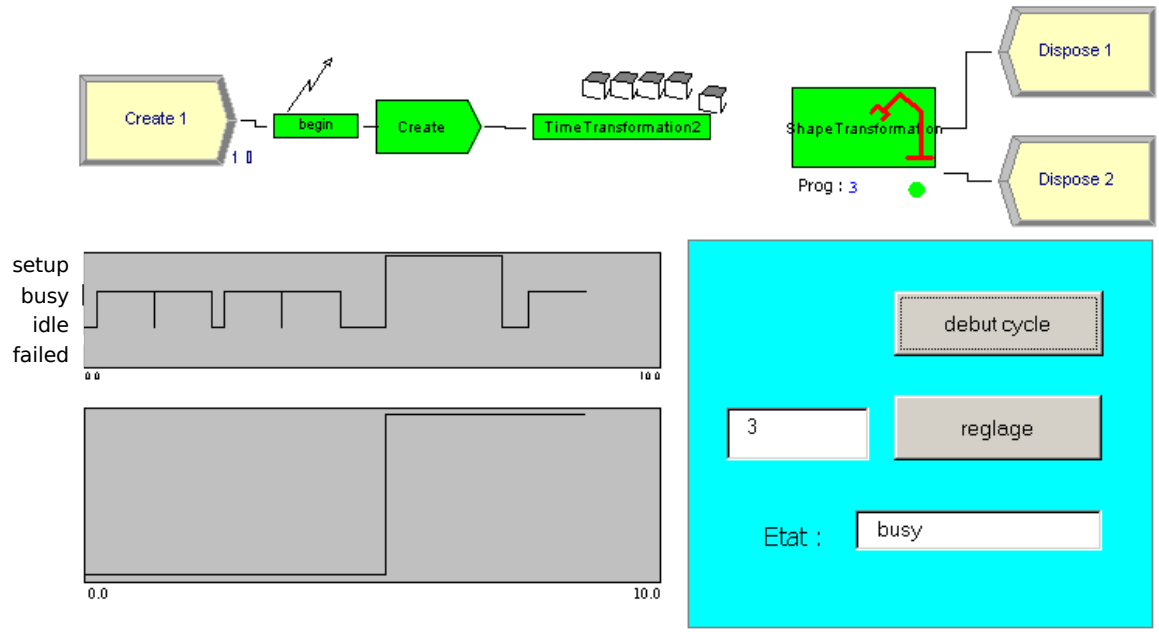


FIG. 2.18 – Le modèle Arena de test unitaire d'un modèle de transformation de forme. On distingue l'interface de pilotage du module (en bas à droite) et des graphiques représentant l'état de l'équipement émulé en fonction du temps (en bas à gauche).

Dans ce même modèle de test, nous avons ajouté des indicateurs graphiques permettant de suivre les changements d'état de l'équipement émulé. Ceux-ci correspondent donc à ce qui se passe *physiquement* dans l'atelier.

Enfin, nous avons muni le modèle d'une interface homme-machine, reflétant l'interface du module de simulation (déclenchement, réglage, obtention de l'état courant). Pour que le modèle puisse interagir avec un acteur externe, la simulation s'exécute en mode temps-réel, c'est à dire que l'écoulement du temps dans la simulation est proportionnel à l'écoulement du temps dans la réalité.

Ce modèle permet facilement de *faire fonctionner* le module. Il est donc utile en phase de développement, pour valider de manière expérimentale et informelle les nouvelles fonctionnalités. Il permet par exemple de valider le fait qu'un seul signal de début de cycle est mémorisé par le module. Par contre il ne permet pas d'appliquer rapidement des scénarios de test prédéfinis.

Tests automatisés

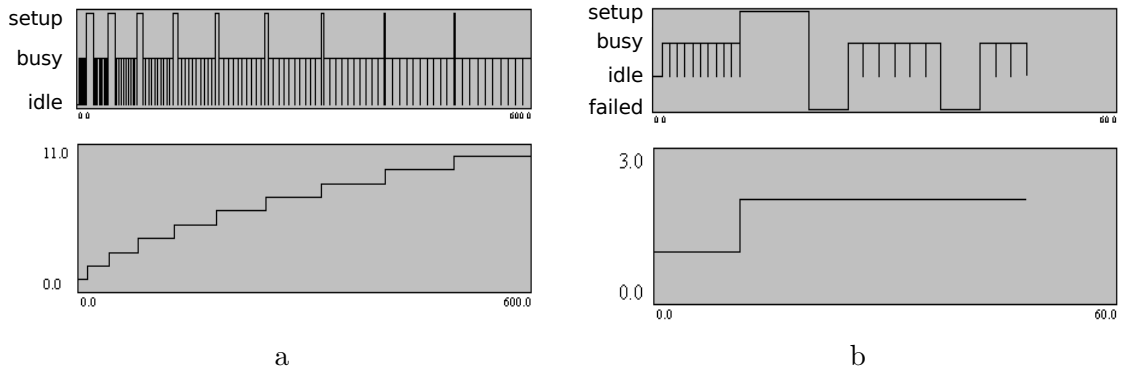
Afin d'accélérer ces tests, nous avons développé une logique de contrôle simple, se substituant à l'utilisateur dans les tests précédents. Ce système de contrôle très simple est embarqué dans l'environnement de simulation, sous la forme de programmes VB ; ainsi, on peut exécuter le modèle en mode événements discrets. La fonction de contrôle est activée par les événements,

correspondants à la présence d'un produit en attente de transformation, et aux changements d'état de l'équipement.

Nous avons paramétré le module de manière fictive : dix programmes ont été définis (de P1 à P10) ; leurs temps de cycle sont égaux à leurs numéros (1 minute pour P1, 2 minutes pour P2, etc...). Les temps de changement de série ont aussi été définis : 9 minutes pour le passage de P1 à P2, 8 minutes de P2 vers P3, etc...

Dans un premier temps, la logique de contrôle est conçue de manière à exécuter dix cycles pour chaque programme, dans l'ordre croissant. La figure 2.19a présente la trace de ce scénario.

Un second scénario permet de valider la prise en compte des pannes : on ajoute une panne avec un temps de bon fonctionnement de 11 minutes, et une durée de 5 minutes. La figure 2.19b présente la trace de la simulation.



a b

FIG. 2.19 – Résultat de l'exécution du premier scénario de test (a), et du second scénario (b). Les graphes représentent en fonction du temps l'état l'équipement émulé (en haut) ainsi que le numéro du programme exécuté.

Ces tests automatisés nous ont été utiles comme tests de non-régression, c'est à dire pour vérifier qu'une amélioration du module (ajout d'une fonctionnalité, correction d'un bogue) ne cause pas d'autres problèmes. De plus, ce test nous permet de valider qu'il était possible de contrôler le module sous forme d'un programme embarqué dans l'environnement de simulation, en mode événements discrets.

Test du pilotage à distance

Enfin, en reprenant le modèle de test manuel, nous pouvons valider l'interface de pilotage à distance du module. Ces tests permettent de vérifier l'encodage et le décodage correct des événements de la simulation.

Une fois la connexion effectuée, on peut envoyer des chaînes XML[19]. Nous avons donc envoyé manuellement des commandes au serveur d'émulation, et pu vérifier que leurs exécutions étaient correctes. De même, on a pu vérifier *de visu* que les états émis par le serveur d'émulation étaient corrects.

2.5.2 Tests d'intégration

Après ces tests unitaires, nous avons tenté d'appliquer notre méthode de modélisation à des cas réels. Ceux-ci sont basés sur les partenariats présents ou passés du laboratoire avec des

[19]Nous avons pour cela utilisé l'utilitaire réseau GNU netcat (http ://netcat.sourceforge.net

industriels. Ces modélisations constituent des tests d'intégration pour notre prototype, mais ils nous permettent aussi d'évaluer l'expressivité de notre méthode de modélisation.

Cas d'un fabricant de meuble

Une présentation de ce cas peut être trouvée dans [Klein et Thomas, 2006]. L'atelier considéré est dédié à la fabrication de meubles en kit. La matière première est constituée par des grands panneaux de particules, des sachets de quincaillerie, des notices de montage, et du matériel d'emballage (carton, film plastique, blocs de polystyrène expansé). Les fournisseurs sont en majorité des entreprises appartenant au même groupe, ou des ateliers voisins. Les produits finis sont des colis, qui sont transmis à un autre atelier, chargé du stockage et de l'expédition vers les clients.

Les panneaux de particules sont d'abord débités en pièces dimensionnées. Ensuite, certaines pièces sont percées, et si besoin usinées. Enfin, l'emballage des différentes pièces constituant un colis (y compris la quincaillerie) est réalisé.

Le déplacement des pièces de bois dans l'atelier se fait par lots, sous la forme d'un « pont de pièces », c'est à dire une pile de panneaux posés sur un panneau martyr. Le déplacement et le stockage de ces ponts de pièces se fait sur des voies de rouleaux. Ces voies ont pour particularité leur capacité limitée, et un fonctionnement strictement FIFO. La figure 2.20 [Klein et Thomas, 2006] présente les flux de matière dans cet atelier.

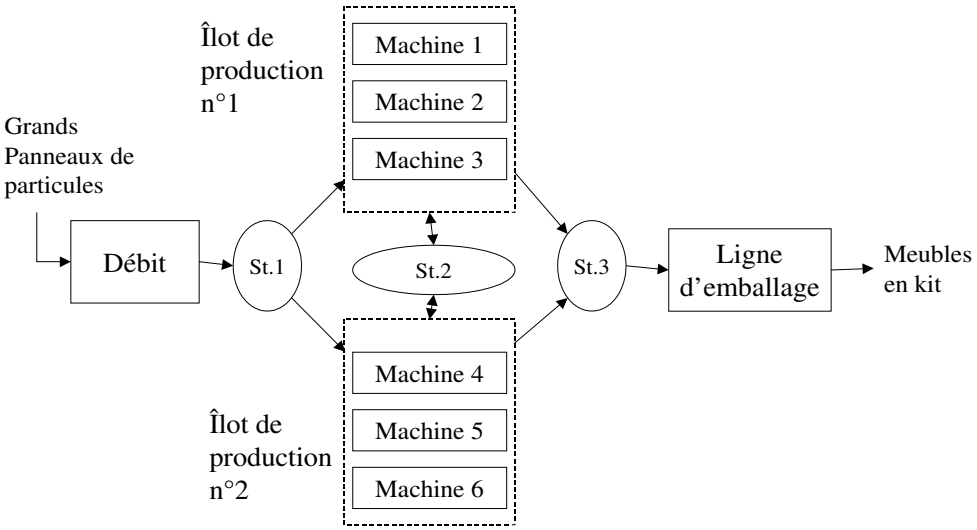


FIG. 2.20 – Application de la méthode d'émulation : cas d'un atelier de fabrication de meubles en kit

La modélisation a été réalisée de la manière suivante : Chaque voie de rouleaux a été assimilée à une transformation d'espace. Les transbordeurs, permettant le transfert d'un pont de pièces à un autre ont été modélisés par des transformations d'espace. Enfin, les machines de perçage et d'usinage ont été représentées par des transformations de forme. Les produits n'ont pas été modélisés à l'échelle de la pièce, mais à l'échelle du pont de pièces.

Cette modélisation peut être comparée à un modèle de l'atelier qui avait été réalisé d'une manière plus classique. Dans ce modèle, les produits sont porteurs de leurs gammes, ainsi que des temps de cycle correspondants à leurs passage sur chaque machine. Ainsi, dès qu'une opération est terminée, le produit est routé vers la machine suivante. Celle-ci traite les entités qui arrivent

au plus tôt. Les attributs des entités circulant dans le modèle sont initialisés au début de la simulation d'après les informations contenues dans un fichier excel. Le pilotage des produits dans l'atelier simulé est donc prescrit par ces informations.

On constate donc que l'utilisation de notre méthode de modélisation offre comme avantage la possibilité d'adjoindre facilement au modèle d'émulation un système de pilotage pouvant interagir avec les équipements virtuels tout au long de l'exécution du modèle. Ainsi, l'impact des règles de routage et de pilotage peut être étudié, ce qui n'était pas possible dans le modèle classique. Une comparaison des deux modèles montre aussi que notre prototype simplifie et clarifie la modélisation, facilitant du même coup maintenance et ré-utilisation du modèle.

Cette application met cependant en exergue certaines limites de notre méthode, et surtout de notre prototype. En premier lieu se pose le problème de la modélisation de l'emballage. En effet, cet aspect n'a pas été étudié, ce qui restreint fortement le champ d'application de notre méthode. Cet aspect pourra toutefois être ajouté dans la suite ; un module spécifique à un modèle peut aussi être utilisé.

Ensuite, on peut remarquer que notre module de transformation de forme ne tient pas compte de la distance entre l'espace source et destination pour évaluer le délai de transfert. Ce n'est pas très grave ici, puisque ces délais sont faibles (ils étaient d'ailleurs négligés dans le modèle classique), mais pourraient l'être si on modélise une chaîne logistique.

Une autre limite de ces modules de transformation de forme est la gestion des erreurs. En effet, que se passe-t-il lorsqu'un déplacement impossible est demandé (vers une voie de rouleaux pleine, ou suivant un chemin qui n'existe pas) ? Dans le prototype actuel, Arena génère une erreur dans cette situation, mais il faudrait gérer ces erreurs par des échanges avec le système de contrôle.

Enfin, la gestion des programmes des ressources a posé elle aussi problème. En effet, le nombre de programmes pouvant être exécutés est énorme (plusieurs milliers), et change régulièrement. Il sera donc nécessaire de munir nos modules d'un moyen simple d'automatiser leur paramétrage. Ce problème de gestion de l'information se pose cependant dans la réalité, il n'est donc pas anormal qu'il soit présent dans le modèle d'émulation.

Cas d'un atelier organisé en flux tiré (DFT)

Ce cas a été présenté dans [El Haouzi *et al.*, 2007]. Sa particularité principale est son pilotage, basé sur la DFT (Demand Flow Technology). Selon cette méthode de conception des lignes de production, les opérateurs doivent être relativement polyvalents. Ainsi, lorsqu'un opérateur est inactif sur son poste de travail, il doit se déplacer sur le poste en amont ou en aval, en fonction de ses compétences. Cette mobilité des opérateurs produit un équilibrage de la ligne, et une augmentation substantielle de son débit. Toutefois, la mobilité pose un problème de modélisation. En effet, la date de fin d'une opération en cours, qui avait été programmée lorsque l'opération avait été démarrée, peut être remise en question à tout moment par l'arrivée d'un nouvel opérateur.

Des modules de simulation spécifiques ont été développés pour tenir compte de cette mobilité des ressources. Cependant, notre prototype ne permet pas de gérer cette mobilité des opérateurs. Nous sommes donc limités aux situations dans lesquelles l'affectation des ressources est fixée. Un futur développement des modules d'émulation serait donc la gestion de la mobilité des ressources.

Cas d'un équipementier automobile

L'origine de ce cas, est une étude ayant été présentée dans [Pannequin et Thomas, 2004]. Les données industrielles recueillies à cette époque ne correspondent donc certainement plus à l'usine actuelle ; elle restent cependant réalistes. Une dizaine de milliers de produits, d'une centaine de références différentes, sortent quotidiennement de l'usine considérée.

Le site de production est composé d'une dizaine d'ateliers, de structure semblable, indépendants et dédiés à la production d'une famille de produits (une famille est d'abord constituée à l'aide de critères commerciaux (même constructeur automobile), mais les critères techniques en découlent). Nous avons modélisé l'un de ces ateliers.

La production dans chaque atelier se fait en deux étapes :

- D'abord, un produit semi-fini est fabriqué. L'assemblage final est ensuite réalisé à partir de ces produits semi-finis.
- Une cellule est dédiée à la fabrication des produits semi-finis, tandis que l'assemblage peut se faire sur trois cellules différentes.

L'atelier modélisé comporte donc quatre cellules indépendantes, ainsi qu'un stock de produits semi-finis délimitant deux boucles de production (figure 2.21).

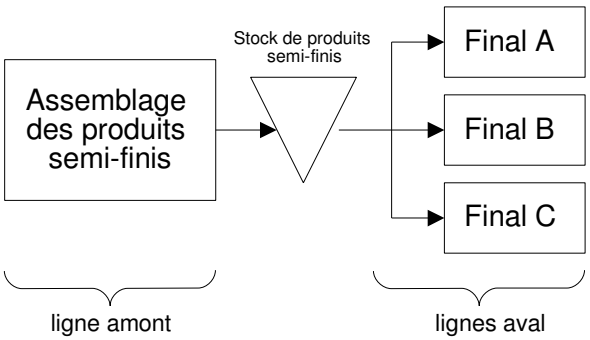


FIG. 2.21 – Application de la méthode d'émulation : cas d'un équipementier de l'industrie automobile

Cet atelier a été modélisé en représentant chaque cellule par une transformation de forme. Le stock de produits semi-finis utilise des transformations de temps, des modules de transformation d'espace permettant de modéliser l'entrée et la sortie de ce stock. Enfin, des observateurs du flux de produits ont été ajoutés pour suivre l'état des produits émulés.

Comme pour le cas du fabriquant de meubles en kit, l'utilisation des blocs émulés conduit à un modèle beaucoup plus clair. La modélisation du stock de produits semi-finis nous avait par exemple posé problème : un modèle avait été mis au point, mais son fonctionnement était capricieux, et sa complication le rendait non ré-utilisable. De plus ce modèle répondait à des demande de libération d'un produit dont on donnait la référence. En comparaison, la transformation d'espace dédié à la sortie du stock ne prend pas une référence, mais un identifiant d'espace. La différence entre ces deux approches est notable. En effet, dans le premier cas, il est impossible d'étudier la gestion des produits dans un stock dont la capacité peut être limitée, ainsi que les problèmes liés à une dé-synchronisation entre la représentation du stock dans le système de pilotage et son état réel (par exemple lorsqu'un produit est mal rangé).

On peut donc constater que dans ce cas d'application l'utilisation des modules d'émulation

a conduit à un modèle plus simple et plus clair, en fait plus fidèle à la réalité de l'atelier que la modélisation traditionnelle.

2.6 Conclusion

Ce chapitre a présenté la définition, le développement et la validation d'un dispositif fondé sur l'émulation du système opérant, permettant l'évaluation d'un système de pilotage par le produit. L'architecture de ce dispositif est proche de celles présentées dans la littérature. Néanmoins, nous contribuons à la création de modèles d'émulation *génériques* en proposant une méthodologie de modélisation basée sur des primitives de modélisation systémiques. De plus, nous contribuons à clarifier les différentes options d'intégration entre le système de contrôle et le système opérant émulé, en définissant deux niveaux d'intégration, qui correspondent aux tests *software-in-the-loop* et *hardware-in-the-loop* de la mécatronique.

Une limite des travaux de ce chapitre est la validation des modèles spécifiant les composants d'émulation. Dans nos travaux, nous nous sommes en effet intéressés à construire le « bon modèle », c'est à dire celui dont la manipulation serait sémantiquement la plus logique pour développer un environnement de test. Cette approche est validée par l'implémentation du modèle et ses applications à des cas réalistes. Par contre, la construction correcte des diagrammes n'est pas assuré par l'environnement de modélisation UML ArgoUML[20] qui a été utilisé. Assurer que nous avons écrit des « modèles bons » nécessiterai d'utiliser des méthodes plus formelles de validation comme l'utilisation de B [Souquières et Thuan Truong, 2006], ou plus pragmatiquement de vérifier des règles de bonnes pratiques de modélisation.

Un prototype basé sur Arena, supportant cette méthodologie ainsi que ces deux niveaux d'intégration a été développé. La phase de validation montre que la méthode est exploitable. Elle permet en effet une modélisation plus facile et structurée, en proposant des modules de simulation prêts à l'emploi, qui encapsulent la complexité à la manière d'une approche orientée objet. De nombreux aspects restent toutefois à améliorer.

Les futurs développements seront dirigés vers la prise en compte des modifications que la validation de notre outil nous a conduit à suggérer. En particulier, nous chercherons à intégrer le paramétrage des modules afin de pouvoir automatiser la génération des modèles d'émulation et faciliter leur intégration avec les bases de données techniques. De même, le comportement des modules d'émulation pourra être affiné, de manière à prendre en compte la mobilité des ressources. Enfin, nous devrons définir la manière de modéliser les assemblages et dé-assemblages, et introduire les modules correspondant dans notre outil d'émulation.

Pour faire de l'outil d'évaluation par émulation un véritable logiciel distribuable, et non plus un simple prototype, il nous faudra nous affranchir de la dépendance à un simulateur particulier. Cela peut être réalisé en spécifiant un format neutre pour stocker et échanger les modèles d'émulation, et éventuellement en utilisant un autre environnement de simulation, spécifiquement développé, ou basé sur des bibliothèques libres.

Dans sa version actuelle, le prototype convient à nos besoins. Néanmoins, il ne permet que de représenter le système opérant ; il nous faut donc compléter ce système d'émulation par un système de pilotage par le produit effectif, qui est présenté au chapitre suivant.

[20] ArgoUML : http ://argouml.tigris.org

Chapitre 3

Environnement à base de composants orientés-agent pour le pilotage hybride de systèmes de production contrôlés par le produit

Pour étudier le couplage par le produit des modes centralisés et distribués de prise de décision, nous nous sommes munis au chapitre précédent d'un outil permettant d'émuler des systèmes de production, afin d'avoir à notre disposition des cas de tests. Dans ce chapitre, nous continuons à construire un environnement d'évaluation du contrôle par le produit, en nous munissant d'un système de contrôle mettant en œuvre un pilotage par le produit.
Ce système est basé sur une approche holonique, et utilise des technologies multi-agents, afin d'étudier les phénomènes liés à l'exécution parallèle des centres de décision, et aux échanges d'informations entre centres.
Dans un premier temps, nous définirons le contexte de fonctionnement d'un tel système, ses fonctions, les entités qu'il comprend.
Ensuite nous verrons comment on peut développer un tel système sous la forme d'un système multi-agents.
Enfin, nous appliquerons ce système à un cas simple afin de valider son fonctionnement.

3.1 Contexte

3.1.1 Principe : la stigmergie

Nous avons présenté au premier chapitre la notion de *stigmergie*, et nous avons montré comment elle a été utilisée pour générer un pilotage émergeant de la production, en s'inspirant du comportement des colonies de fourmis. Le principe de stigmergie peut toutefois être interprété différemment. Il est par exemple possible d'assimiler le fonctionnement de l'encyclopédie collaborative Wikipedia à de la stigmergie. En effet, cette encyclopédie publiée sur internet est écrite collectivement *par ses lecteurs*. Ceux-ci ont la possibilité de modifier un article lorsqu'ils l'estiment erroné ou incomplet. Les différents rédacteurs d'un article ne communiquent donc pas *directement* (en se transmettant par courrier électronique les versions successives de l'article), mais *indirectement* en se basant sur l'article lui-même et sur les annotations qui l'accompagnent

éventuellement. C'est donc bien l'œuvre collective (les articles) qui sert de vecteur de coordination.

La stigmergie peut donc servir à faire coopérer un grand nombre d'acteurs, en faisant de l'œuvre collectivement élaborée le vecteur des échanges d'information.

Cette vision de la stigmergie peut être transposée au domaine du pilotage de la production. L'œuvre collective, dans ce cas, est constituée par le flux des produits. Ces produits, dotés d'une instrumentation adéquate, pourront être observés, et éventuellement annotés par les différents acteurs du pilotage de la production. Par rapport au problème particulier de l'interaction entre décisions distribuées et décisions centralisées, le produit devient le vecteur des échanges entre chaque système de décision, tout en restant directement observable par chaque sous-système. On peut ainsi assurer la coexistence des deux modes de décision, en même temps que les fonctions de traçabilité des produits.

Le principe de l'*infrastructure de support de la décision* que nous proposons s'inspire de cet usage de la stigmergie. Il s'agit d'offrir aux divers acteurs du pilotage de la production la possibilité d'enrichir chaque produit d'informations et de percevoir ces informations. La stigmergie est donc une manière alternative de considérer le couplage entre les systèmes de décision centralisés et distribués, fondée par une intégration *indirecte* entre ces deux sous-systèmes.

Nous parlons d'infrastructure de support de la décision en donnant au terme d'infrastructure le sens de « système qui supporte le fonctionnement et l'organisation ». Nous nous focalisons donc sur les échanges d'informations entre les centres de décision, plutôt que sur les problématiques d'intelligence artificielle ou de recherche opérationnelle qui se posent pour mettre au point les procédures de décision.

3.1.2 Acteurs de l'environnement

L'environnement dans lequel s'inscrit l'infrastructure de support de la décision peut être divisé en un environnement décisionnel, et un environnement opérationnel :

L'environnement décisionnel comprend les centres de décision centralisés et distribués dédiés au pilotage de la production.

L'environnement opérationnel comprend les équipements de l'atelier permettant l'observation, les transformations des produits, qui ont déja été présentés au deuxième chapitre.

On peut représenter l'apport du système à son environnement en déterminant les relations qu'ont les acteurs avec, ou à travers le système (tableau 3.1).

	CD Centralisé	CD Distribué	Ressource
CD centralisé	–	enrichir la décision locale	–
CD distribués	suivi des décisions locales	collaboration entre CD distribués	réglage par rapport aux caractéristiques du produit
Observateur	traçabilité des produits, suivi des OF	déclenchement des décisions locales	
Actionneur	compte rendu d'éxécution		–

Tab. 3.1 – Apports du système aux acteurs de son environnement

Ces interactions peuvent être classifiées en trois types de fonctions remplies par le système :

1 Représentation : il s'agit de rendre visible le flux de produits pour les centres de décision, cela recouvre donc les interactions entre les observateurs des produits et les centres de décision,

2 Exécution : cette fonction recouvre les interactions avec les actionneurs sur les flux de produits,

3 Décision : cette fonction désigne les échanges entre centres de décisions.

3.1.3 Fonctions du système

Ces trois fonctions principales peuvent être raffinées en sous-fonctions.

Les figures 3.1, 3.2 et 3.3 présentent respectivement la décomposition des fonctions de représentation des produits, d'exécution et de décision.

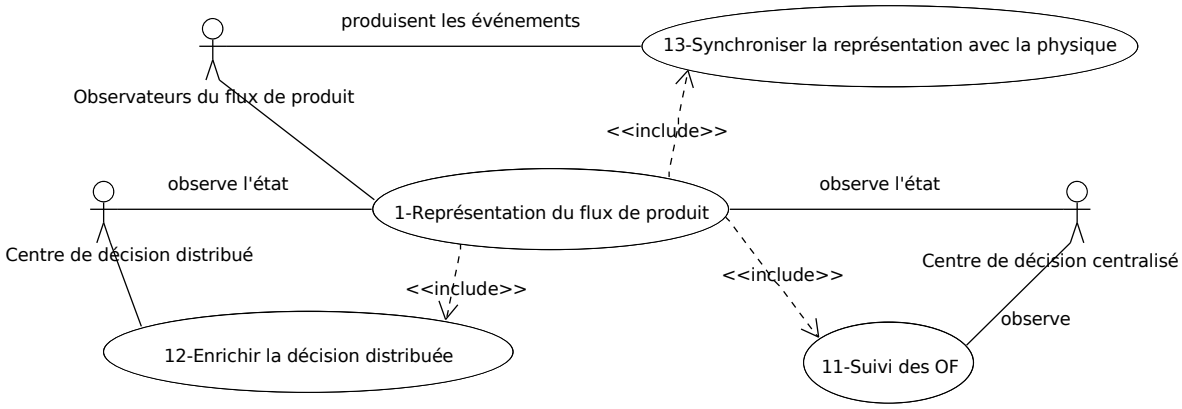


FIG. 3.1 – Décomposition en sous-fonctions des fonctions de représentation du flux de produits.

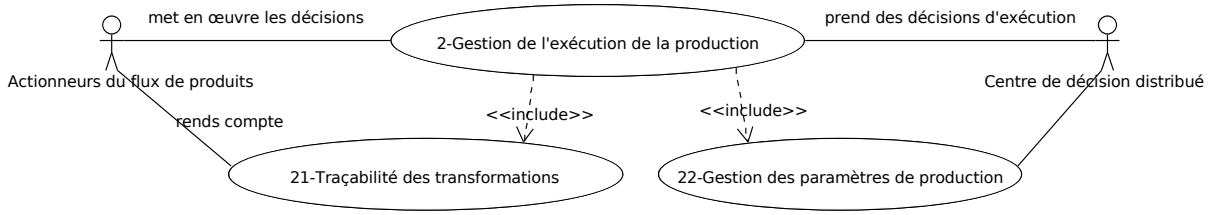


FIG. 3.2 – Décomposition en sous-fonctions des fonctions d'exécution des décisions.

Ces fonctions constituent vraisemblablement en fait un sous-ensemble des onze fonctions classiques d'un MES, comme définies par le MESA. Nous nous sommes focalisés sur les aspects de pilotage des flux, certains aspects ne sont donc pas traités, comme par exemple ceux liés à la maintenance et à la gestion du personnel. Toutefois, notre approche est assez différente de celle du MESA, et il est difficile de trouver une correspondance entre ces deux ensembles.

L'introduction véritable du contrôle par le produit se fait lorsque l'on décide de réaliser ces fonctions par le biais de lectures et d'écritures d'annotations des produits (figure 3.4).

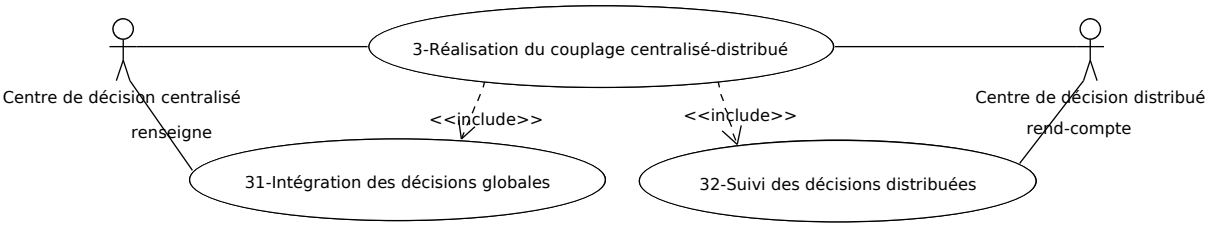


FIG. 3.3 – Sous-fonctions liées aux interactions entre centres de décision.

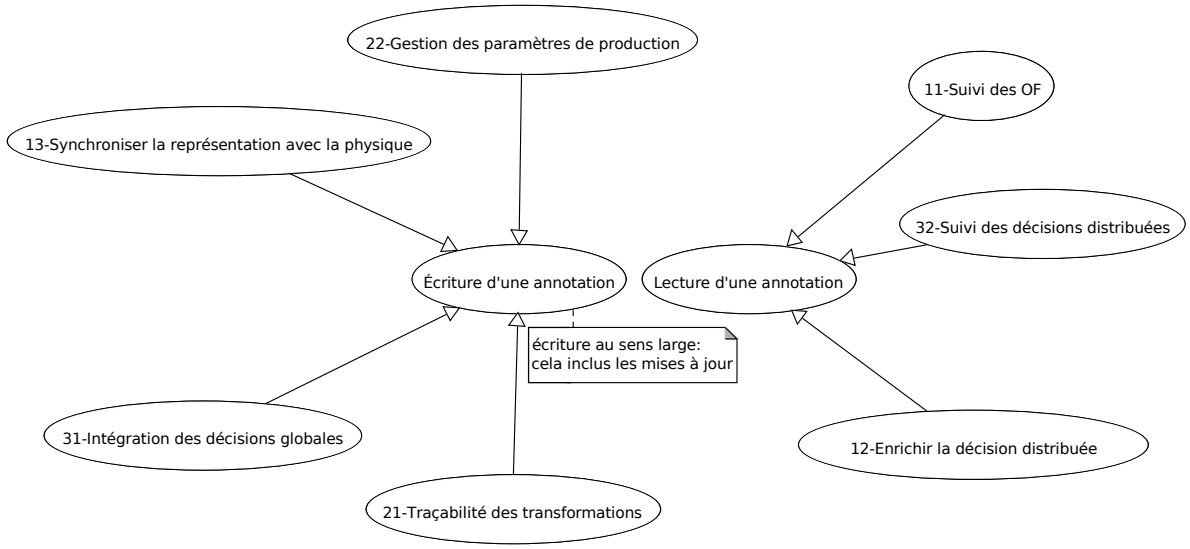


FIG. 3.4 – Interprétation des sous-fonctions comme des lectures ou écritures d'annotations sur le produit.

3.1.4 Modèle du domaine

Le diagramme de classe présenté figure 3.5 présente les entités principales de notre système, ainsi que la manière dont elles sont structurées les unes par rapport aux autres. L'entité centrale est le produit, doté de ses annotations, en application du principe de stigmergie. D'une part, chaque produit porteur d'informations est capable d'interagir avec les acteurs de l'environnement décisionnel, afin de donner accès aux informations qu'il porte. Ces échanges se font suivant un protocole d'interaction permettant éventuellement de donner l'initiative de la communication au produit. D'autre part, il est capable d'échanger des données avec les acteurs de l'environnement opérationnel.

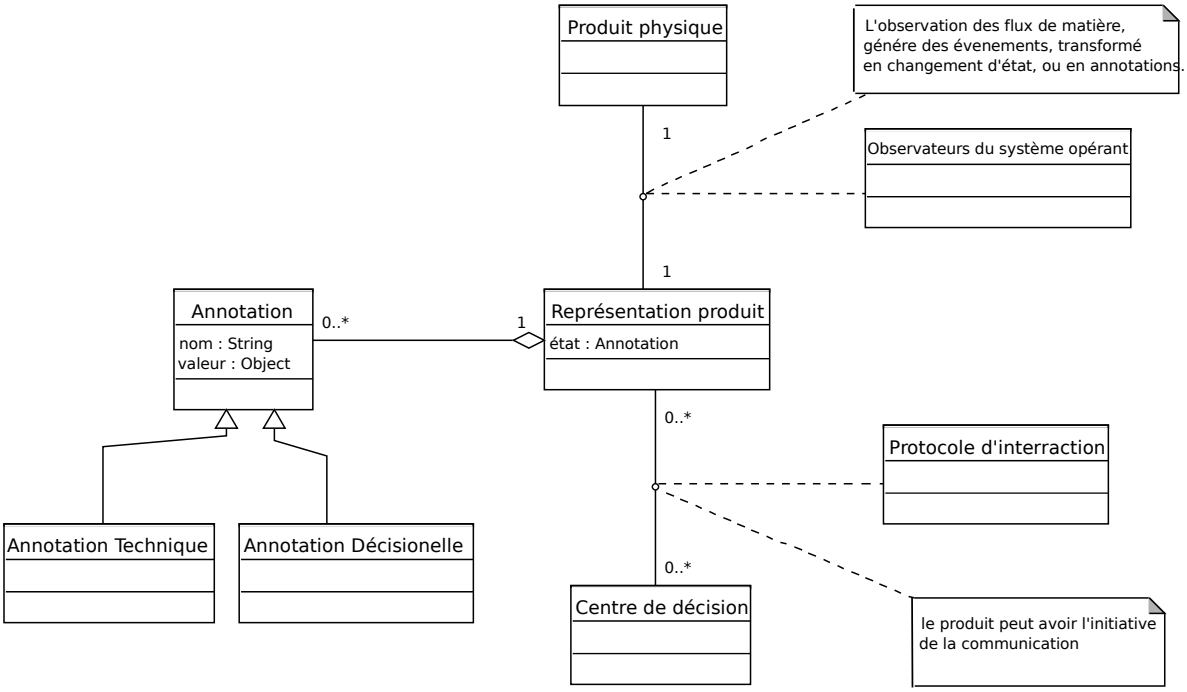


FIG. 3.5 – Modèle du domaine

3.1.5 Architecture choisie

Nous choisissons, pour mettre en œuvre ce système, d'utiliser l'approche holonique. En effet, l'association étroite entre un objet physique (un produit) et un objet informationnel (les annotations du produit) conduit à l'émergence d'un holon.

Un meta-modèle holonique a été élaboré pour la représentation du produit [Baina, 2006] (figure 3.6). La classe Holon définit la structure de base d'un holon, qu'il soit élémentaire ou complexe. Un Holon Complexe est, contrairement à un holon élémentaire, un holon ayant subis au moins un procédé de composition ou décomposition lors de sa fabrication. Une partie physique est une référence à la partie réelle du produit associée à un holon. Chaque partie physique est décrite par des attributs qui peuvent avoir des valeurs (valeur d'attribut). Une partie informationnelle regroupe l'ensemble des propriétés relatives au produit décrit par l'holon en question. Chaque propriété peut avoir des valeurs (valeur de propriété). Elle peut aussi, en cas d'holon complexe, agréger des attributs relatifs aux holons qui composent cet holon. La classe état décrit l'ensemble

des états par lesquels passe un holon. L'état d'un holon est définit par un ensemble de couples (attribut, valeur) et/ou (propriété, valeur).

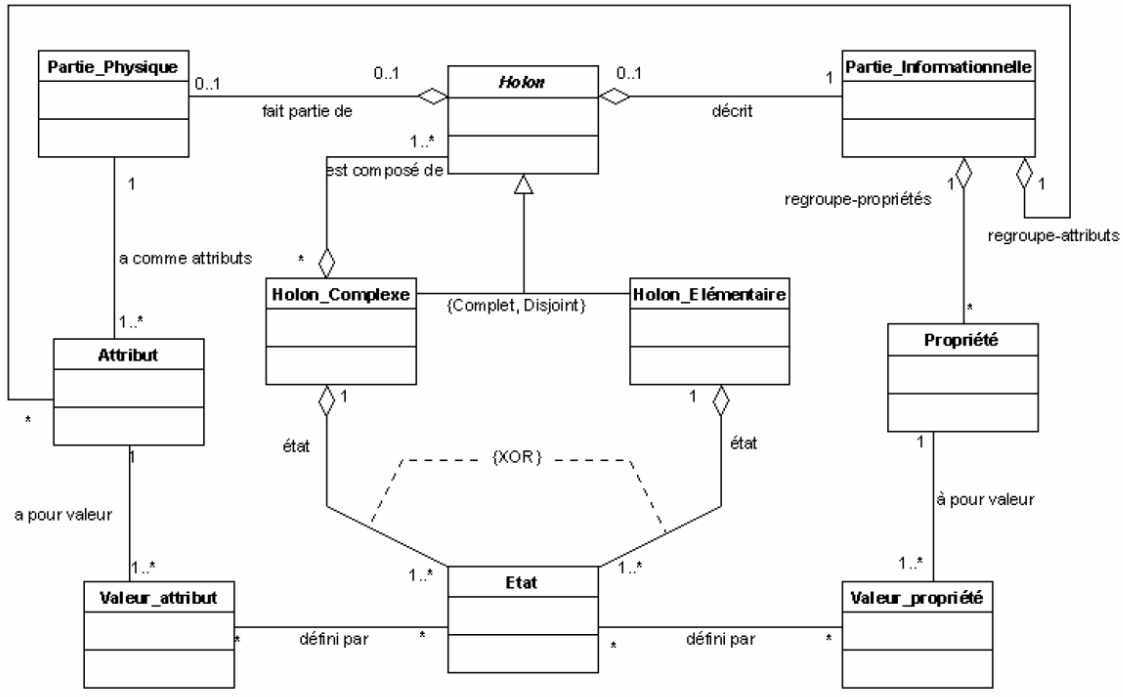


Fig. 3.6 – Meta-modèle holonique pour la représentation du produit [Baina, 2006]

Si on se réfère à l'architecture de référence PROSA, on peut faire correspondre les entités de notre domaine d'intérêt avec les différents types d'holons (produit, ordre, ressource, staff). Ainsi, les produits porteurs d'informations correspondent à la notion d'holons-ordre, bien qu'ils puissent être aussi assimilés à des holons-produit dans la mesure où ils peuvent être porteurs de données techniques spécifiant la manière de les élaborer. Les actionneurs de l'atelier correspondent à des holons-ressource, et les centres de décision centralisés ou distribués à des holons-staff. Lorsque les décisions distribuées ne concernent qu'une unique ressource, ce choix peut être discuté, les décisions étant embarquées dans l'holon-ressource.

Pour mettre en œuvre cette approche, nous utiliserons les technologies multi-agents pour supporter les éléments informationnels associés à chaque holon. En effet, l'état actuel de la technique ne permet pas d'embarquer les entités actives dans les produits ou les machines, à cause des limites de leur puissance de traitement et leur capacité à transmettre toutes les informations requises. Les entités actives doivent donc résider dans un ordinateur distant du système physique, et être liées aux éléments, produits et machines, de ce système.

De plus, du point de vue de l'expérimentation, les systèmes multi-agents permettent de prendre en compte le caractère distribué d'un système contrôlé par le produit. Ainsi, un système multi-agents permet l'exécution concourante de différentes procédure de décision. De plus, il permet l'étude approfondie des échanges d'informations entre agents, le transfert d'une information ne pouvant être réalisé que par l'utilisation de messages et de protocoles de communication explicitement définis.

3.2 Définition du système multi-agents

3.2.1 Types d'agents

Cette section décrit le système multi-agents qui a été développé.

Le modèle de développement qui a été adopté repose sur le paradigme MVC (Modèle Vue Contrôleur). Chaque type d'agent est donc constitué d'un modèle, qui comprend à la fois les données permettant de représenter l'état du domaine d'intérêt de l'agent, ainsi que les méthodes offrant une interface avec ces données, et permettant de les faire évoluer[21]. Les vues et contrôleurs permettent respectivement de représenter le modèle ou d'agir dessus. Le principal avantage à l'utilisation du MVC est la dissociation entre le modèle des données, et la présentation de ces données. Pour adapter ce modèle de développement à la structure d'un système multi-agents, nous considérons la communication avec les autres agents comme une vue ou un contrôleur. En effet, les messages reçus peuvent produire des évolutions du modèle, de même que l'état du modèle est restranscrit dans les messages émis, de la même manière que dans une vue.

De plus, le développement des agents a été réalisé à l'aide de la plateforme multi-agents JADE[22] [Bellifemine *et al.*, 2001] qui implémente les normes de la FIPA[23]. Ces normes définissent de très nombreux aspects du fonctionnement d'un système multi-agents, y compris le cycle de vie des agents, la syntaxe utilisée pour les messages de communication inter-agents (ACL – Agent Communication Language), les protocoles d'interactions dans lesquels s'inscrivent ces messages.

La plateforme JADE facilite donc considérablement le développement d'un système multi-agents, en offrant des fonctions de gestion du cycle de vie des agents, des fonctions facilitant l'envoi ou la reception de messages, etc... Ainsi, le développement des agents se résume à la programmation du modèle (les données maniées par l'agent et les algorithmes qui les font évoluer) et des divers comportements que l'agent devra suivre tout au long de son cycle de vie (en particulier ceux permettant la communication avec ses pairs par envoi et reception de messages).

La programmation des agents est réalisée grâce à la spécialisation des classes génériques offertes par JADE. Ainsi, tous les agents développés héritent de la classe jade.core.Agent, qui comprend les attributs et les méthodes communes à tous les agents. Ces classes spécialisées, peuvent alors remplacer certaines méthodes génériques par des instructions spécifiques. C'est par exemple le cas de la méthode *setup*, qui est appelée dès que l'agent est prêt à fonctionner, et dont l'implémentation générique ne fait rien.

Nous avons repris à notre compte cette méthode de développement, en construisant un arbre de spécification (figure 3.7), permettant à la fois de mutualiser la programmation des fonctionnalités communes, et d'autre part d'obtenir les différents types d'agents requis dans le système.

La présentation du système multi-agents est donc structurée d'une même manière, en définissant successivement les différents types d'agents utilisés, et les composantes principales qui leur correspondent.

[21] Le motif de conception (*design patern*) MVC à été introduit en 1979 au Laboratoire PARC de Xerox, et développé en SmallTalk. Ce modèle est maintenant implémenté dans de nombreux langages et bibliothèques. En particulier, l'API Swing de java, est fondé sur ce modèle.

[22] Java Agent DEvelopment framework, http ://tilab.jade.org

[23] Foundation for Intelligent Physical Agents, http ://www.fipa.org. La FIPA est le onzième *technical comitee* de l'IEEE computer society

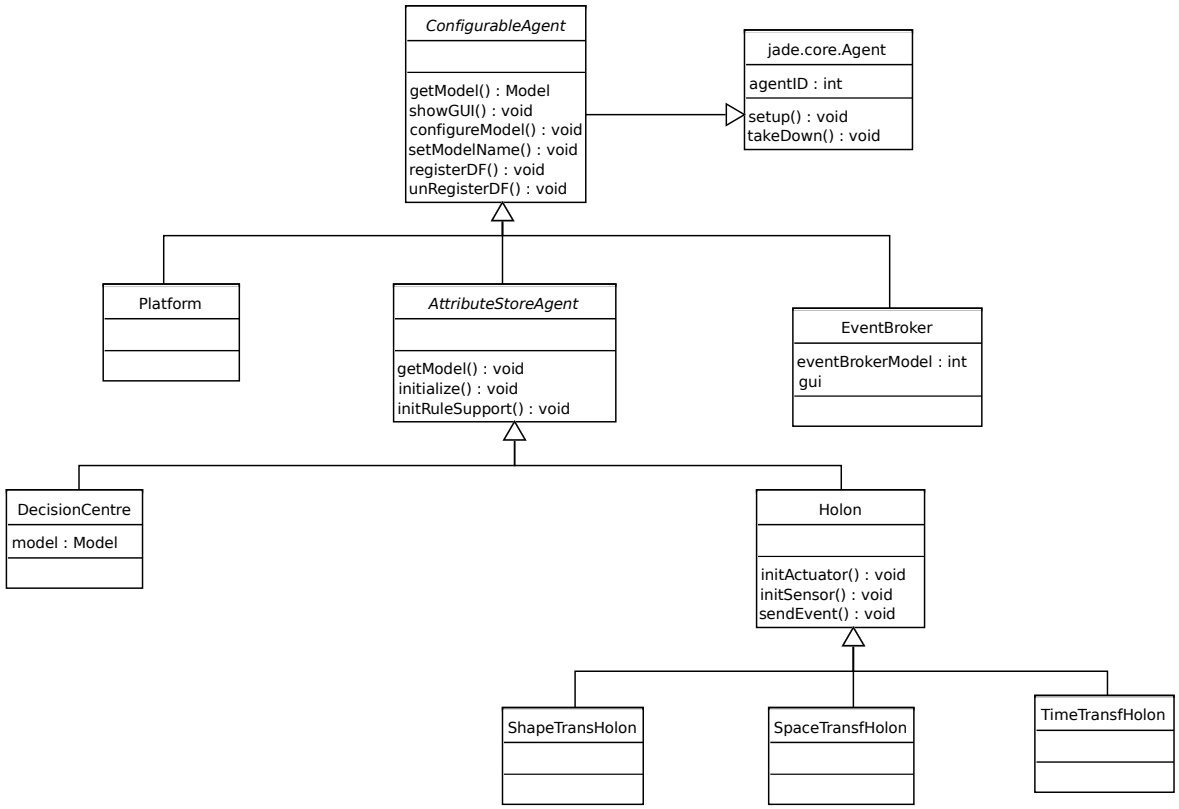


FIG. 3.7 – Typologie des agents développés

3.2.2 Configuration des agents

Les agents ont un mécanisme de configuration commun, défini dans la classe abstraite AbstractConfigurable, qui spécialise jade.core.Agent. En effet, tous les agents sont basés sur différents types de modèles, mais qui implémentent tous l'interface Model. Cette interface comprend en particulier une méthode de configuration, qui prend en argument un fichier XML, désigné par son chemin. Ainsi, la première action effectuée par un agent est l'appel de cette méthode, afin de charger une configuration du modèle. Les fonctions de conversion des fichiers XML en objets java ont été réalisées en utilisant le composant Jibx[24]. L'utilisation de ce composant facilite grandement les conversions, puisqu'il suffit de définir un fichier de mapping entre les éléments XML et les attributs d'une classe ; les conversions sont ensuite totalement transparentes.

L'interface Model définit aussi des fonctions permettant l'accès aux données du modèle (par exemple le nom de l'agent, les noms des services qu'il remplit, etc...). L'appel de ces fonctions permet d'inscrire l'agent nouvellement créé dans le registre (le *directory facilitator* ou DF) de la plateforme agent.

3.2.3 Gestion de la plateforme agent

Le premier agent qui est lancé est l'agent de gestion de la plateforme. Celui-ci remplit les fonctions liées à la création et à l'initialisation des autres agents.

La création d'un agent requiert de spécifier le nom du nouvel agent, ainsi que son type (le nom de la classe devant être instanciée). En plus de ces informations nécessaires, la plateforme JADE permet d'associer des arguments à chaque agent qui est lancé. Ce système d'argument a été préféré à une initialisation directe du modèle de l'agent. En effet, ce passage par les arguments permet une plus grande flexibilité dans le lancement de l'agent, tout agent pouvant indifféremment être lancé par l'agent spécifique de gestion de plateforme, ou bien en utilisant les interfaces standards de JADE.

Nous utilisons donc deux arguments : le premier est le chemin du fichier XML contenant les véritables informations de configuration, le second est un booléen permettant de choisir si l'agent nouvellement créé affichera une interface graphique.

On peut donc définir une classe AgentConfig, comprenant ces quatre informations, et pouvant être représentée soit sous la forme d'une structure XML, soit sous la forme d'un objet java standard.

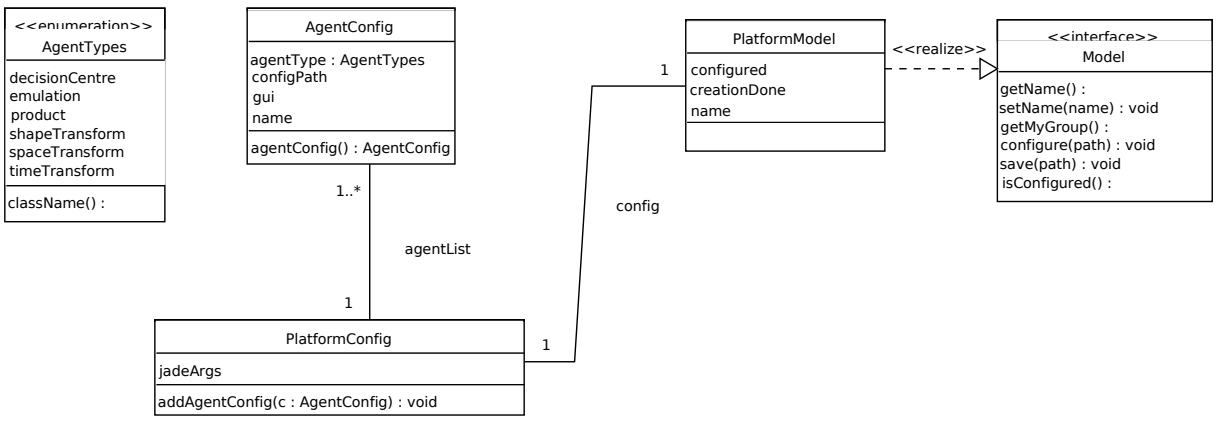


FIG. 3.8 – Structure interne de l'agent de gestion de la plateforme.

[24]http ://jibx.sourceforge.net

```
<platformConfiguration>
    <jade arguments="-container -gui"/>
    <emulationAgent>
        <agentConfig name="emuBrocker" path="./config/emu.xml" gui="true"/>
    </emulationAgent>
    <shapeTransformAgents>
        <agentConfig name="P1" path="./config/chargeur.xml" gui="false"/>
        <agentConfig name="P2" path="./config/chargeur.xml" gui="false"/>
    </shapeTransformAgents>
    <spaceTransformAgents>
        <agentConfig name="robot" path="./config/robot.xml" gui="false"/>
    </spaceTransformAgents>
    <timeTransformAgents>
        <agentConfig name="bufferRobot" path="./config/time.xml" gui="false"/>
        <agentConfig name="bufferP2" path="./config/time.xml" gui="false"/>
        <agentConfig name="bufferP1" path="./config/time.xml" gui="false"/>
    </timeTransformAgents>
    <decisionCentres>
        <agentConfig name="planing" path="./config/planing.xml" gui="false"/>
        <agentConfig name="robotStart" path="./config/rule-autofeed.xml" gui="false"/>
    </decisionCentres>
</platformConfiguration>
```

TAB. 3.2 – Exemple de fichier de configuration de la plateforme multi-agents.

L'agent de configuration est lui-même instancié de cette manière. Son propre fichier de configuration comprend donc l'ensemble des paramètres de création des agents. Le fichier XML de création de la plateforme comprend donc autant de structure Agentconfig que d'agent devant être créés. Celles-ci sont classées par type d'agent (voir figure 3.2).

Un protocole d'interaction permettant de s'assurer de la synchronisation des agents de la plateforme a été développé. Cette synchronisation est nécessaire car certaines opérations d'initialisation ne doivent être faites que lorsque tout les agents sont créés. L'agent de plateforme mémorise donc l'état des agents qu'il crée. Chaque nouvel agent notifie l'agent de plateforme après son initialisation (ajout à la plateforme agent, configuration, inscription des services offerts dans l'annuaire du système). Lorsque tous les agents sont prêts, l'agent plateforme notifie chaque participant.

3.2.4 Connexion avec le système physique

En plus de l'agent de plateforme, un autre agent utilitaire a été développé. Il s'agit de l'agent de connexion avec le système physique (EventBrokerAgent). Cet agent remplit un double rôle :

- Il crée et maintient une connexion avec le système contrôlé, réel ou émulé,
- Il route les événements en provenance de ce système vers les agents concernés, en demandant éventuellement leur création s'ils n'existent pas.

Les fonctionnalités de création et de maintien d'une connexion sont naturellement dépendantes de la nature du système avec lequel il faut se connecter. Un système de greffon a été mis au point, afin de pouvoir choisir parmi les modes de connexion possibles, et pour pouvoir facilement ajouter de nouvelles cibles de connexion. Les modes de connexion actuellement implémentés sont d'une part la connexion au serveur d'émulation, à travers une socket TCP, et d'autre part la connexion à la plateforme d'essai de l'ERT TRACILog.

Les messages échangés avec la plateforme sont souvent transmis sous une forme qui n'est pas directement exploitable. L'agent de connexion est donc chargé de décoder et d'encoder les événements en provenance ou à destination du système physique. En particulier, lorsque la cible

est le serveur d'émulation, les messages sont échangés en XML, il est donc nécessaire de les convertir en objets java standards. Comme pour le traitement des fichiers de configuration nous avons utilisé jibx pour réaliser cette conversion.

La deuxième fonctionnalité est le routage des événements vers l'holon correspondant. Ce routage se base sur l'identité entre un composant émulé et l'agent qui le représente. Le protocole utilisé pour la transmission des événements vers les holons est l'abonnement (norme FIPA SC35). Ainsi, l'agent d'émulation est capable de répondre aux demandes d'abonnement des autres agents. À la réception d'une telle demande, il enregistre le nom du composant émulé concerné et l'adresse de l'agent associé à ce composant. Par la suite le routage des événements se fait sur la base de ces enregistrements.

Un cas particulier est celui des événements résultant de l'observation des produits. En effet, si on considère que les produits émulés pré-existent aux produits informationnels, il faut déclencher l'instanciation d'un holon-produit lorsque le produit physique entre dans le système. Cela est rendu possible par l'ajout de règles associant un événement à une action (par exemple la création) et à un type de produit (sous la forme d'un chemin vers le fichier de configuration correspondant). Ainsi, lorsque qu'un tel événement est reçu, un message requérant la création de cet agent est envoyé à l'agent de gestion de la plateforme ; lorsque cet agent est prêt, l'événement lui est transmis. À l'inverse, les agents représentant les produits peuvent demander à être associés à un produit physique. Dans le modèle d'émulation cela se traduit par la création d'une entité mobile, et l'assignement de son identifiant de manière à ce qu'il corresponde à celui de l'agent.

3.2.5 Gestion des attributs

Le sous-système de gestion des attributs permet l'association d'informations au produit. Ce sous-système est en fait commun à tous les agents (mis à part l'agent de gestion de la plateforme, et l'agent de connexion avec le système physique). En effet, il permet non seulement d'associer des attributs informationnels aux produits, mais aussi de gérer les attributs des autres types d'agents. Ce sous-système est organisé en deux parties : d'une part le modèle, qui permet de mémoriser et de faire évoluer les attributs, et d'autre part un ensemble de comportements, qui permettent à un agent d'accéder aux attributs des autres agents.

Modèle

D'une manière générale, un attribut est l'association entre un nom et une valeur. Pour plus de simplicité, les valeurs mémorisées sont des chaînes de caractère (string). Ces attributs se déclinent en sous-types :

- les attributs simples (PlainAttribute) sont de simples associations nom/valeur, avec possibilité de mettre à jour directement la valeur,
- les attributs basés sur un automate à états finis (FSMAttribute) voient leurs valeurs changer en fonction des évolutions d'un automate. La réception d'un événement peut produire un changement d'état, qui cause le changement de la valeur de l'attribut.
- les attributs à incréments (IncrementAttributes) sont des attributs à changement d'état particulier ; ils sont basés sur une valeur entière, qui est incrémentée ou décrémentée à la réception de certains événements.

De plus, les attributs sont observables, c'est à dire que des vues (dans le paradigme MVC) peuvent être tenues informées des changements d'état d'un attribut.

Chaque attribut est décrit par un nom unique. Celui-ci est composé de trois parties hiérarchiquement ordonnées :

– la classe d'agent auquel l'attribut est rattaché.
– le nom de l'agent de rattachement,
– le nom de l'attribut lui même ;

La concaténation de ces trois éléments désigne de manière unique chaque attribut (Type.Agent.Attribut).

On introduit de plus deux mots clés pour nommer les agents. Il s'agit de * pour désigner tous les agents, et de self pour désigner l'agent qui exécute la règle. Le premier mot clé permet ainsi de désigner tout une classe d'agents (par exemple Product.*.State désigne par exemple les attributs d'états de tous les produits) tandis que le second permet d'écrire des règles plus génériques.

Le composant de gestion des attributs comprend exactement un attribut d'état (soit un FSMAttribute, soit un IncrementAttribute) et un nombre quelconque d'annotations, sous la forme de PlainAttribute. Il est de plus doté de méthodes permettant d'obtenir la liste de ces attributs ou leur description, sous la forme de l'ensemble de leurs noms. La figure 3.9 présente la structuration des classes du modèle.

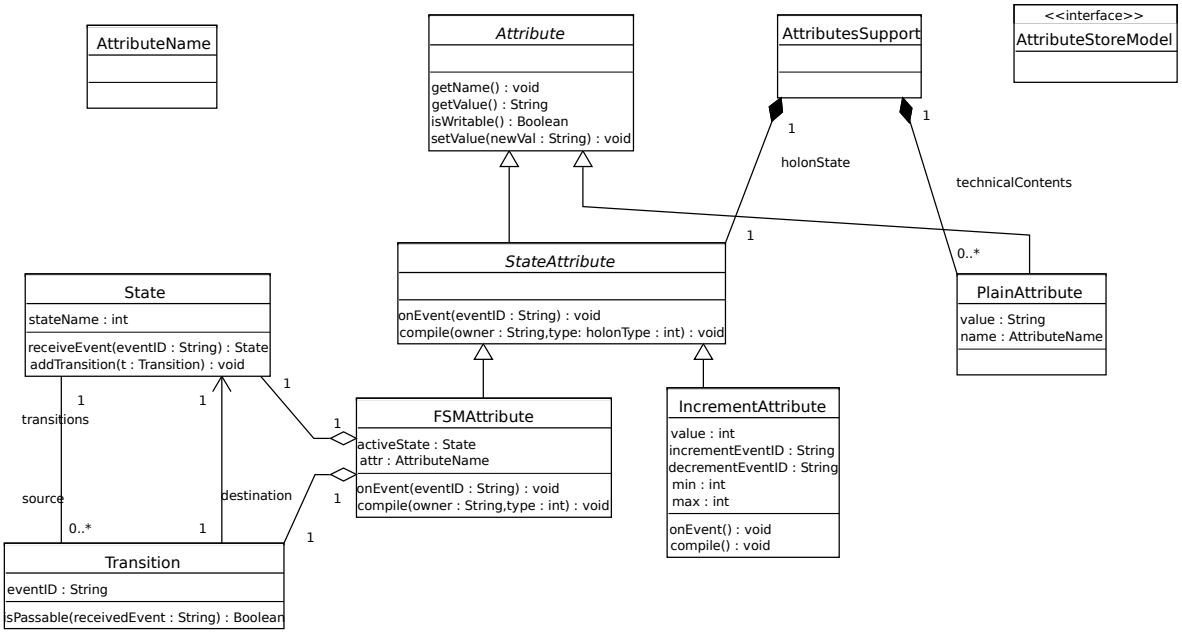


FIG. 3.9 – Structure du modèle de gestion des attributs

Comportements

Un certain nombre de comportements ont été développés pour permettre aux agents de communiquer par rapport à leurs attributs respectifs. Ces comportements correspondent aux différents modes d'interactions envisageables. Deux types d'interactionsde base peuvent être distingués :

– l'interrogation d'un pair sur la valeur de l'un de ses attributs,
– la requête faite à un pair de changer la valeur de l'un de ses attributs.

Ces deux interactions correspondent aux protocoles QUERY et REQUEST de la FIPA.

Par ailleurs, comme le produit peut avoir un rôle actif dans le système, nous ajoutons le protocole d'abonnement SUBSCRIBE (figure 3.10). Selon ce protocole, les agents intéressés par la valeur d'un attribut d'un pair s'abonnent à cet attribut. Le pair mentionne son acceptation

ou son refus au demandeur, et le notifie lorsque la valeur de l'attribut change. Cela évite donc la scrutation régulière des attributs d'un pair, comme ce pourrait être le cas si le protocole QUERY uniquement était utilisé.

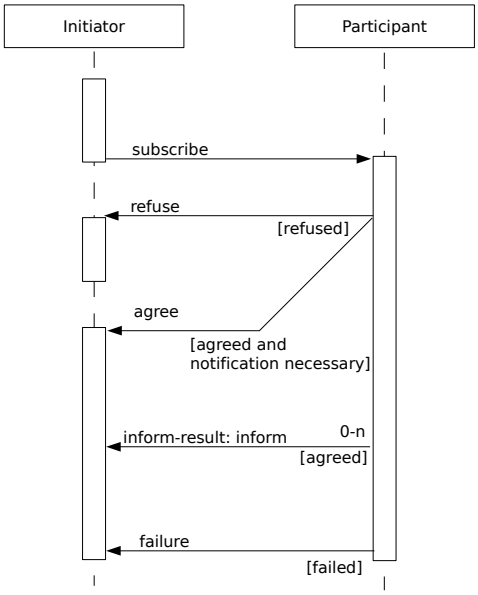


Fig. 3.10 – Le protocole d'abonnement, d'après les normes FIPA (SC00035) [FIPA, 2005]

Enfin, pour que l'abonnement soit possible, un protocole ayant pour but la déclaration des attributs a été mis au point. Ainsi, lorsque tous les agents sont prêts, chaque agent fournit à ses pairs (sur la base de leurs déclarations dans le registre des agents) la liste attributs qu'il possède. Ensuite, chaque agent recevant cette déclaration peut demander un abonnement.

Les produits, munis de leurs attributs, et capables de remplir ces trois types de services (QUERY, REQUEST, SUBSCRIBE), sont donc bien à même d'être intégrés au coeur du système d'information. Reste toutefois à synchroniser l'état des agents avec l'état des éléments du système opérant qu'ils reflètent.

3.2.6 Connexion avec le système opérant

Les agents qui sont liés à un élément du système opérant sont appelés agents holoniques. Leur fonctionnement requiert d'une part le traitement des messages provenant du système opérant, et d'autre part l'émission de commandes vers ce système (figure 3.11).

Pour se synchroniser avec l'équipement qu'il représente, chaque agent holonique s'abonne auprès de l'agent de connexion avec le système opérant. Les messages émis par l'équipement reflété lui sont donc par la suite transmis, et mis en attente dans une file. Un message d'état est toujours porteur d'un événement, qui sert à faire évoluer l'attribut d'état de l'agent holonique. De plus il peut comporter des attributs (par exemple le programme sur lequel une transformation est réglée) dont la valeur sert à mettre à jour un attribut du modèle.

Pour l'envoi de commandes vers le système opérant, chaque agent holonique est doté d'un ensemble d'opérations. À chacune d'elles correspond un message de commande pouvant être émis. Lors de son invocation un message est généré, en utilisant éventuellement les valeurs des attributs de l'agent. Le message est ajouté à la file des messages en attente, pour être ensuite transmis à l'agent d'émulation.

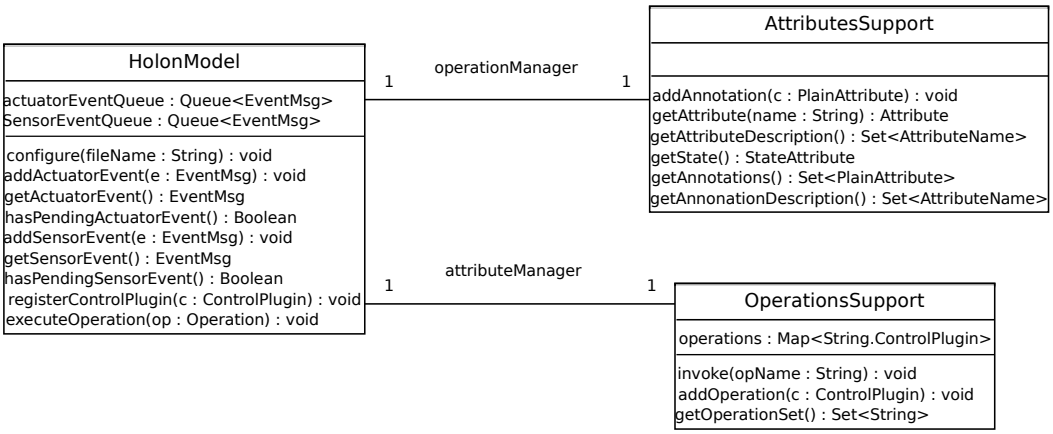


FIG. 3.11 – Modèle d'un agent holonique

Les opérations sont contenues dans des greffons de contrôle (figure 3.12), afin de gérer de manière modulaire la configuration des agents.

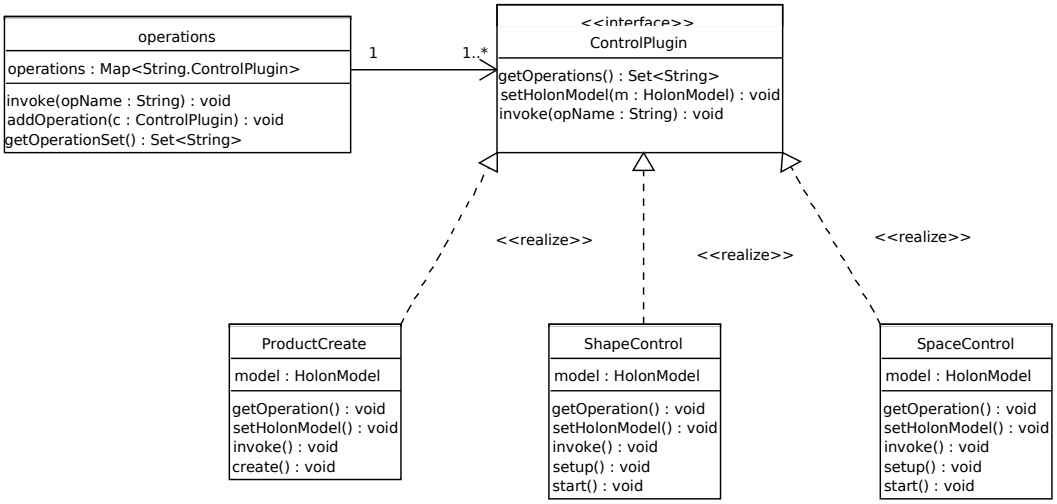


FIG. 3.12 – Sous-système de support de la commande des équipements

Les agents holoniques sont dotés de comportements qui s'interfacent avec le modèle. Ceux-ci permettent la communication avec l'agent d'émulation (initiation et gestion de l'abonnement, envoi des messages) et permettent aussi de recevoir des requêtes des autres agents, demandant l'exécution d'une opération

3.2.7 Système à base de règles

Pour animer les agents, on les dote d'un système à base de règles. L'architecture de ce système a été proposé pas Simaõ [Simao, 2005] [Simao *et al.*, 2006]. Les règles sont composées de conditions et d'actions déclenchées par les conditions. La valeur d'une condition est déterminée par la valeur des attributs de certains holons, la notification d'un changement d'état étant à

l'initiative de l'holon, en utilisant le système d'abonnement présenté précédemment. À l'inverse, les actions consistent à appeler les méthodes des holons, ou bien à changer la valeur de leurs attributs. L'architecture globale est présentée figure 3.13 [Simao, 2005].

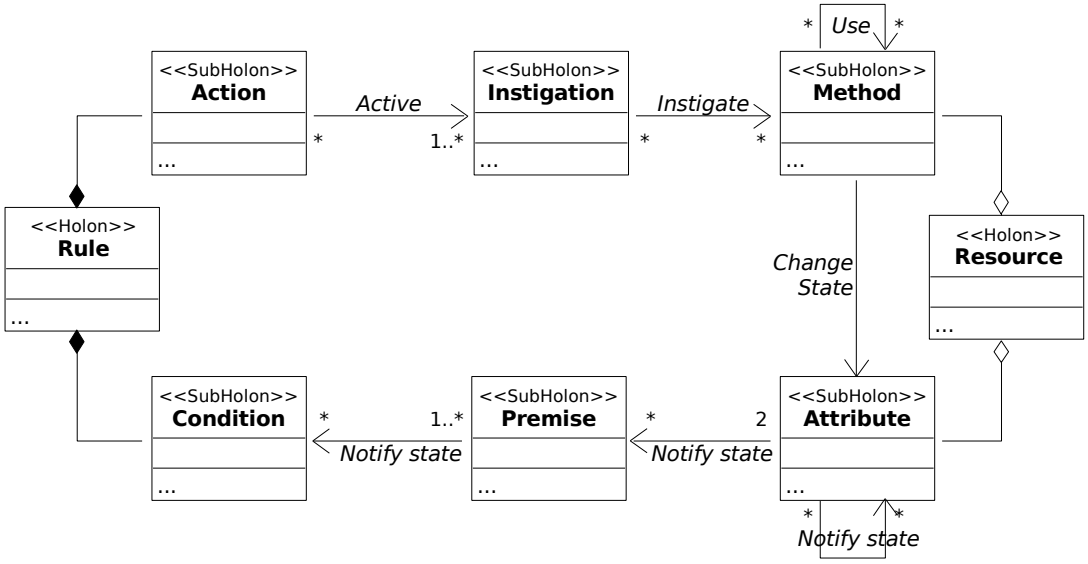


FIG. 3.13 – Système à base de règles par notification.

Notre implémentation de ce système à base de règles repose sur la description d'une interface pour les règles, permettant de mettre en place un système de greffon. De cette manière, il est possible d'intégrer des règles spécifiquement développées pour un cas d'application particulier.

En plus de cette interface, divers types de messages ont été définis afin de permettre l'action des règles sur les holons. Ces messages permettent d'appeler les méthodes de contrôle des holons, de changer la valeur d'un attribut ou de demander la valeur d'un attribut, ou enfin de demander la création d'un nouvel agent produit (figure 3.14).

En conclusion de cette section, un ensemble de classes a été développé, pour faciliter le développement des (modèles de) systèmes de décision. Ces classes constituent un environnement de développement comprenant :

- des services comme la connexion à un système opérant, la création et la configuration des agents,
- la représentation sous forme d'agents les différents éléments du système de décision,
- divers modes de communication entre ces agents,
- la possibilité d'adjoindre des règles comportementales spécifiques.

La section suivante présente les tests effectués afin de valider le bon fonctionnement de ce système, et d'illustrer de manière concrète son utilisation.

3.3 Validation

3.3.1 Tests unitaires

Comme dans le chapitre précédent, nous avons mené des tests unitaires tout au long du développement du système multi-agents[25]. De cette manière, nous avons pu vérifier le bon fonction-

[25]les tests ont été réalisés en utilisant le composant de test JUnit

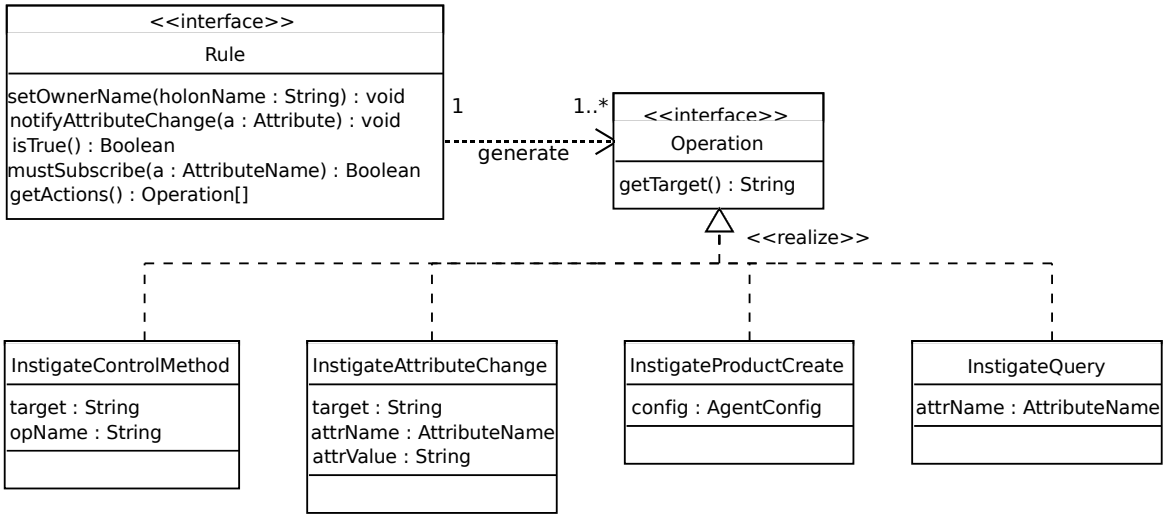


FIG. 3.14 – Interface d'une règle, et différents types d'actions pouvant résulter de l'exécution de la règle

nement des modèles pour chaque type d'agent, en particulier pour les fonctions de configuration.

Toutefois, l'utilisation d'un système multi-agents limite les possibilité de tests unitaires. En effet, les fonctions essentielles des agents sont liées à leurs échanges avec les autres agents. Cet aspect ouvert est difficilement intégrable dans l'environnement de test.

Par ailleurs, les principaux problèmes liés aux systèmes multi-agents *émergent* du fonctionnement concourant des agents. Ces difficultés peuvent être identifiées à trois problèmes élémentaires de synchronisation des processus :

- l'interblockage, (ou dead-lock) se produit lorsqu'un agent est en attente d'une action d'un autre agent, lui même bloqué en attente d'une action du premier.
- le live-lock, est une situation comparable, avec une sorte de résonance entre les comportements des deux agents. Un exemple de la vie courante illustrant parfaitement cette situation est : deux piétons sur un trottoir, qui, cherchant mutuellement à s'éviter, se trouvent face à face.
- la famine, est causée lorsque l'accès aux ressources n'est pas garanti. Elle se produit par exemple lorsqu'une tâche de faible priorité entre en compétition avec de nombreuses tâches de haute priorité, la tâche de faible priorité pouvant subir une attente indéfiniment longue.

Les protocoles et les comportements que nous avons définis doivent donc être validés en regard de ces problèmes potentiels. Malheureusement, ces derniers ne se produisent que lors de l'exécution du système. À cause des difficultés liées au test d'intégration des systèmes multi-agents, et plus généralement à la difficulté de tester les applications à exécutions parallèles (lorsque plusieurs flux d'exécution sont simultanément actifs), le système multi-agents est principalement testé dans ses conditions normales d'utilisation, en l'appliquant à un cas d'étude.

3.3.2 Initialisation du système multi-agents

Le premier test que nous menons porte sur l'initialisation du système multi-agents. Pour effectuer ce test, nous avons choisi une configuration très simple, puisque seul un agent holonique est utilisé. Le système opérant émulé comporte un unique module de transformation, pouvant agir sur des produits se trouvant dans un tampon en entrée, et comportant un tampon en sortie. Le système comporte, en plus de cet agent holonique, les agents utilitaires :

- soit les fonctions de base de l'environnement JADE (notablement le registre ou DF *directory facilitator*),
- soit celles spécifiques à notre environnement (connexion au système opérant, gestion de la plateforme).

La figure 3.15 présente la configuration utilisée dans ce premier test. La figure 3.16 présente les messages échangés lors de l'initialisation de ce système. Cette trace à été obtenue en utilisant l'agent « espion » (*sniffer agent*) faisant parti des outils de développement et d'analyse de JADE.

Ce test nous permet de vérifier que l'initialisation de la plateforme se déroule comme prévu. Il nous permet de plus de vérifier que l'état de l'équipement émulé est correctement représenté par l'agent holonique, et qu'il est possible de commander cet équipement en utilisant l'agent.

3.3.3 Application à un cas simple

Le deuxième test a pour but de vérifier que le système multi-agents peut exécuter des procédures de contrôle simples. Le système opérant émulé est sensiblement le même que précédemment. Toutefois, en plus d'un agent représentant l'équipement R, les produits émulés sont munis d'agents holoniques. Les agents liés aux produits sont créés à l'occurrence d'un certain événement du système opérant à l'initiative de l'agent de connexion avec le système opérant, comme

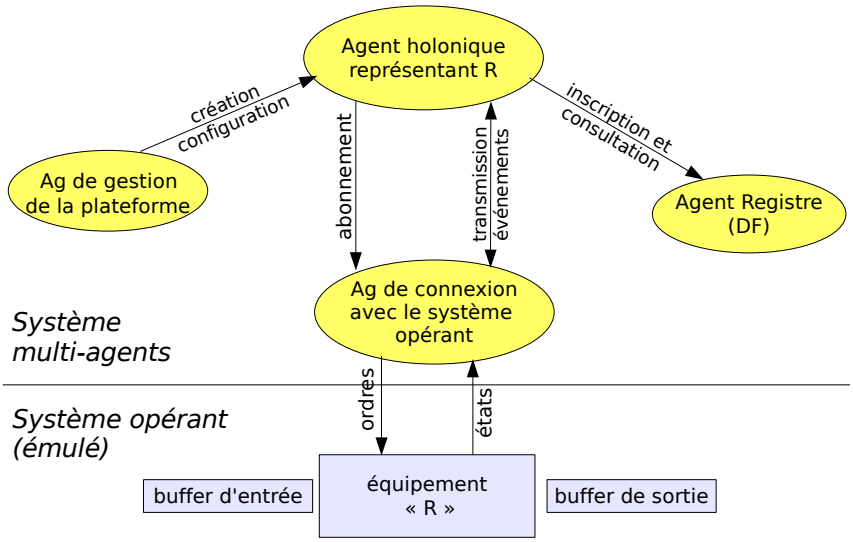


FIG. 3.15 – Configuration du système multi-agents, permettant de tester son initialisation.

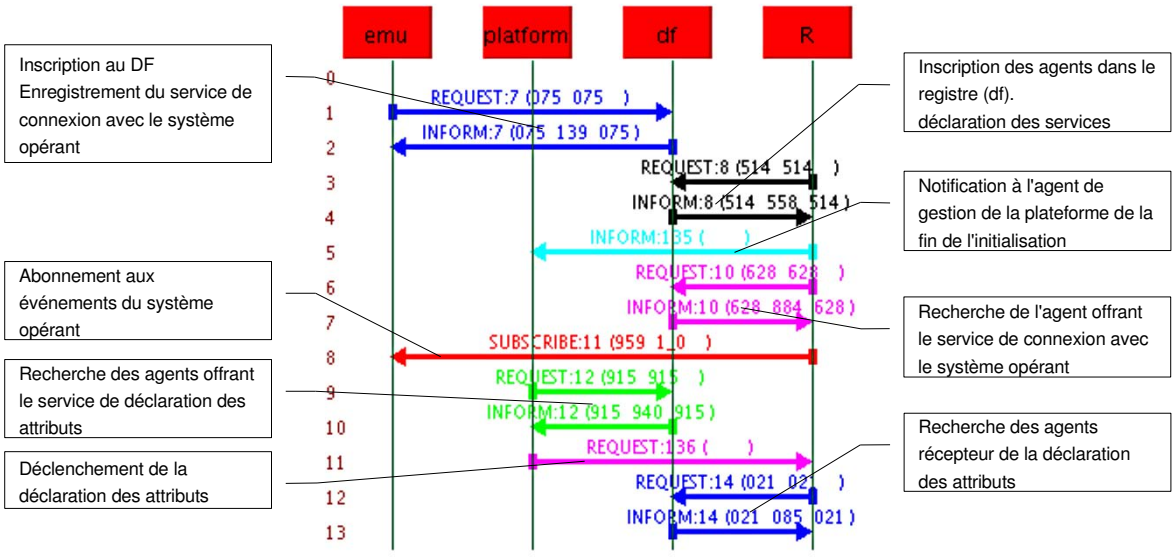


FIG. 3.16 – Messages échangés à l'initialisation du système

```
<holonAgConfiguration>
        <stateMachine>
            <fsmAttribute initialActiveState="initial">
                <states>
                    <state name="initial"/>
                    <state name="ready"/>
                    <state name="processing"/>
                    <state name="finished"/>
                </states>
                <transitions>
                    <transition sourceState="initial" destinationState="ready" eventID="readybufferR"/>
                    <transition sourceState="ready" destinationState="processing" eventID="exitBufferR"/>
                    <transition sourceState="processing" destinationState="finished" eventID="e1"/>
                </transitions>
            </fsmAttribute>
        </stateMachine>
</holonAgConfiguration>
```

TAB. 3.3 – Configuration des agents produits. Seul un attribut d'état est utilisé.

indiqué dans la configuration de ce dernier.

De plus, l'agent représentant l'équipement émulé R est muni d'une règle. Celle-ci stipule que lorsque des produits sont en attente dans le buffer d'entrée et que la ressource est disponible, la production doit commencer.

Il y a donc communication entre les agents représentant les produits et l'agent contrôlant la ressource. La configuration des produits comprend un automate à états permettant de faire évoluer l'attribut d'état de l'agent représentant le produit en fonction des événements d'observation venant du système opérant émulé (table 3.3).

Un extrait de l'échange de messages entre agents est présenté figure 3.17. Cet extrait correspond à l'arrivée d'un produit.

Ce test nous a permis de vérifier la bonne implémentation des protocoles de communication entre agents, ainsi que la gestion correcte des agents produits (leur création, l'évolution de leur état).

De plus, l'observation du système opérant nous a permis une évaluation de l'ordre de grandeur des délais globaux d'initialisation et de prise de décision (figure 3.18). Ceux-ci sont de l'ordre d'une demi seconde pour la prise de décision, et de l'ordre de quelques secondes pour l'initialisation. Le matériel utilisé pour réaliser ce test était particulièrement lent (pentium 2, 450 MHz). Les temps indiqués sont donc des majorants des délais qui seraient obtenus avec un équipement plus récent.

3.3.4 Application à un cas plus complet

Dans ce dernier test, les différents types d'agents du système opérant sont utilisés : agents utilitaires, holons, centres de décision. Les mécanismes de contrôle sont donc plus réalistes que dans les tests précédents, bien que le système émulé reste de petite échelle.

Nous considérons le cas d'étude simple de deux machines parallèles (P1 et P2), et d'un robot permettant d'acheminer les pièces d'un tampon d'entrée vers l'une ou l'autre machine. Trois types de produits doivent être traités par l'une de ces machines. Le système émulé se compose donc d'une transformation d'espace, représentant le robot, et de deux transformations de forme qui modélisent les machines P1 et P2.

Le système multi-agents est configuré de la manière suivante :

– d'une part la conduite des équipements du système opérant est effectuée par trois agents

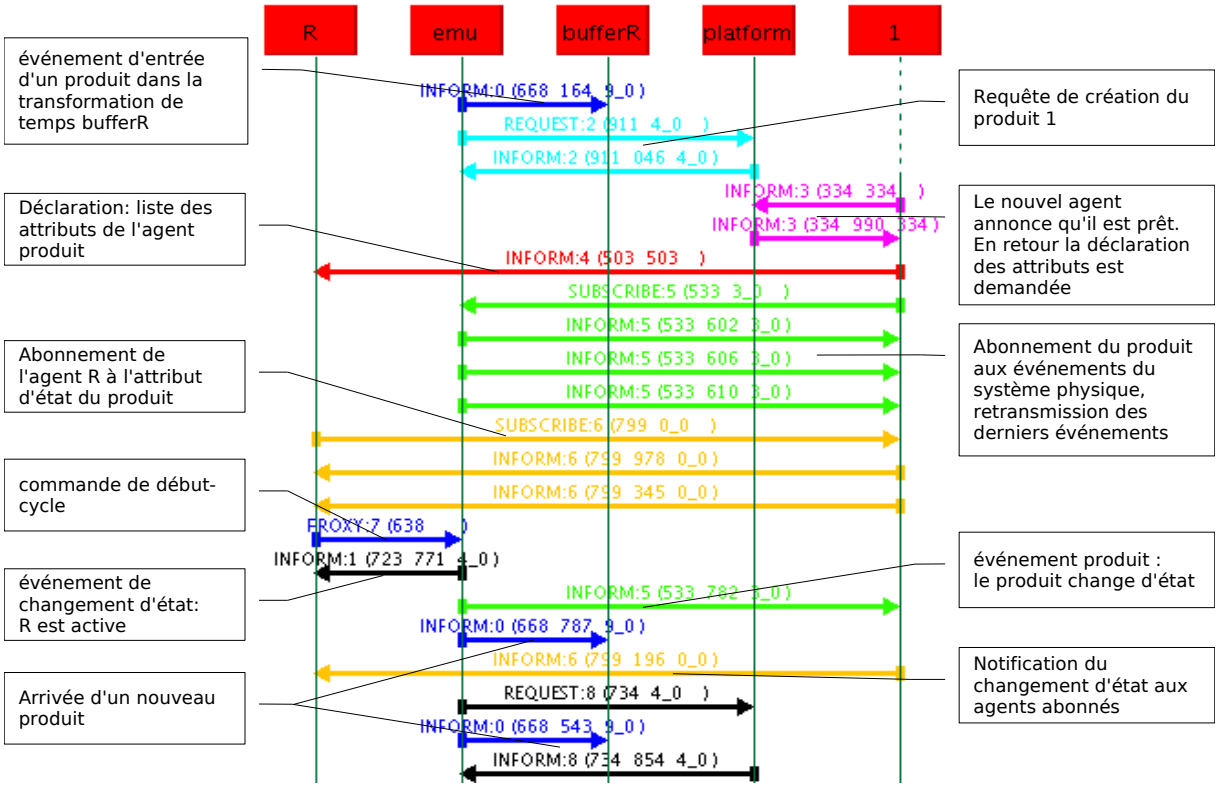


FIG. 3.17 – Messages échangés entre les agents dans un cas simple

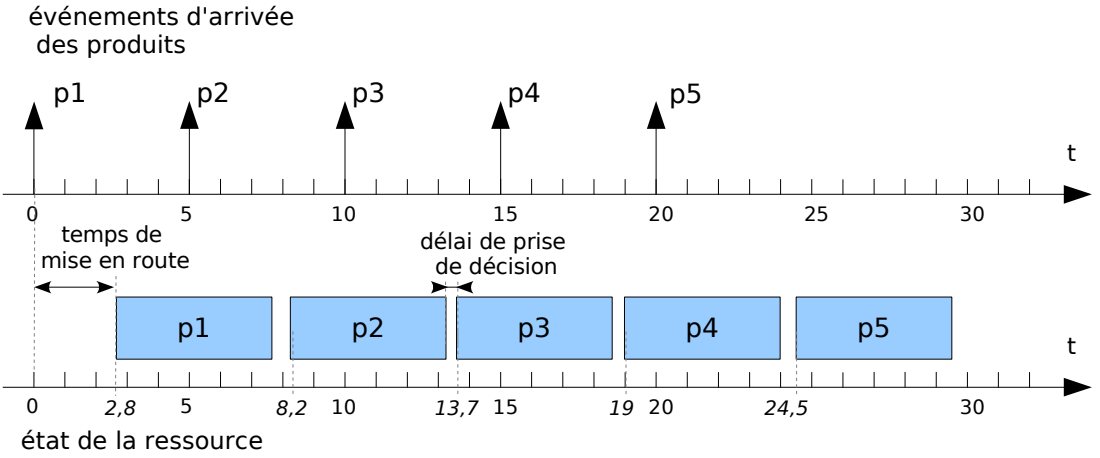


FIG. 3.18 – Chronologies des événements dans le système opérant, mettant en évidence les délais de décision (les temps sont en secondes)

holoniques. Chaque agent est doté d'une règle. Ces règles observent l'état des produits ainsi que l'état de la ressource à laquelle elle est liée. Lorsque qu'un produit se trouve dans l'état de demande de la ressource sa présence est mémorisée, et la règle interroge le produit sur ses attributs. Ensuite, en fonction de la valeur des attributs du produit et de l'état de l'environnement de l'agent, une décision est prise, par exemple le choix de la machine vers laquelle acheminer un produit. Enfin, lorsque la ressource se libère, la règle effectue un changement de série si besoin, et déclenche un cycle de production.

– d'autre part, le système comprend deux autres agents, supportant chacun une règle. La première, qui est exécutée dès l'initialisation du système multi-agents, est chargée de créer un agent pour chaque produit devant être fabriqué dans le système. Cet agent pourrait, dans une version plus aboutie de ce système, notifier un client de l'état de la production, ou encore changer la configuration d'un produit, en utilisant les mécanismes d'observation et d'action sur les attributs de l'agent lié au produit. Le second agent est chargé de l'introduction des produits dans le système émulé. Il se base sur l'observation des agents produits ainsi que sur l'occupation du tampon d'entrée. Lorsque ce tampon d'entrée est vide, un produit émulé est créé et associé à l'agent. Cet agent décide donc de l'ordre dans lequel les produits sont traités.

Ce cas de test, qui est encore imparfait, et nécessiterait de plus amples développements, montre à la fois un fonctionnement globalement correct, mais aussi certaines de ses limites, liées à sa conception ou à son implémentation.

La première de ces limites est le manque d'un environnement accessible permettant la configuration du système multi-agents. Un tel environnement d'édition devrait permettre :

– l'édition des fichiers de configuration de la plateforme,
– l'édition simple des attributs des holons (de manière graphique, en particulier lorsque ceux-ci sont des automates à état),
– la conception des règles, en utilisant éventuellement des mécanismes de traduction depuis d'autres langages et des modèles de règles,
– un support de l'exécution du système multi-agents.

Ces tâches sont actuellement réalisées par l'édition manuelle de fichier XML et de code java, ce qui les rend difficilement accessibles comme outil de recherche et développement.

Une autre limite provient de la manière dont le système à base de règles a été mis en œuvre. En effet, l'observation des attributs se fait sur leur état, et non sur une condition. Ainsi, lorsqu'un produit change d'état, il notifie tous les agents abonnés. Une solution à envisager serait de réaliser l'abonnement par rapport à une condition sur la valeur d'un attribut. Cela permettrait de réduire considérablement les échanges de messages.

Pareillement, l'annonce des attributs peut être source de problèmes. Chaque attribut n'est déclaré qu'une seule fois, ce qui pose un problème de robustesse si ce message est perdu. Il nous faudra donc considérer des mécanismes de renouvellement des déclarations et des abonnements, afin de conférer au système plus de robustesse et de flexibilité.

Mais au final, on constate que le système proposé fonctionne globalement, bien qu'il requiert encore de nombreux efforts de développement. Il permet de mettre en œuvre le concept de pilotage par le produit que nous avons énoncé au premier chapitre. La modularité des agents utilisés permet de représenter une hybridation à travers le produit, comme nous l'avons esquissé dans ce dernier cas d'étude. Cette architecture permet aussi de représenter un fonctionnement hiérarchique, puisque les centres de décision peuvent agir à la fois sur les agents produits et les agents ressources.

3.4 Conclusion

Ce chapitre a présenté la définition, le développement et la validation d'un ensemble de composants logiciels orientés agents facilitant la réalisation d'un système de décision, en particulier d'un système de pilotage par le produit.

Le premier apport de cet outil est de permettre l'expérimentation de divers scénarios d'allocation de la décision afin de pouvoir comparer un contrôle par le produit avec d'autres modes de prise de décision. Les composants logiciels qui ont été développés constituent une contribution à la construction, un outil accessible de recherche et développement sur le contrôle par le produit (et plus généralement sur le contrôle distribué). Chacun de ces composants met en œuvre des techniques connues, mais leur mise en commun dans une architecture cohérente et ouverte permet à l'expérimentateur de se libérer d'une partie du fardeau de l'implémentation. L'accessibilité et la facilité d'utilisation de ce système reste toutefois à améliorer, grâce au développement d'interfaces graphiques, à la simplification des procédures de codage des règles, et en améliorant la stabilité et la robustesse du système multi-agents. Il serait aussi souhaitable de développer des outils spécifiques d'analyse du comportement et des communications des agents.

Le second apport de ce travail de définition et de développement est de préciser la manière dont un système piloté par le produit pourrait être concrètement implémenté. Nous retenons en particulier l'utilisation des mécanismes de notification (protocol FIPA-SUBSCRIBE), qui est bien adaptés au rôle de déclencheur de la prise de décision que nous souhaitons conférer au produit.

On peut enfin discuter des limites de cet outil, qui n'est somme toute qu'un prototype, et des perspectives de son développement. D'abord, des tests plus complets et réalistes sont requis. Ceux-ci permettront de valider les performances du système multi-agents dans des situations complexes où de nombreux agents doivent interagir(*scalability*), et d'améliorer sa stabilité et sa maturité.

Ensuite, certaines améliorations sont nécessaires. Les améliorations d'accessibilité ont déjà été évoquées ; on peut aussi envisager une plus grande utilisation des extensions de JADE, en particulier l'extension de gestion de la sémantique[26], et de gestion des sociétés d'agents[27]. Ces extensions aurai pu simplifier nos développement, mais n'étaient pas encore publiées au moment du développement des composants multi-agents.

Enfin, d'autres travaux plus fondamentaux sont requis.

- Il s'agit en premier lieu d'introduire la persistance des agents. En effet, chaque type d'agent (par exemple chaque type de produit) doit être enregistré sous la forme d'un fichier XML. Chaque fichier doit actuellement être édité individuellement, ce qui n'est pas gérable lorsque la diversité des produits est grande. Il nous faudrait donc développer un mécanisme permettant de générer à la volée ces configurations, depuis une base de données techniques par exemple. L'holon-produit de l'architecture PROSA pourrait être la base de ce développement.
- En second lieu, Il est nécessaire de préciser et d'éclaircir la manière dont sont gérés les échanges de messages entre agents, d'un point de vue sémantique. Pour ne pas compliquer inutilement la mise au point du système multi-agents, on a fait abstraction de ces questions, en basant les échanges de contenus sur des objets java sérialisés. Cette solution devrait faire place au développement d'un système de messagerie plus ouvert, basé sur une ontologie bien définie. Mais s'il est évident qu'il faut définir une ontologie correspondant aux mécanismes

[26]Semantics disponible sur http ://jade.tilab.com développé par France telecom R&D, disponible depuis juillet 2005

[27]ASCML développé par Aachen university of technology, disponibles à la même adresse

d'échange d'attributs entre agents, la manière de définir la sémantique des valeurs de ces attributs est moins évidente. On peut envisager d'utiliser une ontologie pour la production manufacturière [Leitão et Restivo, 2005], ou alors choisir d'adapter une norme comme IEC 62264 [ISO/IEC, 2004], on encore choisir de laisser la définition de cette sémantique au concepteur du système de contrôle.

Chapitre 4

Évaluation d'une architecture de pilotage par le produit sur un cas industriel

Ce chapitre présente une étude expérimentale comparant à un algorithmes de pilotage centralisé divers algorithmes de pilotage par le produit, permettant une plus ou moins grande autonomie dans la prise de décision locale.
Un protocole expérimental, appliqué à un cas d'étude industriel, permet de comparer ces diverses organisations du système de pilotage, en mesurant pour chacune d'elle les effets de facteurs perturbant sur des critères de productivité.
Ces mesures sont conduites en utilisant un modèle d'émulation de l'atelier considéré, permettant d'effectuer un nombre significatif d'expériences.
Les résultats montrent que le pilotage par le produit, permettant l'hybridation des approches centralisées et distribuées de prise de décision, conduit à de meilleures performances qu'une organisation classique, ces performances dépendant cependant de la qualité des algorithmes de décision distribuée.

4.1 Introduction

4.1.1 Démarche expérimentale

Ce chapitre présente une série d'expériences permettant de mettre en œuvre et de valider certains aspects de l'architecture de pilotage par le produit présentée au troisième chapitre. Les expériences sont menées sur un cas industriel particulier, servant de banc d'essais, à l'aide du dispositif d'évaluation par émulation présenté au deuxième chapitre.

Nous nous focalisons, dans nos expériences, sur la validation de l'une des principales fonctions du système de pilotage par le produit. Il s'agit de la capacité à *enrichir la prise de décision distribuée*, grâce à des annotations du produit, correspondant aux décisions centralisées. Cette fonction correspond en fait à la réalisation du couplage entre décisions centralisées et décisions distribuées. Nous cherchons donc à répondre à l'interrogation suivante : « Dans quelle mesure les informations portées par le produit sont-elles pertinentes dans une prise de décision distribuée ».

Ce questionnement porte sur deux points :

- d'une part, sur la manière dont les informations portées par le produit peuvent être prises en compte dans les procédures de décision distribuée,
- d'autre part, sur le fait que *le produit* soit le porteur de ces éléments d'information.

Le premier point est relatif aux *procédures* de décision. Il s'agit de comparer la prise de décision enrichie par les informations des produits, avec d'autres procédures de prise de décision. La pertinence des informations portées par le produit découle d'une évaluation des performances opérationnelles relatives à chaque système de contrôle.

Le second point est intimement lié à des problématiques de transmission et de stockage de l'information. Il s'agit de montrer, par exemple, en quoi l'utilisation de produits *annotés* comme vecteur d'information simplifie le routage des informations vers les acteurs concernés, ou encore en quoi l'association d'informations aux produits peut renforcer la cohérence du système d'information.

La démarche d'expérimentation consiste donc en deux étapes, la première orientée vers les procédures de décision (le traitement de l'information), la suivante orientée vers le stockage et le transfert de l'information.

Dans ce chapitre, nous traiterons des procédures de décision. Pour simplifier la réalisation des expériences, nous ferons abstraction des problèmatiques de transfert et de stockage des informations. En revanche, ces dernières seront étudiées au chapitre suivant.

4.1.2 Utilisation du système multi-agents

Il nous parait important de souligner dès maintenant que la plateforme multi-agents ne sera pas utilisée en tant que telle dans les expériences présentées dans ce chapitre. Ce choix s'appuie sur deux considérations.

D'une part, le principal apport de l'infrastructure multi-agents est relatif à la manière de transmettre et de stocker l'information. Si notre objectif est focalisé sur la nature des informations portées par le produit, ainsi que sur les procédures de décision qui utilisent ces informations, les problèmes liés à la manière de stocker et transmettre ces informations sont secondaires. Il importe donc peu –lors de cette première étape– que les données relatives aux produits soient distribuées dans de nombreux agents, ou quelles soient mémorisées dans une base centrale.

D'autre part, l'utilisation d'un système multi-agents a un coût important. Celui-ci repose d'abord dans la difficulté accrue de développer les régles de décisions devant être mise en œuvre. Ensuite, si le contrôle de la production est réalisé par un programme externe, l'émulateur doit fonctionner en mode « temps réel »[28], ce qui allonge considérablement la durée des essais, et limite donc leur nombre.

Les expériences présentées dans ce chapitre sont donc basées sur l'hypothèse que *l'ensemble de l'information disponible peut être connue de tout acteur à tout moment (sans délai)*. Les procédures de décision seront codées dans la couche VB, ce qui permettra de les embarquer dans le logiciel de simulation avec le modèle d'émulation. Les simulations pourront donc être réalisées par événements discrets.

Nous sommes bien conscients que l'hypothèse que nous avons faite est importante, et discutable. Toutefois elle constitue une première étape, permettant d'obtenir des résultats initiaux sur le pilotage par le produit. Elle permet de plus de nous placer dans des conditions expérimentales favorables, où il est possible de :

- changer facilement les algorithmes de décision et les informations portées par les produits,
- faire varier les facteurs influant sur le modèle d'émulation ou le système de contrôle,
- faire de nombreux essais

Les problèmatiques de stockage et de transfert de l'information ne sont néanmoins pas négligeables ; elles seront étudiées à l'aide d'une plateforme physique d'essai au chapitre suivant.

[28] Dans le mode « temps réel », l'avancement du temps simulé se fait proportionnellement à l'horloge de l'ordinateur.

4.1.3 Plan du chapitre

Dans nos expériences, il s'agit de simuler un système de production virtuel, que nous appelerons « procédé », composé d'un modèle d'émulation d'un atelier, et d'un système de pilotage. Ce système est configuré à l'aide d'un ensemble de paramètres, qui restent les mêmes pour toutes les expériences. Il prend en entrée certains facteurs, choisis par l'expérimentateur, dont on veut mesurer l'effet sur le comportement du procédé. Cette mesure est réalisée grâce à l'évaluation d'un ensemble d'indicateurs de performance (figure 4.1).

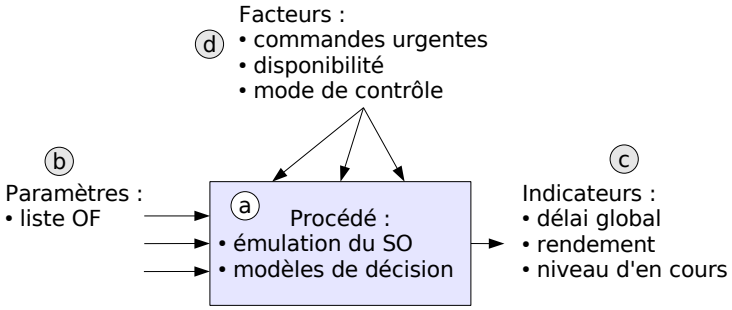


FIG. 4.1 – Principe de la démarche expérimentale. Nous décrirons d'abord le cas sur lequel nous allons expérimenter (a), puis les paramètres des expériences (b), les indicateurs de performance (c) et enfin les facteurs que nous étudierons (d).

Parmi les facteurs, on distingue le mode de pilotage. En effet, le système de pilotage peut être soit un système contrôlé par le produit, soit un système centralisé plus classique. La mesure des performances est donc réalisée en comparant les performances relatives des deux systèmes. En effet, il n'est pas possible de réaliser une mesure absolue des performances ; nous devons donc procéder par *double pesée*.

Nous présenterons donc d'abord les différents éléments de notre dispositif expérimental : modèle de l'atelier, paramètres utilisés, indicateurs de performance, facteurs étudiés. Nous insisterons sur les modèles de pilotage utilisés, et en particulier sur la vérification que le système de pilotage par le produit est correctement codé.

Ensuite, nous donnerons le protocole des expériences, et enfin, nous présenterons leurs résultats.

4.2 Protocole expérimental

Structure de l'atelier modélisé

Nous choisissons d'étudier le cas d'un équipementier automobile, rapidement présenté au deuxième chapitre, pour effectuer les expériences. La production comporte deux étapes : un produit semi-fini est d'abord réalisé ; ensuite une phase d'assemblage permet d'obtenir le produit fini.

L'atelier comporte quatre cellules : la première est dédiée à la réalisation des produits semi-finis (dénommée SF), les trois autres à l'assemblage final (dénommées FA, FB et FC). Ces trois cellules d'assemblage fonctionnent indépendamment les unes des autres, c'est à dire qu'elle peuvent chacune traiter des produits de types différents. Pour être plus précis, nous avons fait l'hypothèse que les lignes F ne peuvent simultanément produire des produits de même référence.

Enfin, l'atelier comporte un stock de produits semi-finis, alimenté par la cellule SF, et consommé par les cellules F.

Le choix de cas a été motivé par les problèmes industriels qui s'y posent. Ceux-ci sont essentiellement liés aux changements de séries sur la cellule SF, c'est à dire à la manière de fractionner les ordres de fabrication en lots. Des travaux antérieurs [Pannequin et Thomas, 2004] ont montré qu'il était possible de piloter cet atelier de manière centralisée ou distribuée.

Quant aux paramétrages du modèle d'émulation, chaque cellule est capable de fonctionner selon 12 programmes différents[29].Le tableau 4.1 présente des temps de cycles, qui correspondent à des données réelles de l'entreprise.

De plus, nous définissons des temps de changements de série suivant une loi normale de moyenne 20 minutes, avec un écart-type de 10 minutes. Ces durées sont indépendantes des références, et sont les mêmes pour chaque cellule. Cette distribution des valeurs des temps de réglage a été choisie compte tenu des données réelles dont nous disposions et des objectifs de nos essais.

Les temps de transport en entrée ou en sortie des stocks sont fixés à une demi minute. Les pièces sont transférées à l'unité.

référence du produit	temps de cycle (SF)	temps de cycle (F)
1	1.34	3.6
2	1.06	3.49
3	1.14	3.7
4	1.14	3.57
5	1.17	3.54
6	1.13	3.75
7	1.22	4.09
8	1.22	3.69
9	1.11	4.03
10	1.23	3.62
11	1.23	3.75
12	1.11	3.96

Tab. 4.1 – Temps de cycle utilisés dans notre modèle (en minutes)

4.2.1 Paramètres

Le paramètre principal que nous appliquons au modèle est la liasse d'ordres de fabrication devant être effectués.

Les quantités à réaliser pour chaque type de produit sont présentées au tableau 4.2. La simulation ne s'arrête que lorsque tous les produits demandés ont été effectués.

4.2.2 Indicateurs de performance

Compte tenu des objectifs de ces essais rappelés au début de ce chapitre, la performance du système de pilotage est évaluée en regard de la productivité du système de production. En effet,

[29]En examinant les temps de cycle on constate que certains programmes sont identiques, et auraient pu avoir été agrégés (par exemple P3 et P4, pour SF). Toutefois nous avons maintenu un peu artificiellement ces distinctions, pour faciliter la mise en œuvre des algorithmes de pilotage, puisque on a ainsi autant de programmes que de produits différents.

référence du produit	quantité à réaliser	date de fin souhaitée
1	1775	7200
2	192	3600
3	15	3600
4	832	7200
5	448	7200
6	18	7200
7	600	7200
8	192	7200
9	240	7200
10	640	7200
11	128	7200

TAB. 4.2 – Production devant être réalisée

dans un contexte d'évaluation de la pertinence de donner au produit des fonctionnalités nouvelles pour quantifier les gains de productivité, nous utilisons quelques indicateurs de performances classiques.

Taux de rendement synthétique C'est le rapport entre le temps utile et le temps total. Cet indicateur est calculé pour chaque cellule de production, en utilisant les outils de mesures de performance fournis par l'environnement de simulation.

Délai global C'est la durée totale qui est nécessaire pour effectuer la production des ordres de production. Il est calculé dans le simulateur comme la date de fin du dernier ordre, la simulation commençant à l'instant zéro.

Occupation moyenne des stocks C'est l'occupation moyenne du stock de produits semi-finis, toutes références confondues. Cette quantité est évaluée tout au long de la simulation, puis l'occupation moyenne est calculée par le simulateur.

De plus, chaque simulation est représentée par un diagramme de gantt, ce qui permet d'analyser plus finement le comportement du système de pilotage.

4.2.3 Facteurs étudiés

Le premier facteur sur lequel on agit sera le mode de pilotage. Nous pouvons de cette manière sélectionner le type de pilotage (contrôle centralisé ou pilotage par le produit) devant être utilisé dans la simulation.

Les autres facteurs permettent d'évaluer la performance du pilotage par le produit dans un environnement perturbé. En effet, l'un des avantages attendus du pilotage par le produit est sa robustesse dans un environnement incertain et perturbé. Nous déterminons les facteurs de manière à pouvoir évaluer cette caractéristique de robustesse : les facteurs correspondent aux niveaux de perturbation subis par le système, leurs effets sur les indicateurs de performance correspondent à une mesure de robustesse.

Nous retenons deux types de perturbations :

- des perturbations du système physique (*process*) ; nous représentons en particulier les indisponibilités des ressources (quelle que soit leurs causes).
- des perturbations de gestion (*business*), représentées par les changements de la quantité d'un ordre de fabrication, ou par le raccourcissement de la date de fin souhaitée.

Modalité	Loi d'apparition des pannes	Durée des pannes
0	-	-
1	expo(600)	norm(45,6)
2	expo(3000)	norm(360,180)

TAB. 4.3 – Modalités du facteur d'indisponibilité des ressources (en minutes).

Nous traitons donc de trois facteurs : ils sont nommés par F_{pp}, F_{pb} et F_{ctrl} désignant respectivement les perturbations du système physique (*process*), de la gestion (*business*), ou le choix du type de contrôle.

Disponibilité des ressources

Nous définissons donc un facteur d'indisponibilité, ayant trois modalités (table 4.3) :
– pas d'indisponibilité des ressources.
– des indisponibilités courtes et relativement fréquentes.
– des indisponibilités longues et peu fréquentes, qui s'ajoutent aux précédentes.
Les caractéristiques des modalités du facteur (durée et fréquence des indisponibilités) ont été déterminées d'après les données de maintenance du système industriel étudié.

Le choix d'une loi exponentielle pour modéliser les temps de bon fonctionnement est justifié par l'hypothèse que le taux de panne reste constant sur la période considérée. En effet, nous simulons environ une semaine de fonctionnement, ce qui est une petite durée par rapport à la durée de vie de la machine. On ne prend donc pas en compte les phénomènes liés au vieillissement.

On peut remarquer que nous ne traitons pas exhaustivement des facteurs pouvant perturber le système physique. En particulier, le modèle ne simule pas de rebuts, ni d'erreur dans le routage des flux.

Modification des ordres de production

Pour soumettre le système de pilotage par le produit que nous souhaitons évaluer à un autre type de perturbations, nous choisissons d'introduire des modifications dans les ordres de production devant être réalisés. Il nous faut prendre en compte deux paramètres :
– la quantité supplémentaire devant être réalisée
– la date de fin souhaitée
Nous traitons de deux cas de figure : d'une part l'ajout d'un ordre urgent, mais correspondant un faible volume à réaliser. D'autre part la modification d'un ordre peu urgent, par une augmentation significative de la quantité devant être réalisée. Pour déterminer les quantités, on se base sur la taille standard des lots, Q_{eco} , égale à 250 unités. Ainsi, une quantité significative sera égale à $Q_{eco}/2$ (125), tandis qu'une quantité faible sera égale à $Q_{eco}/10$ (25).

Pour mettre en œuvre la première modalité, on augmente la quantité correspondant aux produits de référence 1, car ils sont produits tout au long de la période simulée. La date de fin souhaitée est donc la fin de la période simulée. Pour la seconde modalité, on crée un nouvel ordre (référence 12). Pour modéliser l'urgence, le délai imparti pour réaliser cet ordre est égal à 1,5 fois le temps de travail prévisionnel. Les caractéristiques des modalités du facteur sont synthétisées table 4.4.

L'événement déclenchant la perturbation se produit en moyenne à t=4000 (c'est à dire un peu après la moitié de la période simulée) en suivant une loi exponentielle. À la réception de cet événement, l'ordre correspondant à la modalité est créé, et l'ordonnancement est mis à jour.

Modalité	Urgence	Quantité
1	forte : fin souhaitée à 1,5 (temps de prod)	$Q_{eco}/10 = 25$
2	faible : fin souhaitée à 7200 min	$Q_{eco}/2 = 125$

TAB. 4.4 – Modalités du facteur de modification des ordres de production.

4.3 Modèles de prise de décision

4.3.1 Problème de décision à résoudre

Comme nous l'avons vu précédemment, la production est réalisée en deux étapes. D'une part, la fabrication du produit semi-fini est réalisée sur une cellule de production, avec un rythme de production de l'ordre d'une unité par minute. D'autre part, trois lignes parallèles et indépendantes réalisent l'assemblage final, avec un rythme de l'ordre d'une unité toutes les trois minutes. Le problème qui se pose sur le terrain est donc lié à la gestion du stock de produits semi-finis. En effet, alors qu'une seule référence est produite par la ligne en amont, trois références sont consommées simultanément par les lignes en aval. Il s'agit donc de décider des changements de séries, en tenant compte de l'occupation des stocks, et en minimisant le nombre de changements de série, qui prennent du temps.

Les ordres de fabrication correspondant à de grandes quantités doivent donc être fractionnés en lots, dont la taille doit être égale à une taille minimum. Nous choisissons comme taille de lot minimum 250 unités, correspondant à une logique de quantité économique. La liste des lots (table 4.5) est obtenue à partir de la liste des OF. Le nombre de lots créés à partir d'un OF est calculé en divisant la quantité de l'OF par la taille standard.

4.3.2 Procédure de décision centralisée

La procédure de décision centralisée se fait en deux étapes :
- d'abord on ordonnance les lots sur la cellule SF,
- ensuite, on ordonnance le passage des lots sur l'une des cellules F, en se basant sur ce premier ordonnancement.

L'ordonnancement sur SF se fait en fonction du critère de ratio critique (rc). Celui-ci se calcule comme le rapport du temps nécessaire à la réalisation de la tâche et du temps restant avant livraison.

$$rc = \frac{Q_{restante} \times T_{cycle}}{D_{fin} - D}$$

On ordonnance donc les lots en fonction de ce critère, grâce à un algorithme de tri standard.

Ensuite, on décale les ordres, de manière à assurer que pour tout groupe de trois lots consécutifs, il n'y ait jamais la même référence à produire. Ce second critère permet de maximiser l'utilisation des lignes F (on rappelle que l'on a fait l'hypothèse que les lignes F ne peuvent fonctionner simultanément sur le même ordre).

L'ordonnancement des lots sur les lignes FA, FB et FC se fait en les attribuant à la première ligne disponible.

Le calcul se déroule de la manière suivante :

On initialise les dates de fin des tâches (ou job) attribués aux lignes F.
$dateFin_i = t + tempsRestant(job_i)$
avec job_i l'ordre actuellement en cours sur la ligne F_i

Ordre de fabrication	numéro de lot	quantité	date de fin
OF1 (Q=1775)	11	251	1029
	12	251	2057
	13	251	3086
	14	251	4114
	15	251	5143
	16	251	6171
	17	251	7200
OF2 (Q=192)	21	192	3600
OF3 (Q=15)	31	15	3600
OF4 (Q=832)	41	277	2400
	42	277	4800
	43	278	7200
OF5 (Q=448)	51	224	3600
	52	224	7200
OF6 (Q=18)	61	18	7200
OF7 (Q=600)	71	300	3600
	72	300	7200
OF8 (Q=192)	81	192	7200
OF9 (Q=240)	91	240	7200
OF10 (Q=640)	101	213	2400
	102	213	4800
	103	214	7200
OF11 (Q=128)	111	128	7200

Tab. 4.5 – Fractionnement en lots des ordres de production.

On initialise la date de disponibilité
$dateDispo = t + tempsRestant(job_{sf})$

Pour chaque *ordre* de *listeOrdo*,
faire
Soit n le numéro de la ligne pour laquelle la date de fin est la plus petite.
on programme cet ordre à la ligne F_n.
$date_finin[n] = min(datefin[n], date_dispo) + temps_restant(ordre)$
$date_dispo = date_dispo + temps_restant(ordre)$
Fait

Par la suite, le pilotage est réalisé en effectuant les ordres dans l'ordre programmé. Le réglage se fait dès la fin de l'ordre précédent, et autant de produits que spécifiés doivent être réalisés avant que l'ordre courant ne soit considéré terminé. L'ordre suivant est alors commencé. Par conséquent, la politique de pilotage par rapport aux indisponibilités est l'attente : si la cellule de production est indisponible ou en rupture d'approvisionnement, la production est décalée, pour reprendre à la fin de l'indisponibilité.

4.3.3 Procédure de décision distribuée

Modélisation de l'infrastructure de pilotage par le produit

À chaque produit est associé un objet. Les attributs de cette classe d'objets représentent donc les informations portées par les produits. Parmi celles-ci on trouve un attribut d'état, et l'identifiant du produit émulé, qui permettent ensemble de synchroniser l'état du produit informationnel et du produit physique émulé.

De plus, les produits portent des attributs permettant d'enrichir la prise de décision distribuée avec des informations résultant des décisions centralisées. Ainsi, chaque produit porte un attribut de priorité, ainsi qu'un attribut correspondant à la route initialement prévue.

Les produits sont dès leur création enregistrés dans une structure (un *registre* des produits) dont le rôle est de réagir aux événements de l'émulation en faisant évoluer l'attribut d'état des produits, et de fournir la liste des produits se trouvant dans un certain état. Cette dernière fonction permet aux centres de décision de connaître la liste des produits dont l'état requiert un traitement.

Centres de décision

Il existe deux types de centres de décision, dont les rôles sont différents : les premiers effectuent la conduite des ressources en se basant de manière autonome sur les informations portées par le produit. Les seconds ont des fonctions dans le pilotage des flux de matière (figure 4.2).

Ainsi, chaque cellule de l'atelier (SF, FA, FB, FC) est dotée d'un centre de décision suivant un algorithme simple. Celui-ci réagit aux événements signalant l'arrivée d'un produit ou la libération d'une ressource. Les informations du produit (en particulier sa référence) sont alors lues, ce qui permet de connaître l'opération à réaliser. Enfin, les commandes de début de cycle (et éventuellement de réglage) sont envoyées vers la ressource émulée.

Le pilotage des flux se fait par l'intermédiaire de certains éléments du système émulé :

– l'introduction des produits devant la cellule SF permet de contrôler les changements de série sur cette cellule,

– la transformation d'espace du stock de produits semi-finis vers les cellules F permet de contrôler leur production.

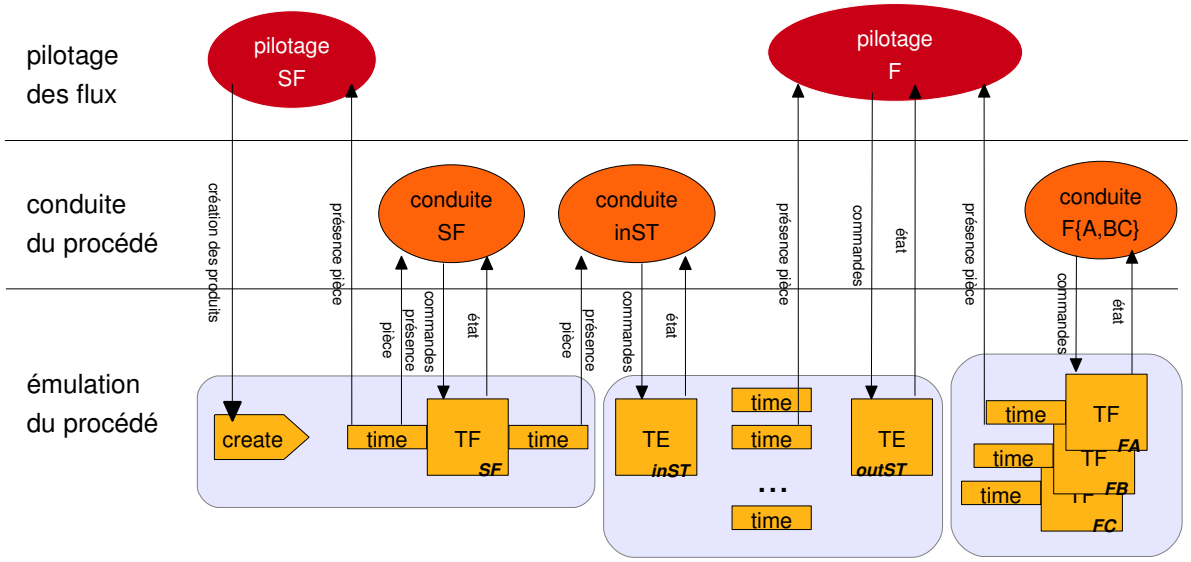


FIG. 4.2 – Vue schématique du modèle d'émulation et des centres de décision associés

Algorithme simple de pilotage par le produit (PP-contraint)

La manière la plus simple de mettre en œuvre le pilotage par le produit est de baser uniquement les décisions de gestion des flux sur les informations liées aux produits, en particulier sa priorité. En théorie, si on choisit les produits par ordre de priorité croissante, il est possible de reconstruire l'ordonnancement centralisé.

L'algorithme ci-dessous présente donc le choix, en SF, du produit le plus prioritaire.

Soit L l'ensemble des produits pouvant être lancés
Soit priorité une fonction de L dans $[0;1]$ telle que
si p_1 est ordonnancé avant p_2 alors priorite(p_1) < priorite(p_2)

$m \leftarrow 2$
Pour p dans L faire
si priorite$(p) < m$ alors
$m \leftarrow$ priorite(p)
$p_{choisi} \leftarrow p$
fin si
fait
créer p_{choisi}

Dans ce mode de décision, les centres de décision ne présentent aucune autonomie, car leurs comportements sont entièrement conditionnés par les informations portées par les produits. Par la suite l'appellera « pilotage par le produit contraint » (ou CCP-contraint).

4.3.4 Algorithme plus élaboré de pilotage par le produit (PP-autonome)

Alternativement à l'implémentation simple que nous venons de présenter, on peut doter les centres de décision locaux d'une certaine autonomie vis-à-vis des informations des produits. Ainsi, la prise de décision tiendra compte à la fois de ces informations, mais aussi de l'état courant de l'environnement local du centre de décision, comme le ferait un opérateur intelligent. Le mode de décison est donc appelé pilotage par le produit avec autonomie, ou PP-autonome.

L'algorithme ci-dessous a été utilisé pour contrôler la cellule SF. Il cherche à imiter le comportement que pourrait avoir un opérateur, en se basant sur les niveaux de stock aval, et sur la taille de lot, ainsi que sur les informations de priorité portées par les produits.

Soit L l'ensemble des produits pouvant être lancés.
Soit R l'ensemble des références des produits.
Soit s_{min} et s_{max} les valeurs seuils pour chaque référence dans le stock de produits semi-finis.
Soit r_{SF} la référence sur laquelle le cellule SF est réglée.

Soit ref une fonction de L dans R telle que
  $\mathsf{ref}(p)$ soit la référence du produit p
Soit priorité une fonction définie comme précédemment
Soit stock une fonction de L dans $\mathbb{N}^+$ telle que
  $\mathsf{stock}(p)$ soit la quantité de produits semi-finis de référence $\mathsf{ref}(p)$ stockés.
Soit $\mathsf{opportunité}$ une fonction de L dans $[0;1]$ telle que
  $\mathsf{opportunité}(p) = 0$ si $\mathsf{stock}(p) < s_{min}$,
  $\frac{stock(p)-s_{max}}{s_{max}-s_{min}}$ si $s_{min} < \mathsf{stock}(p) < s_{max}$,
  1 si $\mathsf{stock}(p) < s_{max}$.

si changerRef alors
  $p_{choisi} \in L$ est tel que l'expression
    $\alpha * \mathsf{priorité}(p) + (1-\alpha)(\mathsf{opportunité}(p))$
  est minimale (avec $\alpha = 0.5$).
sinon
  Soit $L_r = \{p \in L, \mathsf{ref}(p) = r_{SF}\}$
  si L_r n'est pas vide alors
  $p_{choisi} \in L_r$ est tel que l'expression
    $\mathsf{priorité}(p)$ est minimale.
  sinon
  $p_{choisi} \in L$ est tel que l'expression
    $\alpha\mathsf{priorité}(p) + (1-\alpha)(\mathsf{opportunité}(p))$
  est minimale (avec $\alpha = 0.9$).
  fin si
fin si

créer p_{choisi}

4.3.5 Plan d'expériences

Nous avons réalisé les expériences en suivant un plan complet, c'est à dire que toutes les combinaisons des facteurs ont été prises en compte.

Pour étudier les interactions, nous utilisons une table de Tagushi ($L_2 7$).

L'exécution de ces simulations a été réalisée en utilisant le logiciel Arena Process Analyser. Ce logiciel permet de définir et d'exécuter facilement notre plan d'expériences, sans manipulation inutile. Pour chaque combinaison des facteurs, 15 simulations ont été exécutées, afin de fournir des résultats dans un intervalle de confiance raisonnable. Chaque simulation prend un peu moins de cinq minutes pour s'exécuter. La durée totale de simulation a été d'environ trente heures.

4.4 Résultats expérimentaux

4.4.1 Vérification des modèles de prise de décision

Les premiers résultats expérimentaux peuvent servir à valider notre implémentation des procédures de décision. Cette validation est nécessaire, car notre manière d'implémenter le pilotage par le produit peut introduire un biais dans l'évaluation. Ce biais peut artificiellement réduire les performances du système de contrôle, si les algorithmes de décision ou les informations portées par le produit sont incorrectes, ou mal utilisées. Il nous faut donc nous assurer de la correction du modèle de contrôle. Cette procédure, que l'on pourrait qualifier « d'étalonnage » dans la mesure où elle nous permet de réduire un biais, repose sur l'évaluation des performances des divers systèmes de contrôle dans une situation non perturbée (table 4.6).

type de contrôle	délai global	en-cours	taux de rendement
centralisé	6985,57	250,78	88,70%
PP-contraint	6995,53	260,55	88,59%
PP-autonome	7185,76	141,29	86,23%

TAB. 4.6 – Performance des divers modes de contrôle dans une situation non perturbée

On constate que le niveau des indicateurs de performance est très sensiblement le même entre le pilotage centralisé et le pilotage par le produit contraint. En effet l'écart sur le délai global entre le pilotage centralisé et le pilotage PP-contraint est de 0,14%, tandis que les écarts par rapport au niveau d'en cours et au taux de rendement sont respectivement de 4% et 0,13%. Notre implémentation du pilotage par le produit conduit donc au même comportement que le pilotage centralisé. Nous pouvons donc affirmer qu'elle est satisfaisante[30]. Ce résultat valide aussi le choix des données à embarquer dans le produit, puisqu'il prouve qu'elles sont suffisantes pour reconstruire l'ordonnancement global.

[30]Ce résultat n'a naturellement pas été obtenu du premier coup. Il s'agit en fait d'un développement itératif, dans lequel le test présenté a été réalisé (sans succès) de nombreuses fois, avant de valider notre implémentation

Facteurs			interaction		Indicateurs (valeurs moyennes)					
Pert. du process	Pert. du business	Mode de pilotage	PP-ctr	PB-ctr	en-cours	délai global (min)	TRS FA (%)	TRS FB (%)	TRS FC (%)	TRS SF (%)
0	0	0	0	0	251 (±3)	6986 (±18)	92 (±0,24)	92 (±0,24)	82 (±0,2)	89 (±0,23)
1	0	0	1	0	263 (±12)	7658 (±97)	84 (±1,05)	84 (±1,06)	74 (±0,93)	81 (±1,01)
2	0	0	2	0	340 (±66)	9304 (±428)	70 (±3,13)	70 (±3,17)	62 (±2,82)	67 (±3,03)
0	1	0	0	1	260 (±9)	7406 (±143)	87 (±3,22)	86 (±3,07)	80 (±2,1)	84 (±1,54)
1	1	0	1	1	275 (±18)	7973 (±137)	81 (±2,35)	79 (±2,76)	75 (±1,64)	78 (±1,29)
2	1	0	2	1	360 (±61)	9407 (±502)	68 (±3,93)	71 (±4,46)	62 (±4,07)	67 (±3,69)
0	2	0	0	2	263 (±11)	7444 (±178)	88 (±4,10)	88 (±2,28)	80 (±2,61)	86 (±1,95)
1	2	0	1	2	276 (±17)	8039 (±191)	83 (±2,99)	80 (±2,92)	74 (±1,71)	79 (±1,82)
2	2	0	2	2	360 (±74)	9328 (±384)	71 (±3,90)	72 (±3,67)	63 (±2,77)	68 (±2,88)
0	0	1	1	2	141 (±5)	7186 (±46)	88 (±0,74)	87 (±1,08)	83 (±1,14)	86 (±0,56)
1	0	1	2	2	170 (±14)	7693 (±75)	84 (±2,68)	81 (±2,47)	77 (±1,51)	80 (±0,79)
2	0	1	0	2	224 (±49)	8975 (±420)	71 (±5,42)	70 (±3,63)	68 (±4,15)	69 (±3,05)
0	1	1	1	0	154 (±5)	7302 (±34)	93 (±0,93)	83 (±1,69)	80 (±1,39)	85 (±0,4)
1	1	1	2	0	182 (±16)	7874 (±90)	84 (±2,29)	79 (±2,31)	74 (±2,69)	79 (±0,9)
2	1	1	0	0	239 (±41)	8998 (±564)	72 (±4,58)	71 (±4,69)	66 (±5,69)	70 (±3,96)
0	2	1	1	1	154 (±5)	7384 (±52)	92 (±0,58)	85 (±1,17)	81 (±0,82)	86 (±0,59)
1	2	1	2	1	181 (±16)	8093 (±86)	83 (±2,93)	78 (±3,66)	74 (±2,82)	79 (±0,84)
2	2	1	0	1	241 (±44)	9456 (±611)	70 (±4,70)	70 (±5,78)	64 (±4,59)	68 (±3,83)
0	0	2	2	1	261 (±2)	6996 (±20)	92 (±0,32)	92 (±0,3)	82 (±0,32)	88 (±0,26)
1	0	2	0	1	274 (±9)	7509 (±63)	85 (±0,86)	84 (±1,27)	79 (±1,49)	82 (±0,7)
2	0	2	1	1	329 (±50)	9008 (±387)	74 (±2,37)	69 (±4,41)	65 (±4,58)	69 (±2,96)
0	1	2	2	2	295 (±8)	7534 (±78)	92 (±1,69)	85 (±2,73)	72 (±2,68)	83 (±0,85)
1	1	2	0	2	315 (±15)	8098 (±103)	85 (±1,74)	75 (±3,03)	70 (±3,7)	77 (±0,96)
2	1	2	1	2	389 (±42)	9430 (±452)	72 (±4,40)	66 (±3,85)	61 (±4,44)	66 (±3,11)
0	2	2	2	0	295 (±7)	7637 (±51)	90 (±1,74)	85 (±0,89)	74 (±1,94)	83 (±0,54)
1	2	2	0	0	318 (±14)	8092 (±104)	85 (±1,68)	77 (±2,36)	73 (±1,69)	79 (±1)
2	2	2	1	0	384 (±46)	9542 (±450)	73 (±4,29)	67 (±4,13)	61 (±3,6)	67 (±3,25)

Tab. 4.7 – Résultats des simulations. Les valeurs entre parenthèse sont les intervalles de confiance.

4.4.2 Effet des facteurs sur les performances

Les effets des facteurs et de leurs interactions sur les volumes d'en-cours, le délai global et le taux de rendement des ressources sont respectivement présentés en figures 4.3, 4.4 et 4.5.

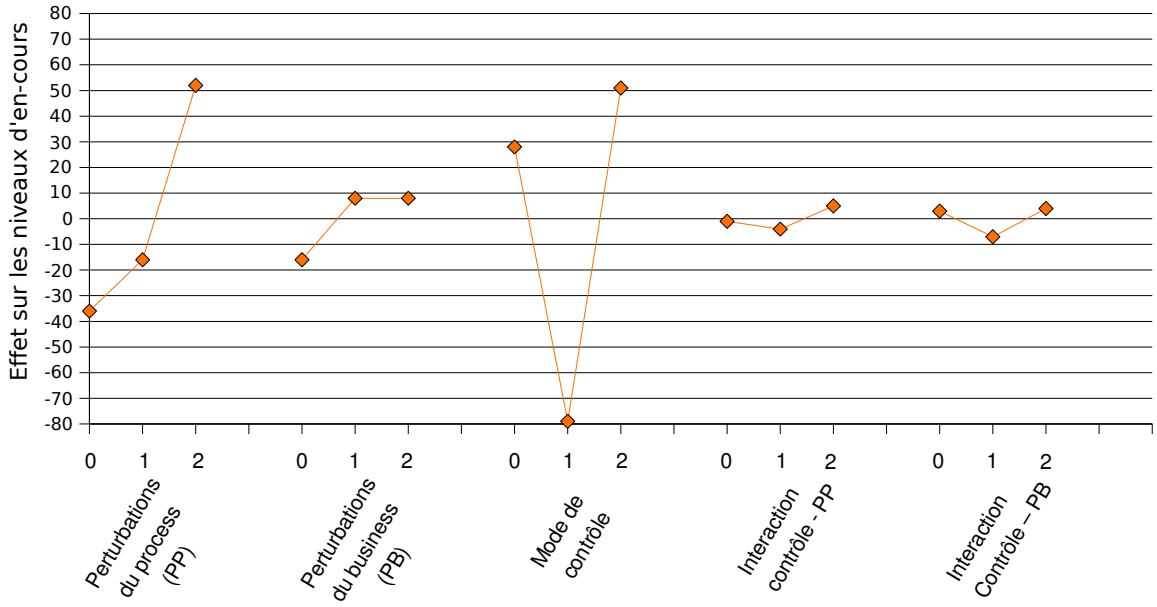


FIG. 4.3 – Effets sur le volume d'en-cours (volume moyen du stock de produits semi-finis)

Dans ces trois graphiques, on constate que les perturbations du process ou du business conduisent à une nette dégradation des performances, comme on pouvait s'y attendre. L'effet du troisième facteur, le mode de pilotage, est beaucoup plus intéressant.

En effet, ce dernier, présenté sur la troisième courbe, montre les performances moyennes de chaque type de contrôle. Au vu de cette courbe, on constate que le pilotage par le produit, lorsqu'il s'exécute dans le cadre d'un prise de décision autonome (PP-autonome, modalité 1), conduit à des niveaux d'encours plus bas que pour le contrôle centralisé (modalité 0), ainsi qu'un délai global légèrement plus court. À l'inverse, le pilotage par le produit montre de moins bonnes performances que le contrôle centralisé lorsque la prise de décision distribuée est contrainte par les produits (PP-contraint, modalité 2). Les effets sur le taux de rendement synthétique sont comparables aux effets sur le délai global.

Les deux dernières courbes présentent les interactions entre les différents types de perturbations et le facteur de contrôle. On constate que les interactions sont relativement faibles puisque la pente des courbes d'effets n'est pas très marquée. Pour pouvoir mieux étudier l'éffet de ces interactions, on choisit de représenter plus finement leurs modalités. Ainsi, neufs modalités sont choisies pour représenter tous les couples envisageables entre le mode de pilotage et le niveau de perturbation. Les trois premières, numérotées de 0 à 2, correspondent au pilotage centralisé ; les trois suivantes au mode de pilotage PP-autonome, et les trois dernières au mode PP-contraint.

La figure 4.6 présente sur le même graphe l'effet sur le délai global du facteur d'interaction pilotage/pertubation du process (PP-Ctrl), et pilotage/perturbation du business (PB-Ctrl).

Le graphe montre que l'effet du facteur de perturbation domine largement l'effet du mode de pilotage, surtout dans de cas de perturbations du process. On peut cependant remarquer que

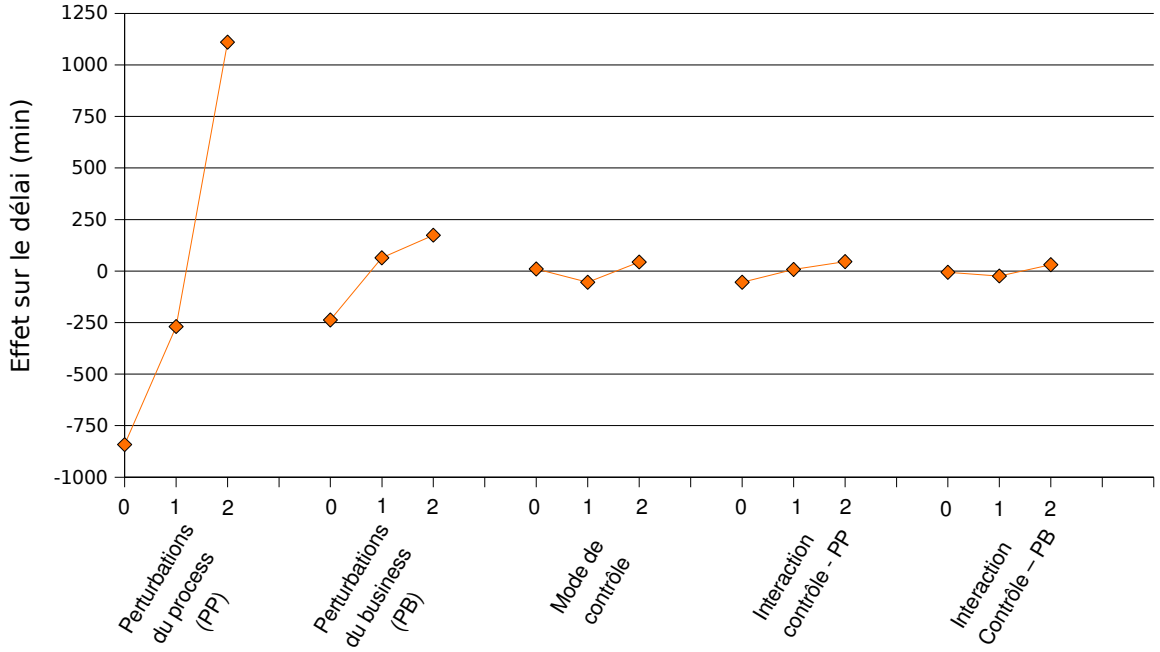


FIG. 4.4 – Effets sur le délai global

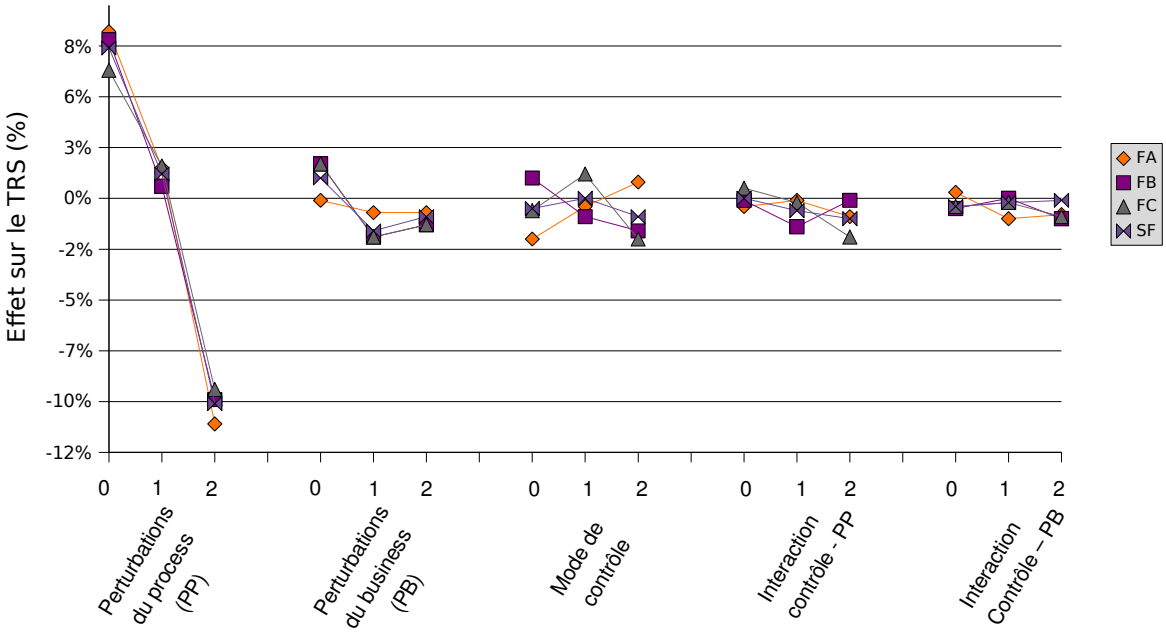


FIG. 4.5 – Effets sur le taux de rendement des ressources

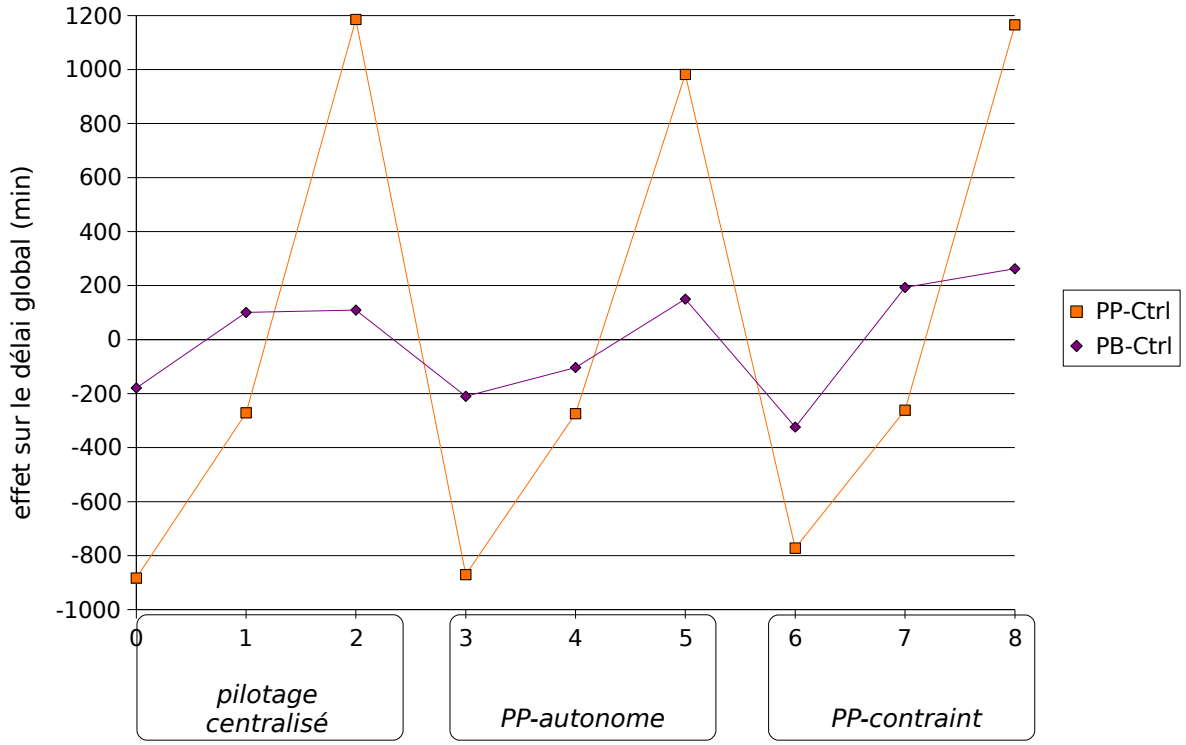


FIG. 4.6 – Effet des facteurs d'interaction mode de pilotage/perturbation sur le délai global

lorsque les perturbations sont les plus importantes (modalités 2, 5 et 8), le pilotage PP-autonome s'avère meilleur. De la même manière, le pilotage par le produit PP-autonome est meilleur pour certains types de perturbations du business.

On ne détecte donc pas une amélioration majeure du pilotage par le produit par rapport à la réduction des délais. Toutefois, Celui-ci ne dégrade pas non plus les performances (par rapport au pilotage centralisé qui sert de référence) y compris dans une situation non perturbée.

Les effets sur les niveaux d'en-cours des deux facteurs d'interaction sont présentés figure 4.7.

Contrairement aux effets sur le délai global, les effets des facteurs d'interaction sur les niveaux d'en-cours dépendent nettement du mode de pilotage. En particulier, le pilotage PP-autonome conduit à une forte réduction de en-cours, tandis que le pilotage PP-contraint conduit à leur augmentation, surtout en cas de perturbations du business.

Ce graphique montre donc l'apport potentiel du pilotage par le produit par rapport à la réduction des en-cours, ainsi que l'importance de la manière de le mettre en oeuvre.

Ces résultats montrent globalement qu'une approche de pilotage par le produit permet de combiner les bonnes performances du pilotage centralisé dans une situation non perturbée, tout en pouvant s'adapter —à la manière d'un système distribué— aux diverses perturbations.

4.4.3 Discussion

Les résultats de notre plan d'expériences montrent la faisabilité du pilotage par le produit. La robustesse du pilotage par rapport aux perturbations, c'est à dire sa faculté à s'adapter à divers types de perturbations, présentée par le mode pilotage PP-autonome est en effet supérieure

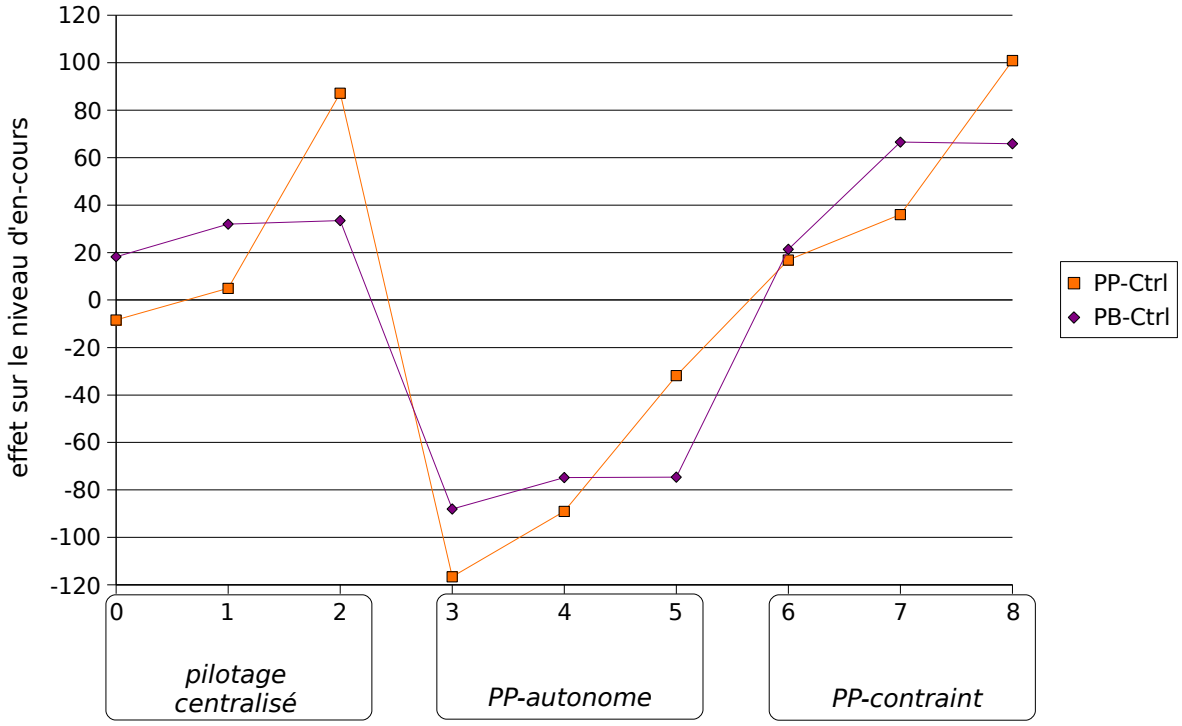


FIG. 4.7 – Effet des facteurs d'interaction mode de pilotage/perturbation sur le délai global

à celle d'un système de pilotage plus classique. Cette évaluation se base sur la moyenne des performances mesurées, conformément au plan d'expériences. Une exploitation plus fine des données expérimentales collectées, en particulier les diagrammes de Gantt relatifs à chaque simulation, peut être envisagée pour affiner l'interprétation des résultats. La quantité de données à gérer (405 traces de simulation) est cependant une limite importante.

On remarque toutefois que les résultats sont très différents en fonction de la nature des décisions locales. Le mode de pilotage par le produit est donc très sensible à l'environnement décisionnel, en particulier à la pertinence des décisions locales.

Lorsque l'autonomie des décisions locales est faible (mode PP-contraint), le pilotage par le produit présente de bonnes performances dans les situations non perturbées, mais sa robustesse est faible (inférieure à la robustesse du pilotage centralisé). Ceci peut s'expliquer parce que les décisions locales ne peuvent reconstruire l'ordonnancement global à partir des données produit que si ceux-ci arrivent dans l'ordre nominal. Par contre, lorsqu'une certaine autonomie est admise (mode PP-autonome), les performances sont légèrement altérées dans les conditions nominales (non perturbées), mais meilleures dans les conditions dégradées.

La manière dont les procédures de décision locale ont été implémentées est sans aucun doute critiquable, en particulier du point de vue de leur « intelligence ». Toutefois, étant donnée la tendance montrée par la comparaison entre PP-autonome et PP-contraint, on peut affirmer que des procédures de meilleure qualité (utilisant des méthodes plus précises d'intelligence *artificielle* ou encore l'intelligence *naturelle* d'un opérateur humain) auraient conduit à de meilleures performances. En effet, des algorithmes plus avancés seraient capables de percevoir le niveau

de perturbation de l'environnement et d'adapter en conséquence leur autonomie vis à vis des données des produits. À l'inverse, lorsque les prises de décisions centralisées et distribuées se font sur des critères diamétralement opposés, on voit mal la manière dont on pourrait les combiner harmonieusement.

Enfin, ces résultats proviennent d'une expérimentation sur un unique cas d'étude. Pour pouvoir les généraliser, on peut envisager de poursuivre les expériences sur ce modèle avec de nouveaux paramètres (d'autres ensembles d'ordres de production à exécuter), ou encore de modéliser d'autres cas. Expérimenter systématiquement sur un ensemble de cas d'études présentant diverses problématiques pourrai nous permettre d'évaluer la pertinence du pilotage par le produit en fonction d'une typologie des systèmes de production.

4.5 Conclusion

Dans ce chapitre, nous avons mené une série d'expériences sur un cas industriel, afin d'évaluer la pertinence du pilotage par le produit. Pour mettre en place un plan d'expériences, nous avons fait abstraction des problèmes de transmission et de stockage de l'information et nous nous sommes focalisés sur les algorithmes de prise de décision.

Ces expériences illustrent d'une part la manière dont le pilotage par le produit peut être concrètement réalisé. D'autre part, elles comparent diverses procédures de décisions centrées sur le produit avec des procédures classiques. Les résultats expérimentaux montrent la faisabilité du pilotage par le produit. Ils montrent de plus que les performances du pilotage par le produit dépendent fortement de l'environnement décisionnel dans lequel il est placé.

Dans la situation simplifiée dans laquelle nous nous sommes placés (*software-in-the-loop*), nos expériences ont validé la pertinence du pilotage par le produit. Cependant, on peut s'interroger sur les performances de ce mode de pilotage dans une situation plus réaliste, dans laquelle on tiendrait compte des problématiques de transfert et de stockage des informations (*hardware-in-the-loop*).

Pour pouvoir représenter ces aspects, les expériences devraient donc exploiter les composants multi-agents présentés au troisième chapitre. Toutefois, ces expériences n'ont pas pu être réalisées dans le cadre de cette thèse. La première difficulté est le manque de temps nécessaire à leur mise en œuvre (cet obstacle est renforcé par la faible maturité du système multi-agents). Au delà de ces aspects assez contingents, la seconde difficulté est la spécification des facteurs d'une telle expérience (par exemple les reconfigurations et évolutions des systèmes opérant et de pilotage qui sont difficilement modélisables), ainsi que la mesure des performances, qui sont souvent des critères qualitatifs. La conduite d'une telle expérimentation fait partie de nos perspectives de travail.

L'approche utilisée dans ce chapitre consistait à expérimenter sur les algorithmes de pilotage par le produit, en *choisissant* une situation d'étude permettant de faire apparaître spécifiquement des problématiques d'hybridation des approches centralisées et distribuées du pilotage.

Le chapitre suivant montre à l'inverse la manière dont l'environnement de modélisation et d'évaluation peut être exploité pour répondre aux besoins industriels, en offrant des services de benchmarking dans les différentes étapes d'un projet de mise en place d'un système de pilotage par le produit.

Chapitre 5

Vers un service de benchmarking pour le pilotage par le produit des systèmes de production

5.1 Introduction

Ce chapitre présente la manière dont nos travaux peuvent être exploités pour le transfert des technologies de pilotage par le produit dans les activités de l'équipe de recherche technologique (ERT) TRACILog du CRAN. Le sigle TRACILog se réfère aux problématiques de TRAÇabilité, d'Identification et de contrôle par le produit pour les chaînes LOGistiques des filières bois et fibres[31].

Cette problématique s'inscrit de façon plus large dans les actions que mène le pôle de compétitivité « Fibres Naturelles »[32] autour de trois orientations stratégiques :

- à long terme, susciter et accompagner une rupture, avec l'émergence d'une nouvelle industrie de la fibre naturelle à partir de nouvelles fonctionnalités qui lui seront apportées,
- à moyen terme, susciter et accompagner l'innovation dans les produits et process des entreprises du pôle, en particulier dans des démarches transversales,
- et en continu, favoriser l'amélioration de la compétence des hommes, des entreprises, et des organisations, par l'information, la diffusion de la connaissance, la formation, les qualifications, ainsi que les synergies entre acteurs.

Les industries des filières bois et fibres, couvrant en particulier les secteurs du textile ainsi que du bois-ameublement, subissent de fortes exigences de traçabilité et de gestion de la variété. Ces entreprises sont demandeuses de solutions leur permettant de répondre à ces exigences, et le pilotage par le produit nous apparaît désormais comme une approche possible, renforcée par le développement des technologies d'identification par radio-fréquences. Pour faciliter le transfert de l'approche de pilotage par le produit vers les industries, l'ERT tracilog se doit à la fois :

- de faire la démonstration des apports potentiels du pilotage par le produit,
- d'offrir des outils permettant d'estimer sa pertinence face à des approches concurrentes, ainsi que d'assister l'entreprise dans ses choix lors des phases de définition, de développement et de déploiement de ce concept de pilotage par le produit.

L'équipe de recherche technologique TRACILog a par rapport aux entreprises des filières bois et fibres le rôle de *centre de ressources*. Ce chapitre applicatif a pour objectif de montrer en quoi

[31] le site internet de l'équipe peut être consulté à l'adresse http ://tracilog.free.fr ou sur le site du cran
[32] http ://www.pole-fibres.uhp-nancy.fr

nos travaux contribuent à cet objectif.

Nous analyserons d'abord les besoins industriels en matière de modélisation et de test. Ensuite, nous montrerons les apports et limites de l'environnement d'évaluation présenté dans les chapitres précédents par rapport à ces besoins. Enfin, nous présenterons la plateforme d'essais de l'ERT TRACILog, qui nous a permis de compléter la gamme de nos outils d'étude du concept de pilotage par le produit en y ajoutant un point de vue plus technique.

5.2 Besoins industriels en services de modélisation et de test

5.2.1 Problématiques spécifiques des industries des fibres naturelles

En premier lieu, ce type d'industrie doit offrir à ses clients des possibilités de personnalisation très poussées des produits. Par exemple dans les domaines de l'ameublement (cuisines intégrées), de la construction (charpente) ou de la menuiserie (portes, fenêtres), la fabrication doit souvent être réalisée sur mesures et/ou à la commande. Les produits fabriqués en plus grandes séries, correspondant à un marché grand public, sont fabriqués selon des mesures et processus standards, mais sont souvent des produits à options, induisant une certaine personnalisation. De même, l'industrie textile doit gérer une forte variété des produits, qui est due aux diverses tendances de la mode et ainsi se doit de personnaliser les vêtements par exemple (par le choix d'un motif, d'une couleur ou d'un slogan), ou de manière ultime sur certains marchés, de les réaliser sur mesures. La fabrication industrielle de chaussures sur-mesure fait aussi sans doute partie de notre futur proche. En bref, les industries de fibres naturelles sont un domaine ou la « mass-customization » a un fort potentiel commercial.

Par ailleurs, la traçabilité des matières doit être assurée. En effet, les nouvelles directives européennes sur l'éco-certification imposent d'indiquer la provenance du bois utilisé. Il est donc nécessaire de tracer le parcour des matières, depuis l'exploitation forestière jusqu'a la vente d'un produit fini. De même, la traçabilité permet dans le secteur textile d'assurer le client de l'origine du produit, afin de lutter contre la contre-façon.

Enfin, la nature même du matériau vivant qu'est le bois impose une gestion individuelle des produits. En effet, les pièces en bois massif sont hétérogènes : elles varient dans leurs teintes et peuvent présenter des singularités (noeuds, par exemple) ou d'autres imperfections. Les méthodes de production devraient donc s'adapter à chaque pièce, puisqu'elles sont toutes différentes. Ainsi, se posent les problèmes d'appariement par teintes semblables, l'adaptation d'un plan de découpe aux imperfections du bois, etc...

L'ensemble de ces exigences amène les industriels à s'interroger sur l'opportunité de mettre en place des systèmes de pilotage par le produit. Cependant, un certain nombre de décisions doivent être prises dans un tel projet induisant une grande variété de problèmes.

5.2.2 Mise en place incrémentale du pilotage par le produit

Conformément à une approche d'ingénierie-système, nous divisons le projet de mise en place d'un tel système de pilotage par le produit en trois phases principales :

- la *définition* du système de pilotage par le produit, dans laquelle sont spécifiées les données à associer au produit et la manière de les exploiter,
- le *développement*, dans lequel les éléments logiciels permettant de remplir ces fonctions d'association et d'exploitation des données du produit sont codés,
- le *déploiement*, phase dans laquelle les éléments logiciels développés doivent être intégrés dans une architecture technique (serveurs, postes de travail, équipements d'identification

des produits, etc...).

Des problématiques spécifiques sont liées à chacune de ces trois phases. En effet, celles-ci constituent les étapes successives de la projection des concepts abstraits présentés au début du chapitre trois (e.g. une organisation stigmergique du pilotage), vers des éléments opérationnels concrets. À chaque étape, les particularités du cas d'étude doivent être prises en compte, permettant au final une mise oeuvre du pilotage par le produit conforme aux spécificités du cas.

D'abord, la phase de définition est centrée sur les procédures de décisions. Les points principaux devant être définis sont, d'une part, la nature des données devant être associées aux produits, et d'autre part, la manière dont ces données sont synchronisées avec l'état physique du produit, mais aussi la manière dont les processus de décision centralisés ou distribués peuvent utiliser ces données. Doivent donc être définis :

- l'emplacement et la nature des points d'observation des flux de produits,
- la nature des données portées par chaque produit,
- les algorithmes permettant d'initialiser les données des produits, et prendre les décisions de pilotage à l'aide de ces données.

Ensuite, la phase de développement est liée à des problématiques de transmission et de stockage des informations associées aux produits. Dans cette étape, il est, en effet, requis de faire des choix sur l'implémentation des fonctions de base de ce pilotage par le produit. Par exemple, la manière de mémoriser les informations liées aux produits ou la manière dont les centres de décision peuvent interagir avec les produits. Au chapitre trois des composants logiciels permettant de remplir ces fonctions grâce à un système multi-agents ont été proposés. En effet, l'outil multi-agents convient bien pour les études académiques du comportement d'un système contrôlé par le produit ; cependant, pour une application industrielle, différentes architectures (à base d'agents ou plus classiques) peuvent être envisagées.

Enfin, la phase de déploiement amène à se poser des problèmes d'ordre technique. Ainsi, nous devrons choisir la technologie d'identification des produits (RFID, code à barres, etc...) et la localisation des lecteurs et des tags. Les problématiques de fiabilité et de rapidité des identifications doivent être aussi considérées, ainsi que l'intégration des équipements d'identification avec les systèmes automatisés existants.

Le tableau 5.1 synthétise ces diverses problématiques.

Phases du projet	**Problématiques**
Définition	– nature et positionnement des points d'observation des produits, – nature des données portées par chaque produit, – algorithmes permettant d'initialiser les données des produits, – algorithmes de pilotage à l'aide de ces données.
Développement	– mode de stockage des informations associées aux produits, – protocole d'interaction entre produits et centres de décision, – mécanismes de synchronisation produits informationnels et produits physiques.
Déploiement	– choix d'une technologie pour instrumenter les produits, – localisation des lecteurs et des tags, – intégration dans un système automatisé.

Tab. 5.1 – Les problématiques à évaluer pour chaque phase du projet de mise en place d'un pilotage par le produit.

La nature des critères de performance associés à ces problématiques dépend de la phase de développement considérée. Ainsi, les algorithmes de prise de décision sont évalués par rapport à des indicateurs de productivité du système opérant. L'architecture choisie dans la phase de développement est évaluée en fonction de critères du domaine de l'informatique : quantité de messages échangés, consommation de mémoire et de temps de calcul, ainsi que des critères qualitatifs comme la complexité des systèmes développés, leur maintenabilité, etc... Enfin, en phase de déploiement, les critères sont d'ordres techniques : fiabilités des procédures d'identification dans un environnement pouvant être perturbé, coût des installations d'identification, etc...

5.3 Apports et limites de l'environnement d'évaluation : cas d'un fabricant de meubles

Cette section montre en quoi les besoins industriels établis dans la section précédente peuvent être satisfaits par l'utilisation de l'environnement d'évaluation. Elle est illustrée par l'exemple d'un fabricant français de meubles en kit, qui collabore avec l'ERT tracilog dans le cadre de la thèse Cifre de Thomas Klein qui est en cours. Seules les premières étapes de la mise place du pilotage par le produit seront présentées, le projet complet étant d'une envergure bien trop importante pour être étudié dans le cadre de cette présente thèse (la thèse Cifre sus-citée a, entre autre, cet objectif). Le but de cette section est donc de mettre en regard les besoins des industriels en terme d'évaluation, avec les services pouvant être offerts par notre environnement.

Présentation du cas

Comme nous l'avons dit précédemment, les problématiques générales des industries des fibres naturelles se déclinent essentiellement en un problème de gestion de la variété des produits. En effet, le nombre de références est très important pour offrir aux distributeurs une gamme de produits suffisamment large, ainsi que pour permettre à ces derniers de se différencier les uns des autres.

De plus, le pilotage doit tenir compte des objectifs fixés aux différents îlots de production. Comme nous l'avons évoqué au deuxième chapitre, la production s'effectue en effet en trois phases (figure 5.1) :

- une phase de *débit* dans laquelle l'objectif principal est l'optimisation du rendement matière, avec une problématique de choix de plans de coupe,
- une phase de *perçage et d'usinage* où l'objectif est la minimisation du délai et la maîtrise des niveaux d'en-cours,
- une phase *d'emballage* qui nécessite une main d'œuvre importante, et où l'objectif est donc d'optimiser la productivité, c'est à dire d'éviter les ruptures d'approvisionnement en pièces à emballer.

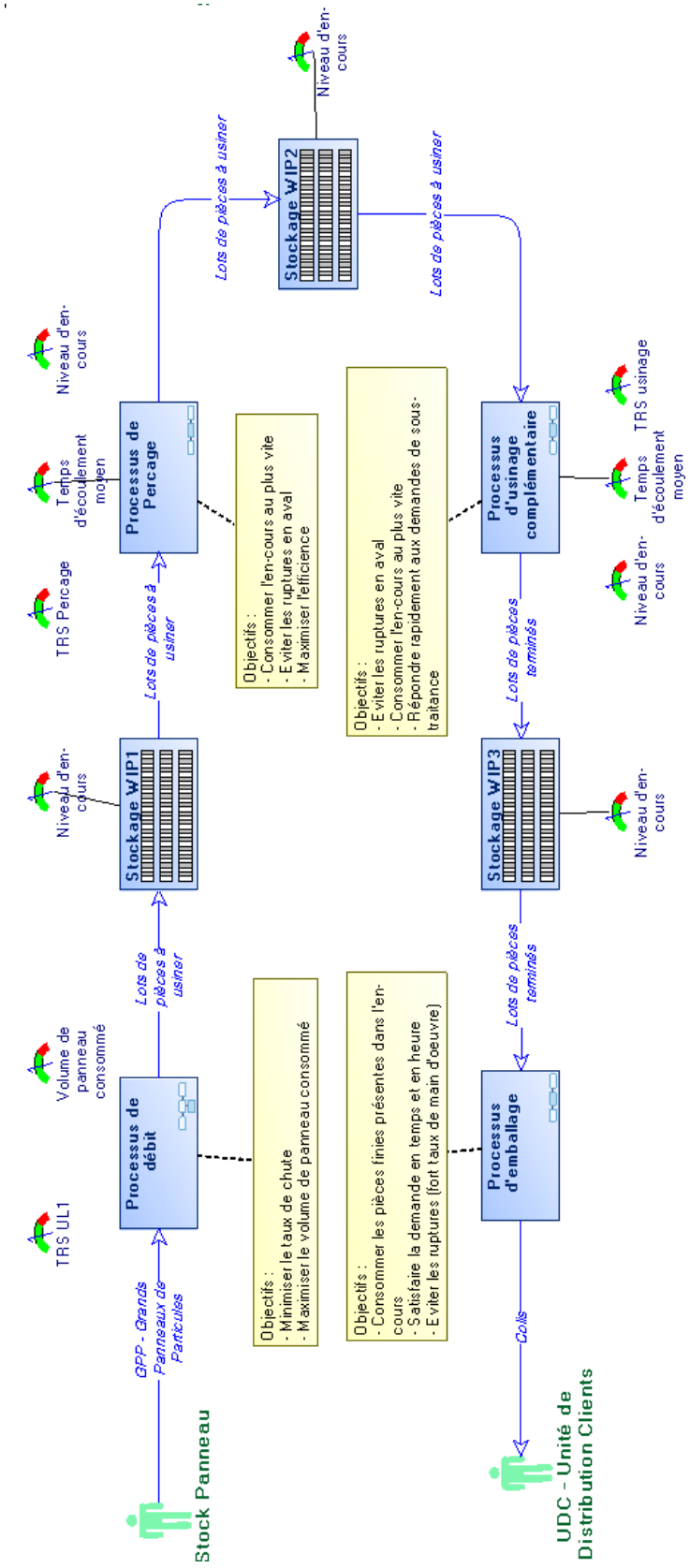


FIG. 5.1 – Diagramme des processus de production du cas d'étude industriel (réalisé avec Mega-Process), faisant en particulier apparaître les objectifs de productivité ainsi que les indicateurs de performance utilisés.

Différentes organisations du système de pilotage peuvent être envisagées pour satisfaire ces objectifs. Dans un premier temps, le pilotage a été organisé de manière centralisée ; par la suite un système de pilotage par kanbans a été mis en place. Il a été possible, par ailleurs, d'adopter une approche de pilotage par le produit, pouvant éventuellement être combinée au pilotage centralisé et/ou par kanbans.

La création du modèle d'émulation relatif à ce cas d'étude industriel a été présentée à la fin du deuxième chapitre de ce mémoire.

5.3.1 Définition des modalités de pilotage par le produit

Pour répondre aux besoins d'évaluation dans cette phase de définition, l'environnement d'évaluation est configuré pour fonctionner en *software-in-the-loop*, c'est à dire qu'un modèle du système de pilotage est utilisé. Ainsi, l'expérimentateur peut se concentrer sur les points à définir, en particulier les algorithmes de pilotage, sans devoir se soucier des problèmes de développement.

L'utilisation de l'environnement d'évaluation peut être illustré par deux exemples : le choix du placement des équipement d'identification, et la sélection de paramètres optimaux pour les algorithmes de décision locaux.

Le choix du positionnement des équipements d'identification (par exemple des transpondeurs RFID) va en fait de pair avec la définition des algorithmes de pilotage. En effet, le fonctionnement des algorithmes de pilotage rend nécessaire la présence d'un minimum de points de lecture. Cependant, certaines décisions de positionnement doivent être faites. Ainsi, dans l'atelier de fabrication de meubles en kit, où chaque machine est précédée d'une voie de rouleaux jouant le rôle de tampon d'entrée, on peut positionner des équipements d'identification des produits en tête de voie, en fin de voie, ou à ces deux emplacements (figure 5.2). Le changement de séquence des ponts de pièces dans la voie de rouleaux étant matériellement impossible (discipline FIFO), des algorithmes de conduite peuvent être imaginés dans ces trois cas.

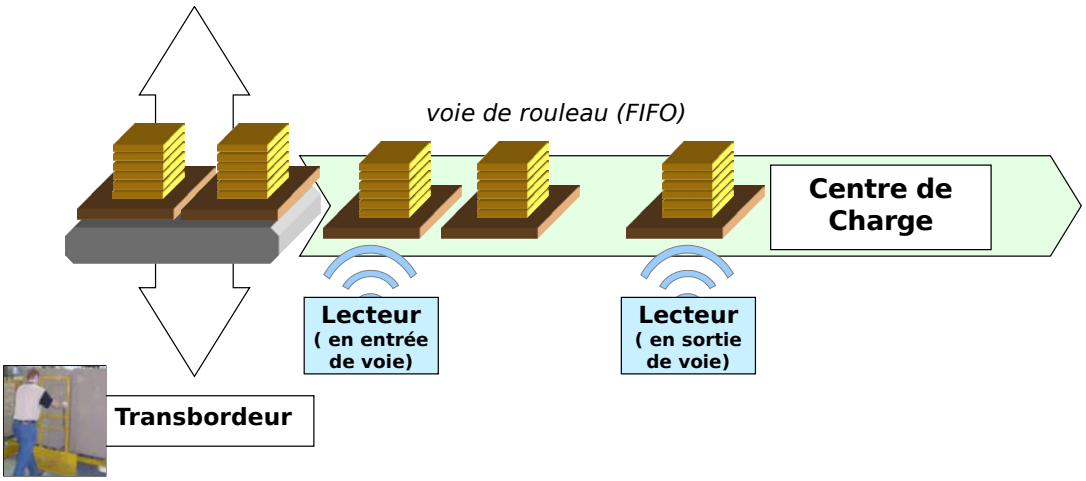


FIG. 5.2 – Exemple du placement des lecteurs RFID en début et/ou en bout de voie de rouleaux

Si le lecteur est placé en tête de voie, l'algorithme de conduite devra maintenir une image de la file d'attente devant la machine, tandis qu'avec un placement en queue de voie la conduite de la ressource pourra être directement basée sur l'identification du lot de pièces à traiter. On constate donc que la complexité de la fonction de conduite à écrire peut être évaluée qualitativement par l'expérimentateur, lui permettant de selectionner la solution la plus simple.

Cependant, des choix plus subtils nécessitent une étude plus poussée. Par exemple, est-il utile de placer un lecteur en tête de voie en plus du lecteur en queue de voie, afin d'augmenter la projection dans le futur du pilotage (e.g. en anticipant une rupture d'approvisionnement) ? Pour évaluer les alternatives, la possibilité de modéliser rapidement et d'évaluer les différentes solutions constitue une aide précieuse. Une première approche consiste à exécuter l'émulation, afin de vérifier le bon fonctionnement des algorithmes de pilotage dans diverses situations d'état. La reproductibilité des tests, ainsi que la disponibilité d'outils de débogage (permettant par exemple une exécution pas-à-pas) permettent de détecter, comprendre et corriger les erreurs pouvant apparaître. Pour une évaluation plus quantitative, par exemple une évaluation du retour sur investissement de l'achat d'un second lecteur, un plan d'expériences comparable à celui présenté au chapitre quatre devra être utilisé.

L'environnement peut aussi être utilisé pour sélectionner les algorithmes de pilotage devant être choisis. Le chapitre précédent présentait ainsi la comparaison entre deux manières de mettre en oeuvre le pilotage par le produit. Dans le cas qui nous occupe, une telle démarche pourra être utilisée pour la mise au point du pilotage de la cellule de débit. En effet, les objectifs qui régissent le choix des pièces à débiter sont principalement la minimisation du taux de chutes, et la maîtrise de niveaux d'encours dans l'atelier. Cependant, les processus de perçage et d'usinage ont une durée très variable, à cause de la complexité des flux, et de perturbations externes (figure 5.3). A cause de ces délais variables, et difficiles de prévoir, la maîtrise des encours est difficile.

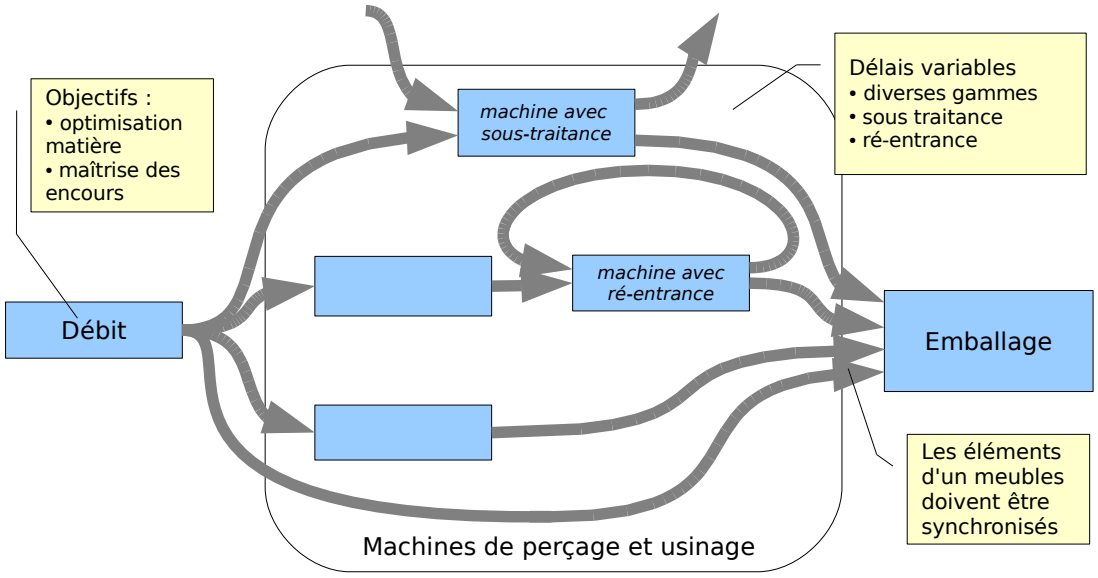


FIG. 5.3 – Complexité des flux dans les îlots d'usinage et de perçage

On peut esquisser l'algorithme de pilotage pouvant être utilisé pour piloter l'îlot de débit. Les informations des produits, correspondant à une prise de décision centralisée, peuvent être choisies de la manière suivante :

– l'identifiant du plan de coupe nominal,
– les identifiants des éventuels plans de coupe alternatifs,
– la date de disponibilité à l'emballage (au plus tôt, au plus tard),

– le délai prévisionnel de perçage/usinage ;

L'algorithme de pilotage doit prendre en compte à la fois le critère de minimisation du taux de chutes et de maîtrise des encours [33]. Il lui est aussi possible de tenir compte de l'état de disponibilité de la machine de débit, ainsi que de la situation d'état de l'atelier, par exemple le fait qu'une machine soit en panne, soit en famine ou au contraire en sur-charge.

La règle de sélection des lots à débiter (c'est à dire le choix d'un plan de coupe) fait donc rentrer des coefficients permettant de doser l'importance relative de l'objectif d'optimisation matière par rapport au respect du juste-à-temps, ou encore de doser l'autonomie des décisions par rapport au respect des données associées au produit.

L'environnement d'évaluation peut être utilisé pour valider cette règle de choix, et pour fixer les coefficients. En effet, différentes modalités de réglage peuvent être appliquées dans un plan d'expériences, permettant ainsi de sélectionner la valeur optimale des coefficients, en fonction de scénarios de perturbations subies par l'atelier.

Une fois définit le positionnement des lecteurs, la nature des informations portées par les produits, et les algorithmes de décision, le projet entre dans la phase de développement.

5.3.2 Développement d'un système de pilotage par le produit

Dans la phase de développement, le système de pilotage par le produit *réel* est utilisé, en conjonction avec l'émulateur. Ce dernier doit donc fonctionner dans le mode *hardware-in-the-loop*.

Les critères d'évaluation sont de nature informatique. Les mesures pourront porter sur la communication (quantité, fréquence, nature des information échangées), sur la capacités du système (e.g. délai d'accès aux informations des produits, mémoire consommée, etc...) et sur sa robustesse (e.g. non fiabilité du réseau).

Le principal facteur à évaluer est l'architecture de mise en oeuvre du pilotage par le produit. Les composants logiciels présentés au chapitre trois pourront ainsi être comparés à d'autres solutions plus classiques, telles que la mémorisation des informations associées aux produits dans une base de données. Il est aussi possible, en combinant les composants orientés-agents de diverses manières, de comparer différents protocoles d'interaction entre les produits et les centres de décision locaux, et différentes topologies du système de décision.

Cependant, un important travail visant à assurer la stabilité des composants multi-agents doit être effectué avant de pouvoir réaliser ces expériences.

5.3.3 Limites d'une approche basée sur l'émulation

Ces quelques exemples montrent les apports de l'environnement d'évaluation. Toutefois, une approche basée sur l'émulation permet difficilement de prendre en compte les problématiques d'ordre technique, qui se posent en phase de déploiement.

L'environnement d'évaluation doit donc être complété par une plateforme d'essais (figure 5.4). Celle-ci permettra, en jouant le rôle de démonstrateur des technologies de pilotage par le produit, de déceler, de comprendre et de résoudre les problèmes techniques, qui ne peuvent pas être modélisés dans l'émulateur.

Nos propositions de système de pilotage auront ainsi été testées dans un environnement réel, qui les confronte à des contraintes de durée de vie des tags RFID, des problèmes d'interférences, de portée ou de vitesse des lecteurs, etc... permettant d'assurer de leur faisabilité technique.

[33] Ce problème est simplifié par le fait que les plans de coupe nominaux résultent d'une décision centralisée. Ils sont donc cohérents par rapport aux dates de besoin.

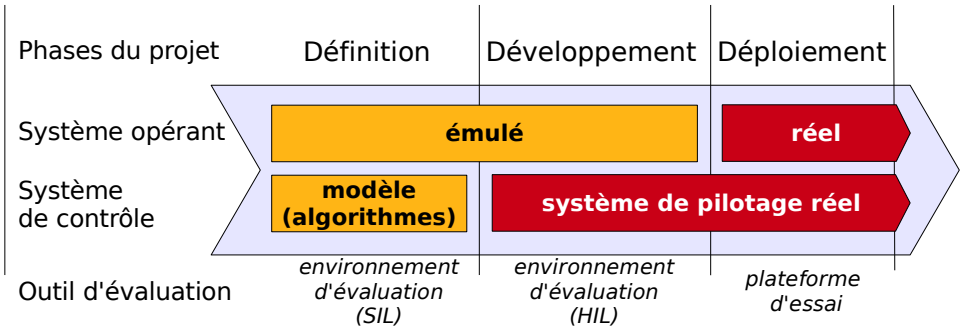


FIG. 5.4 – Les outils de transfert technologique utilisés au cours des phases successives d'un projet de mise en place d'un pilotage par le produit

La fin de ce chapitre est donc dédié à la présentation de la plateforme d'essais de l'ERT TRACILog ainsi qu'à l'application à cette plateforme d'un système de pilotage par le produit.

5.4 La plateforme d'essais de l'ERT

5.4.1 Architecture technique

Fonctions

Les fonctions attendues de la plateforme d'essais sont d'abord la possibilité d'étudier les phénomènes liés au stockage et à l'attente des pièces, aux temps machine. De plus la perte de l'ordre des pièces et les problématiques de tri devront aussi pouvoir être étudiées.

Deuxièmement, la réalisation d'opérations d'assemblage et de désassemblage sont requises. En effet, les industries des fibres naturelles comportent ces deux types d'opérations. Ainsi, dans le domaine textile comme dans les industries du meuble, une opération de débit (découpe de tissu, de bois, de panneaux de particules) est suivie d'opérations d'assemblages (couture, assemblage de pièces de bois, mis en colis d'un kit).

Ensuite, la plateforme d'essais doit comporter différents modes d'identification des produits. Ainsi, elle est équipée d'une instrumentation biométrique, de transpondeurs RFID, tandis que l'identification visuelle des pièces est facilitée par un code des couleurs.

Enfin, la plateforme doit permettre l'intégration d'un opérateur et d'opérations manuelles.

Produits

Les produits ont été choisis pour répondre à ces attentes, en particulier pour un assemblage / désassemblage facilité. Les produits se composent ainsi d'un support carré pouvant accueillir jusqu'à quatre plaquettes carrées de 10 cm de côté, à la manière d'un puzzle. De plus, chaque plaquette peut recevoir une pastille (figure 5.5a).

Un produit est dit de type n lorsqu'il comporte n plaquettes (la manière dont les plaquettes sont disposées n'est pas prise ne compte). De plus les plaquettes sont colorées, ce qui augmente la variété. Avec deux couleurs de plaquettes, 31 produits différents peuvent donc être assemblés.

Chaque sorte de pièces (support, plaquette, pastille) est munie d'une étiquette RFID (figure 5.5b). Une demi-douzaine de lecteurs RFID sont répartis sur la plateforme.

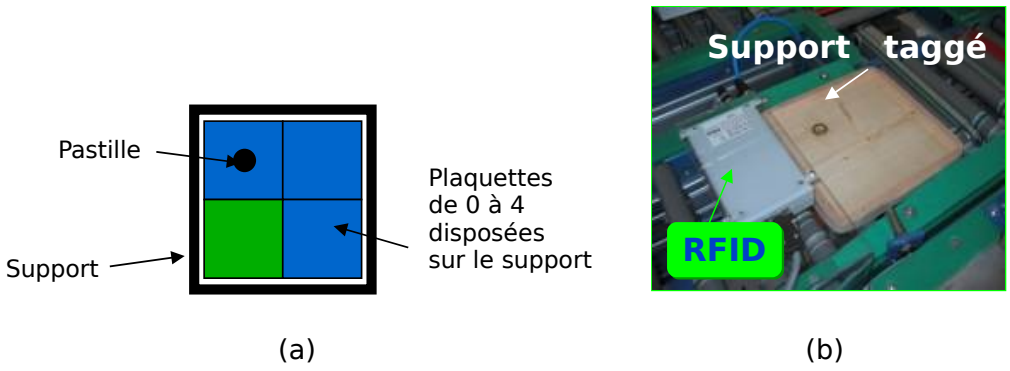


FIG. 5.5 – Les divers types de produits (a) Produit taggé et lecteur RFID (b)

Configuration matérielle

La figure 5.6 présente la plateforme qui a été développée.

Les opérations sur les produits sont réalisées sur deux postes d'assemblage :

- le poste M_1 permet la préhension des plaquettes, grâce à une ventouse. Ce poste est dédié à l'assemblage des plaquettes –disposées dans le magasin S_2– sur les supports, ou encore le désassemblage des produits (les plaquettes sont alors déposées hors du système),
- le poste M_2 permet de manipuler les pastilles, qui sont prises dans le magasin S_3 pour être déposées sur les produits.

Les deux magasins S_2 et S_3 fonctionnent par gravité, en FIFO ; par conséquent seule la première plaquette (respectivement pastille) peut être prise. Cependant, les plaquettes peuvent être ôtées de S_2 et déposées hors du système (comme dans la procédure de désassemblage), ce qui rend possible la recherche d'un type précis de plaquette dans le stock.

En plus de ces deux postes d'assemblage, la plateforme peut accueillir des équipements d'identification et d'analyse des produits : un système de vision et un scanner à micro-ondes pour l'identification biométrique.

Pour assurer l'acheminement des supports entre ces différents postes, un système de convoyage à rouleaux a été déployé. Sur ces convoyeurs ont été placées des butées permettant de bloquer les supports pour effectuer un traitement. Les divers convoyeurs sont reliés par des renvois d'angle ou par des systèmes d'aiguillage, qui sont eux aussi précédés de butées. Les convoyeurs sont de plus équipés de nombreux capteurs optiques permettant de détecter la présence des supports.

Le réseau de convoyeurs permet d'acheminer les produits d'une voie de stockage des supports vides vers une voie de stockage des produit finis, vers une voie de tri, ou encore de les re-diriger dans le système (bouclage). Enfin, le poste M_2 est muni d'une voie de garage permettant de mettre en attente un produit (modification de l'ordre des produits). Les flux de matière sont donc d'une complexité comparable à celle d'une cellule de transformation du bois ou du textile.

Instrumentation RFID

La plateforme est munie de sept lecteurs RFID. Ces lecteurs fonctionnent à courte portée (une dizaine de centimètres), et sont disposés au dessus des convoyeurs à rouleaux (figure 5.5b). Ces lecteurs peuvent lire simultanément plusieurs étiquettes se trouvant dans leurs champs.

Deux lecteurs sont positionnés en amont et en aval du poste M_1, afin de permettre le paramétrage des opérations d'assemblage, et le contrôle de leur exécution correcte. De même, le

FIG. 5.6 – La plateforme d'essais TRACILog

poste M_2 est précédé d'un lecteur. Les aiguillages vers le tapis de tri et vers le tapis de stockage des produit finis sont eux aussi précédés de lecteurs.

En plus des équipements RFID placés au dessus des convoyeurs, la plateforme est équipée

- d'un lecteur manuel « douchette » destiné à identifier les supports vides du magasin de matière première S_1 que l'on introduit dans le système,
- d'un lecteur placé sur le stock S_2 servant à identifier les plaquettes avant leur assemblage.

L'ensemble de ces équipements est schématisé figure 5.7.

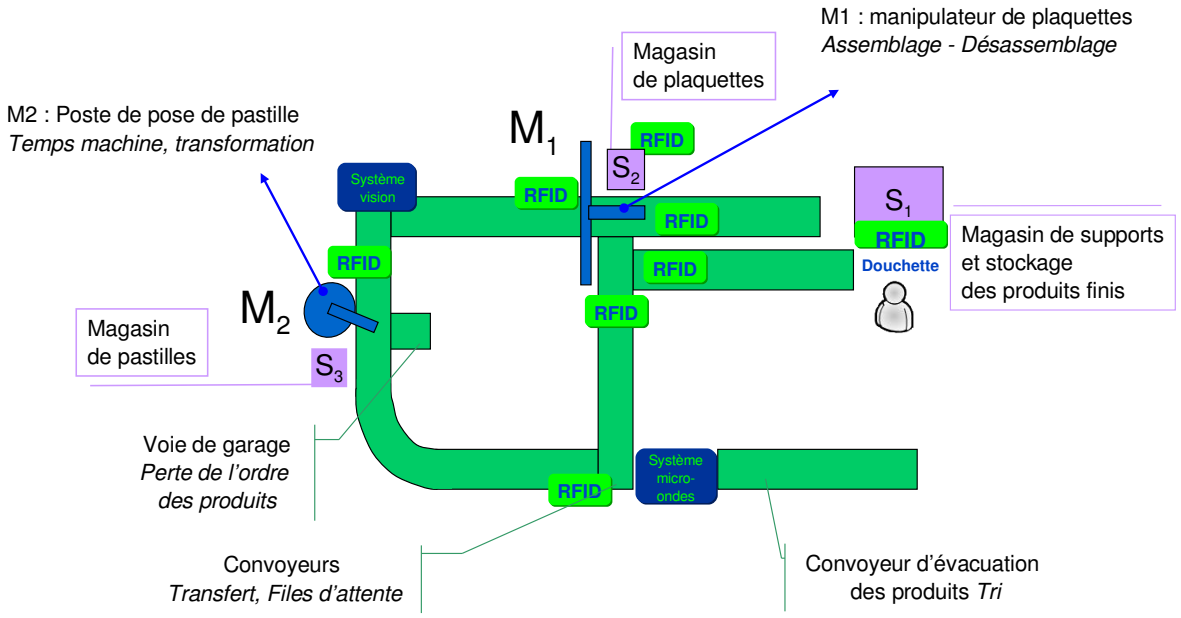


FIG. 5.7 – Schéma de la plateforme TRACILog

Équipements de contrôle et réseau de communication

La commande des équipements se fait grâce à un unique automate Siemens, doté d'un écran tactile. Celui-ci est connecté aux divers actionneurs et capteurs grâce à un réseau de terrain.

Les lecteurs RFID sont quant à eux connectés sur un réseau RS485 dans une topologie « en bus ». Ce réseau RS485 est interfacé avec le réseau ethernet grâce à un coupleur Moxa, ce qui permet d'accéder aux lecteurs RFID depuis une station de travail Sun chargée de les gérer.

Un échange partiel de l'automate avec la station Sun est possible par l'emploi d'un protocole basé sur une connexion sur socket TCP.

Le réseau global (figure 5.8) est donc assez hétérogène.

5.4.2 Système de contrôle existant

La plateforme a été livrée avec un système de contrôle. Celui-ci comporte deux modes de fonctionnement : dans le mode « autonome », le contrôle des équipements est effectué par l'automate sans communication avec les systèmes externes, tandis que dans le mode « intégré », le système de pilotage, supporté par la station Sun, est en interaction avec des logiciels de gestion. Le développement des programmes automates et du logiciel de pilotage a été assurés par deux sociétés différentes. En particulier, la société Sun a développé le middleware assurant le pilotage, et était aussi responsable de l'intégration des divers développements.

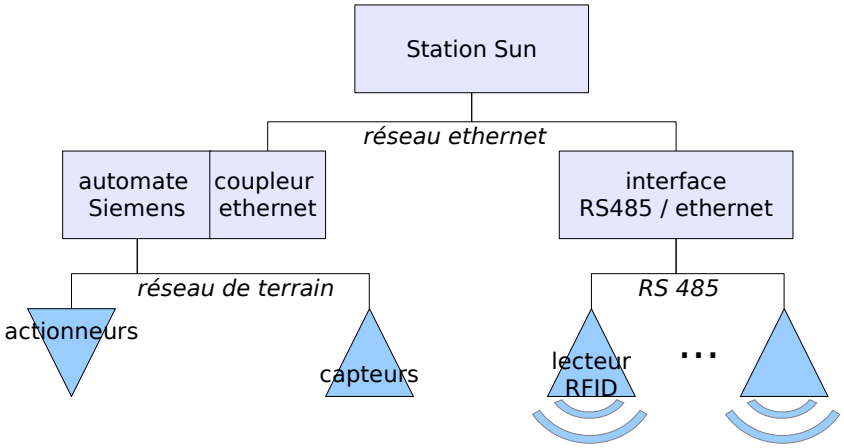


FIG. 5.8 – Équipements de contrôle de la plateforme

L'analyse du système de contrôle actuel nous offre un point de comparaison par rapport au contrôle par le produit. Elle est de plus nécessitée par l'intégration du système multi-agents à la plateforme.

Contrôle autonome par l'automate

La commande des automatismes peut être réalisée de manière isolée et autonome par l'automate. Le fonctionnement de la plateforme peut être réglé par l'opérateur grâce à l'interface tactile connectée à l'automate. Le fonctionnement des postes d'assemblage est déterminé par ces réglages, et déclenché par l'arrivée d'un support vide. Comme la configuration est donnée aux postes d'assemblage de manière centralisée, un changement de réglage exige d'attendre la fin des opérations en cours.

Puisque l'automate est coupé de l'extérieur, les lecteurs RFID (gérés par la station Sun) ne sont pas utilisés. Il n'y a donc pas de choix des couleurs pour l'assemblage des plaquettes, ni de vérification du bon déroulement de l'assemblage.

Dans ce mode de fonctionnement, la plateforme fonctionne comme une machine à commande numérique, dont la conduite est assurée par l'opérateur. Ce dernier doit en effet régler et déclencher les opérations de fabrication, vérifiant manuellement la bonne marche de la machine. L'intégration est donc très faible, les fonctions assurées par l'automate sont plutôt relatives à la commande des automatismes qu'au pilotage.

Pilotage intégré

Ce second mode de fonctionnement permet une plus grande automatisation et une meilleure intégration de la plateforme dans le système d'information.

Dans cette configuration, c'est la station Sun qui joue un rôle central, en exécutant un logiciel de pilotage. Ce système est en interaction avec quatre acteurs de son environnement :

- l'automate auquel sont envoyées des commandes pour superviser l'exécution de la production,
- les lecteurs RFID, qui émettent des évènements d'identification des produits,
- l'opérateur, qui peut être informé sur les ordres à réaliser par l'intermédiaire d'une interface graphique, et peut en retour envoyer des événements grâce à la douchette RFID,

– le logiciel de GPAO prélude production, qui émet des ordres de fabrication, et reçoit des déclarations de production.

L'architecture du système est donc un modèle classique « en trois couches » (figure 5.9). Le logiciel entretient une image informationnelle des produits, qui entrent dans deux boucles cybernétiques :

– vis-à-vis du *process*, l'action sur les produits physiques est assurée par l'envoi de commandes vers l'opérateur et l'automate, tandis que leur observation est réalisée grâce aux équipements RFID,
– vis-à-vis du *business*, l'image des produits du point de vue du pilotage est synchronisée avec l'image du produit du point de vue de l'ERP par l'échange d'ordres de fabrication (OF) et de messages de déclaration de production (DP).

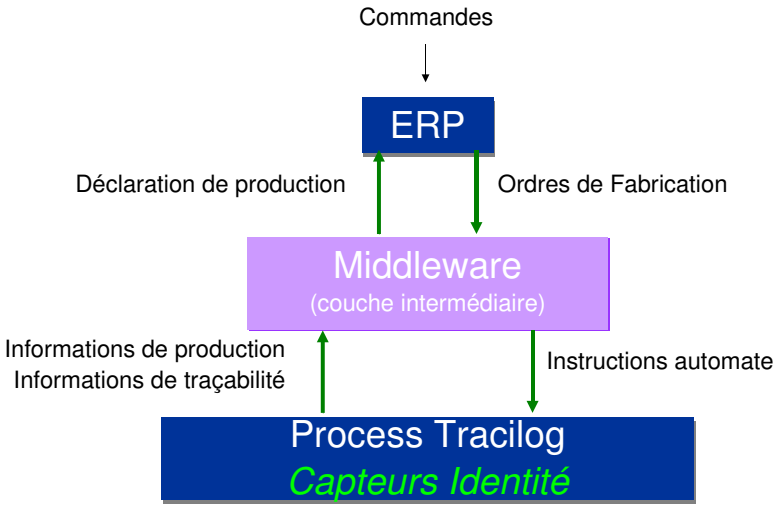


FIG. 5.9 – Positionnement du middleware dans le système de contrôle existant

La technologie utilisée pour associer des données au produit est celle de l'EPC Global[34]. Le code EPC est une évolution du code à barres, comprenant trois champs identifiant le fabriquant, le type de produit, et le produit individuel. Les attributs du produit sont donc stockés dans une base de données externe (dite « base EPC »). Ces données peuvent être initialisées et liées à une pièce particulière en utilisant le lecteur douchette.

On peut illustrer le fonctionnement du pilotage intégré en présentant les étapes d'un scénario typique de fonctionnement :

1. Un ordre de fabrication est reçu. Celui-ci spécifie les quantités à faire et la description du produit sous la forme d'un fichier XML.
2. Les composants requis pour la production sont affichés sur la console de l'opérateur. Ce dernier effectue les sorties du stock en validant son opération avec la douchette RFID. À chaque événement RFID, le code EPC est transmis, et les données relatives au produit sont extraites de la base EPC.
3. Les supports sont perçus par les lecteur RFID lorsqu'ils arrivent devant les postes d'assemblage ou d'aiguillage ; les commandes d'assemblage sont émises vers l'automate. La bonne

[34]EPC Global est une organisation à but non-lucratif créée pour commercialiser la technologie EPC par Uniform Code Council et EAN, les deux organisations qui maintiennent les standards des codes à barres. EPC Global commercialise la technologie originellement développée par l'AutoID Center

exécution des assemblages est vérifiée par les lecteurs RFID de contrôle.

4. Si les diverses opérations se sont déroulées correctement, une déclaration de production est envoyée vers l'ERP.

Bien que ce soit théoriquement possible, le pilotage ne gère par la production simultanée de produits de types différents. La lecture RFID permet en effet de différencier les instances de produits, mais l'architecture du logiciel de pilotage n'est pas adaptée à la gestion de la production simultanée de différents types de produits. Les ordres de fabrication sont donc exécutés séquentiellement, un ordre devant être entièrement terminé avant le début du suivant.

Architecture du middleware

L'architecture du middleware est composée d'un noyau de gestion des événements et d'un ensemble de clients pouvant s'y connecter (figure 5.10). Chaque client est une application à part entière (*standalone java application*) ; la communication avec le noyau événementiel se fait en utilisant des techniques de communication inter-processus (JINI).

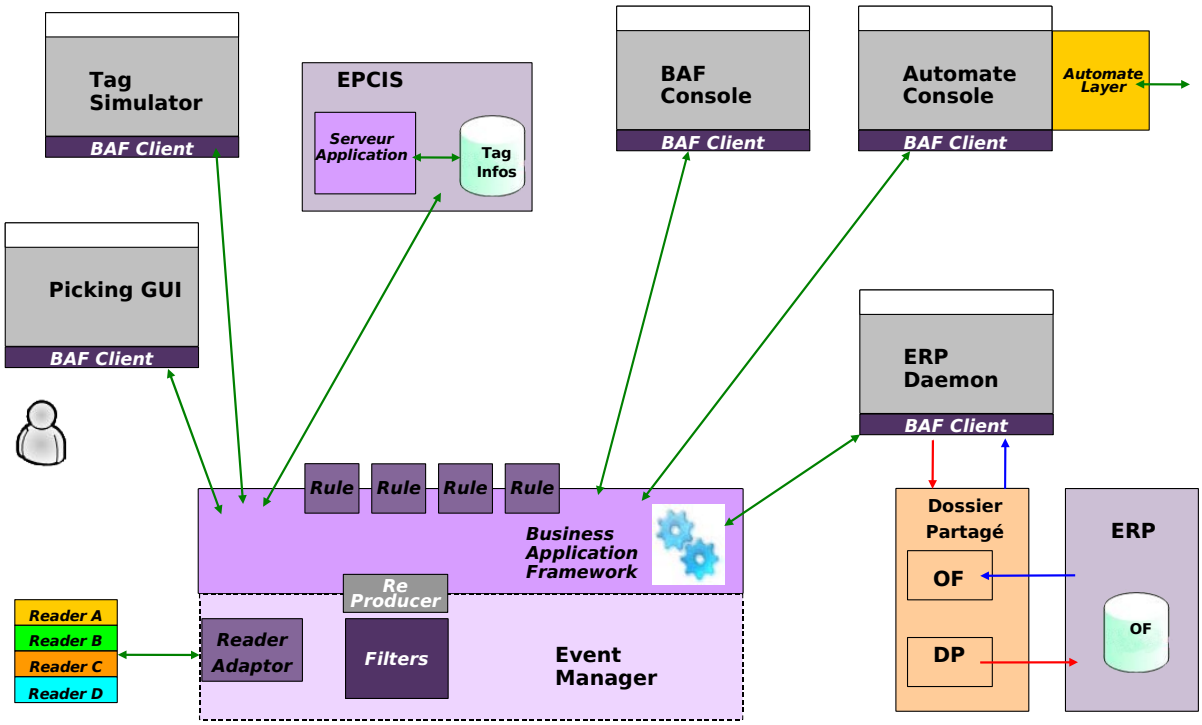


FIG. 5.10 – Architecture du middleware de contrôle livré par Sun

D'une part, le noyau événementiel est composé de l'EventManager, qui offre une couche d'abstraction au-dessus des lecteurs RFID et permet le filtrage et le traitement des événements RFID, ainsi que du framework BAF[35], qui est dédié au routage des évènements en provenance et à destination des clients externes. Ce routage est effectué par un ensemble de règles. En plus des règles, le BAF comporte des *contextes*. Un contexte est une sorte de tableau noir (*blackboard*) : c'est un ensemble d'objets partagés par toutes les règles.

D'autre part, chaque client remplit une fonction particulière :

[35]BAF (Business Events Framework) est un framework événementiel développé par Sun. Ce framework est utilisé pour les développements, mais n'est pas un « produit » officiel.

- Tag Simulator facilite le développement et la validation du système de pilotage en émulant la lecture de tag RFID,
- BAF Console permet de tenir un journal des événements transférés dans le BAF et de l'utilisation des règles,
- Picking est l'interface offerte à l'opérateur pour la manutention des pièces,
- Automate Console est dédié à l'ouverture et la maintenance d'une connexion avec l'automate, ainsi qu'à la transmission des commandes d'assemblage,
- ERP Daemon surveille les dossiers partagés qui servent au transfert des OF et DP.

La limite, déjà mentionnée, sur la production simultanée de plusieurs types de produits est causée par l'utilisation d'un contexte global pour déterminer la recette d'assemblage des produits. Interrogés sur cette limite, les ingénieurs de Sun ont affirmé qu'il aurait été complexe d'associer un contexte de production à chaque instance de produit.

En conclusion de cette section, le système de contrôle livré fonctionne correctement, bien que sa robustesse n'ait pas été confrontée à une véritable campagne de tests. Toutefois, il est limité quant à la gestion de la variété des produits.

5.5 Application d'un pilotage par le produit à la plateforme d'essais

Cette section présente l'application du système multi-agents de contrôle par le produit à la plateforme de l'ERT TRACILog.

Cette intégration se base sur la ré-utilisation du logiciel développé par Sun. En effet, celui-ci offre des fonctions d'interconnection (de *middleware*), et permet ainsi d'intégrer au système d'information les équipements (automate, lecteurs RFID) ainsi que les logiciels de l'environnement (ERP, console opérateur, etc...). Par contre, les fonctions de pilotage de ce logiciel ne sont pas ré-utilisées, puisque le contrôle par le produit s'y substitue.

Cette migration des fonctions de pilotage du middleware vers le système multi-agents est progressive : dans un premier temps l'objectif est uniquement la représentation de l'état des produits (par des agents holoniques produits) ; ensuite on s'intéresse à la commande des équipements (en utilisant des agents holoniques ressources), enfin le pilotage par le produit sera intégré à son environnement opérant (agents ressources) et décisionnel (opérateurs, ERP).

Il nous faut noter ici que seules les premières étapes de cette démarche ont été effectivement mises en œuvre sur la plateforme. Les dernières étapes seront donc présentées brièvement, et il nous faut souligner qu'elles ne sont à l'heure actuelle qu'en cours d'implémentation et de validation. Nous les présentons cependant comme une illustration de l'applicabilité du pilotage par le produit à un cas réel.

5.5.1 Suivi de l'état des produits

Dans cette première étape, il s'agit de faire coïncider l'état des agents-produits, sans influer sur le pilotage de la plateforme. Les fonctions de pilotage sont donc toujours assurées par logiciel de pilotage originel, tandis que le système multi-agents y est intégré. Cette intégration repose sur deux actions :

- l'adaptation de l'agent de connexion au système opérant,
- la configuration des agents-produits

Intégration du système multi-agents au middleware

L'intégration des agents au middleware est réalisée grâce à une adaptation de l'agent de connexion au système opérant. Celui-ci est doté d'une architecture modulaire, qui permet une interopérabilité de classe A [Iung *et al.*, 2001] (figure 5.11). En plus du module de connexion au serveur d'émulation, un module de connexion au BAF a donc été développé. Celui-ci implémente l'interface EventSource, et utilise le composant BAFClient pour assurer la transmission effective des événements. Ce composant embarqué dans l'agent se double d'une règle embarquée au sein du BAF, qui assure la traduction syntaxique entre les événements du BAF et les événements destinés aux agents. En particulier, elle convertit les numéros de lecteur RFID en noms d'événements, et extrait l'identifiant des produits à partir de leurs code EPC.

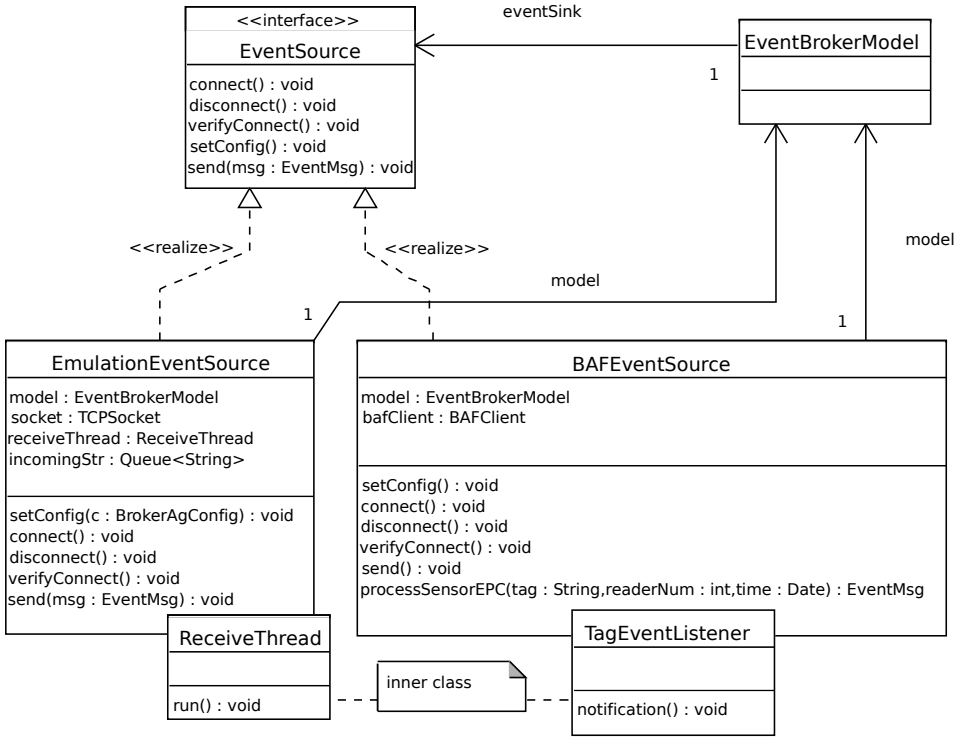


FIG. 5.11 – Diagramme de classes : interopérabilité de l'agent de connexion avec le système opérant physique (BAF) ou émulé

Ainsi modifié, l'agent de connexion au système opérant permet la communication entre les agents et les différentes applications clientes du BAF (figure 5.12). Cependant, à cette étape du développement, seuls les événements de lecture RFID sont transmis vers les agents.

Configuration des agents-produits

La configuration des agents-produits repose sur un automate à états représentant ses changements d'états, ainsi que sur ses attributs.

Parmi les attributs du produit, un ensemble d'attributs relatifs aux spécifications techniques du produit est défini : pour chacun des quatre emplacements du support un attribut est ajouté, dont la valeur correspond au type de plaquette devant être posée et à la présence de pastille.

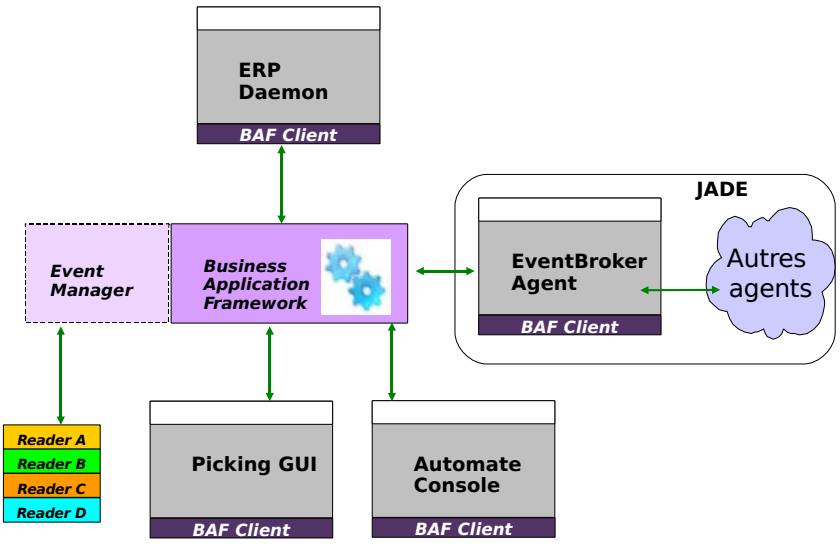


FIG. 5.12 – Intégration du système multi-agents au middleware

Les états observables du produit sont entièrement déterminés par les événements RFID. Chaque lecture RFID d'un produit conduit ainsi à une évolution de son état. La configuration la plus simple des agents-produits est donc l'enchaînement linéaire des états successifs du produit, comme présenté figure 5.13. L'état initial, dans lequel le produit se trouve lors de sa création, est celui d'un ordre à réaliser.

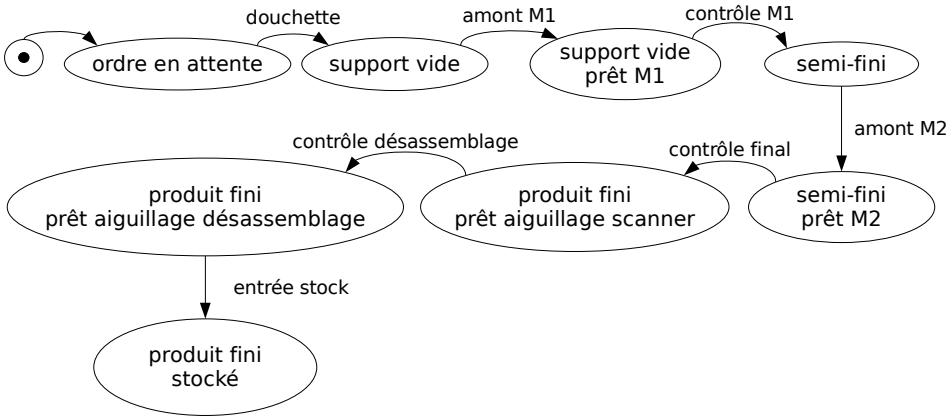


FIG. 5.13 – Configuration de l'agent-produit pour la plateforme TRACILog

La succession d'états du produit est donc déterminée par les capacités d'observation limitées des flux de produits. Dans le cas de la plateforme TRACILog, cette limitation de l'observation des produits est causée par l'impossibilité technique d'utiliser d'autres événements que les informations des lecteurs RFID. En effet, à cause de l'hétérogénéité des réseaux permettant de communiquer respectivement avec les lecteurs et avec les automatismes, les données provenant des capteurs de présence ne sont pas disponibles.

Pour palier à cette difficulté, il est donc envisageable d'intégrer les événements en provenance des capteurs tout-ou-rien de la plateforme. Il est possible de permettre au logiciel gérant une transformation d'envoyer un événement signalant la fin de celle-ci. Cependant, de telles

adaptations peuvent être sources d'erreurs : en effet, la fiabilité de l'observation des produits est assujettie à un fonctionnement nominal de la plateforme[36]. De même, ces observations non liées à une identification du produit peuvent introduire une fragilité dans l'évolution d'état des produits.

5.5.2 Commande des équipements

La deuxième phase du déploiement du système de pilotage par le produit sur la plateforme d'essais est l'association des équipements (poste d'assemblage et aiguillage) à des agents en assurant la commande.

Pour adapter le fonctionnement des agents aux spécificités de la plateforme, deux solutions sont envisageables :

- modifier le comportement des agents, en développant un module de contrôle spécifique chargé par les agents ressources,
- continuer à utiliser les agents génériques de transformation de forme et d'espace et utiliser un système de traduction implanté dans le middleware,

Nous choisissons cette dernière solution, qui permet en particulier de déployer le système multi-agents indifféremment sur système réel ou sur son modèle d'émulation. D'un point de vue architectural, cette seconde solution est plus cohérente, car elle « encapsule » les spécificités de fonctionnement dans un « driver » dont l'interface est générique.

L'application de gestion de l'automate développée par Sun doit donc être modifiée pour accepter les événements émis par les agents, émettre les ordres correspondants vers l'automate, et construire les messages d'état à envoyer vers les agents grâce aux compte-rendus émis par l'automate. Ainsi, à la réception d'une demande de réglage comportant un identifiant de programme, une séquence d'ordres d'assemblages correspondants à ce programme est générée (par exemple, pour du programme spécifiant l'assemblage de deux plaquettes vertes disposées en diagonale, les deux ordres de déplacement correspondants sont générés). Lorsqu'un message de début de cycle est reçu, les ordres générés sont mis en œuvre.

La modification de l'application AutomateConsole consiste donc en l'association à chaque programme d'une liste d'ordres à envoyer à l'automate. Une numérotation judicieuse facilite cette mise en œuvre (par exemple un codage utilisant une décimale pour chacun des quatre emplacement, et un chiffre pour indiquer la couleur). La commande des automates peut être réalisée de la même manière. Pour rendre cette modification fonctionnelle, il faut enfin éditer les règles de routage des événements du BAF pour que les messages destinés aux transformations de forme et d'espace soient dirigés vers l'application de gestion de l'automate.

Les limites de cette approche résultent d'abord de la faible intégration entre l'automate et la station Sun. En effet, le protocole utilisé repose sur l'action directe sur un bloc de données stocké dans la mémoire de l'automate. Il est donc difficile d'ajouter un nouveau type d'échange.

5.5.3 Interaction avec les produits

Pour que le pilotage par le produit soit effectif, il nous reste à mettre en œuvre l'interaction entre les produits et les divers acteurs, en dotant les agents de règles.

L'interaction des produits et des agents ressources peut être réalisée en dotant ces derniers de règles assez semblables à celles présentées à la fin du troisième chapitre : l'agent contrôlant un équipement s'abonne à l'attribut d'état des produits ; lorsqu'un produit passe dans un état

[36] On peut noter d'ailleurs que les capteurs optiques de présence sont facilement accessibles, et qu'il est courant de les déclencher de la main par inadvertance.

où un traitement est requis, l'agent ressource interroge l'agent-produit sur la valeur de certains de ses attributs, et envoie les messages de réglage et de début de cycle appropriés.

Un second type d'interaction concerne la création des agents-produits. La communication avec l'ERP peut se faire à travers le BAF, mais nous préférons ré-utiliser le code de ERPDaemon, pour créer une règle embarquée dans un agent centre de décision. Cette règle observe donc le dossier partagé dédié à la réception des ordres de fabrication : à la réception d'un OF, les demandes de création adaptées sont transmises à l'agent de gestion de la plateforme. Cet agent règle observe de plus les changements d'états des produits, ce qui lui permet de créer une déclaration de production lorsque l'ordre de fabrication est terminé.

On peut adjoindre au système des agents semblables, permettant d'implémenter des fonctions de traçabilité et de personnalisation. Un tel agent peut notifier un client de l'avancement de sa commande (par exemple par courrier électronique), ou encore lui permettre d'agir sur les attributs de l'agent-produit.

Un problème assez délicat est enfin la gestion du *picking*, c'est à dire de l'opération consistant au choix et au déplacement manuel des matières premières pour satisfaire un OF. Les messages d'interaction permettant de déclencher la création d'un produit peuvent être utilisés pour cela. Chaque produit est donc muni d'une règle qui provoque dès que possible l'émission de ce message de création. Ce type de message est routé dans le BAF vers l'application de picking qui extrait l'identifiant du produit à créer, la mémorise, et présente à l'opérateur une interface graphique lui permettant de selectionner un produit à créer. La création se fait par la saisie par l'opérateur à l'aide du lecteur-douchette du code EPC du support devant servir à réaliser le produit. Une association entre le tag EPC et l'identifiant du produit est alors mémorisée pour le routage futur des événements RFID vers les produits.

Ce système permettant le « baptême » des nouveaux produits peut être amélioré en utilisant une règle chargée de sélectionner l'agent-produit dont la production doit commencer.

5.5.4 Résultats

Comme nous l'avons déjà évoqué, ce travail de déploiement d'un pilotage par le produit est en cours. Il est cependant déjà possible de formuler quelques résultats.

Placement des lecteurs RFID

Le premier constat porte sur le placement des équipements d'observation des produits. Alors que dans un système émulé le nombre et le placement des équipements d'identification des produits est choisi par l'expérimentateur, et peut même devenir un facteur expérimental, ces paramètres sont *subis* lors de l'application à un système réel. Dans le cas présent, l'implantation des lecteur RFID a été validée par le fonctionnement correct du système de pilotage originel développé par Sun. Même si dans le cas idéal des lecteurs supplémentaires permettraient une plus grande précision dans le suivi des produits, l'application du système multi-agents de pilotage n'a pas été très difficile.

Le placement des lecteurs était en fait conforme aux besoins du système multi-agents de pilotage. En effet, le protocole de notification qui fonde l'architecture de pilotage par le produit nécessite qu'*au plus un* produit soit à un instant donné dans un état requérant une action. En effet, si deux produits annoncent simultanément qu'ils sont prêts à être traités, sans que leur état

spatial ne soit différent, comment les distinguer ? Le placement des lecteurs doit donc respecter cette règle (figure 5.14).

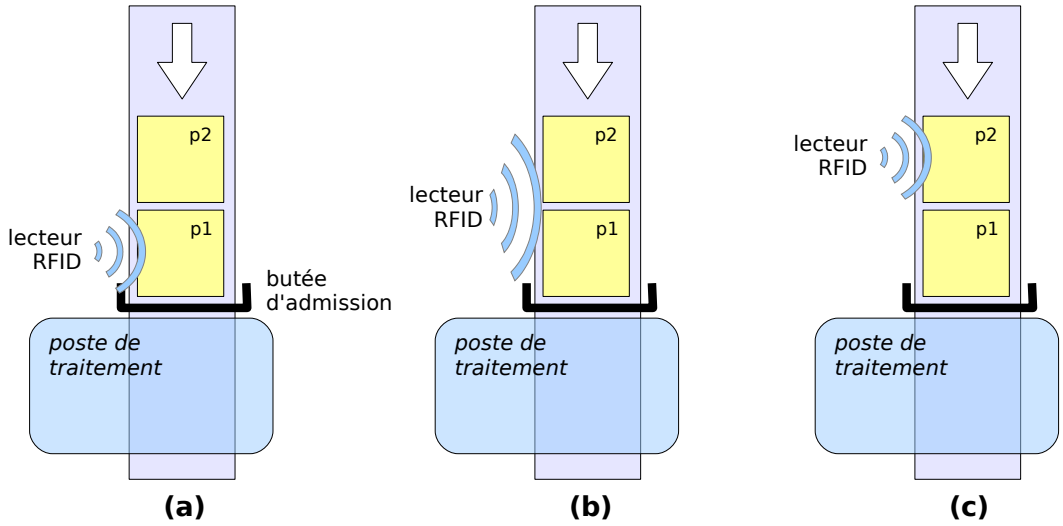


FIG. 5.14 – Placement des équipements identification : (a) seul le produit prêt à être traité est perçu. (b) et (c) d'autres produits sont perçus en plus du produit prêt à être traité

Par contre, les tests que nous avons effectués montrent une faible robustesse des fonctions de suivi des produits lorsque les états successifs des produits sont enchaînés linéairement (figure 5.13). En effet, si un événement ne parvient pas à l'agent, celui-ci reste bloqué et ne prend pas en compte les événements suivants. Les causes possibles d'un tel problème d'acheminement sont multiples :

- les lecteurs partageant le même réseau physique, ils sont scrutés successivement. La période de scrutation est donc relativement élevée, ce qui peut nuire au fonctionnement des lecteurs
- les lecteurs placés au dessus du convoyeur n'ont qu'un temps limité pour lire les puces. La vitesse des convoyeurs et le nombre de tags relatifs à un produit (rappelons que les pastilles et le plaquettes sont elles aussi tagées) renforce ce problème.
- en plus de ces éléments perturbants, lecteurs et tags ne sont pas parfaitement fiables.

Pour éviter ce blocage des produits suite à une lecture manquée, il est possible d'ajouter des transitions permettant de « sauter » un ou deux états. Cette perte relative de l'intégrité de la modélisation se fait au profit d'une robustesse accrue.

Une manière plus rigoureuse de renforcer la robustesse des identifications serait de bloquer les produits sous le lecteur à l'aide d'une butée jusqu'à la réussite de l'identification. Mais même en employant ce moyen comment être sûr de la réussite d'une identification, dans la mesure ou le nombre de tags posés sur un produit est variable : un événement de lecture ne signifie pas que *tous* les tags ont été lus.

Localisation des données relatives aux produits

Un autre point de discussion est la localisation effective des données associées aux produits. Dans le système de pilotage originel développé par Sun, les données associées aux produits sont stockées dans une base de données externe, conformément à l'approche de l'EPC global. Le tag ne porte qu'un pointeur, sous la forme de son code EPC, vers les enregistrements le concernant dans la base.

De la même manière, dans l'approche de contrôle par le produit que nous avons définie, c'est l'agent associé au produit qui est le porteur des données.

Cette approche n'est pas la seule, et peut être discutée : on peut imaginer mémoriser les données à même la puce RFID, pour que la synchronisation entre le produit et ses informations ne dépendent pas d'un système externe.

La mémorisation des données relatives au produit sur l'étiquette RFID nécessite l'écriture sur les tags. Cependant les essais que nous avons pu faire montrent que les temps d'écriture sont relativement élevés (de l'ordre de quelques secondes). Il serai donc nécessaire de bloquer le produit sous le transpondeur RFID pour pouvoir écrire sur la puce.

Cette approche ouvre toutefois des perspectives intéressantes : dans le cadre de notre approche du contrôle par le produit, on peut imaginer stocker la configuration de l'agent-produit sur le tag. L'agent pourrait être extrait du produit, avoir une période d'activité, puis être remis « en hibernation » dans le tag.

Positionner les données relatives au produit sur le tag peut aussi permettre de piloter directement les transformations à partir de ces données. Néanmoins, cela implique une intégration beaucoup plus forte entre les équipements RFID et les automatismes. En effet, dans la configuration actuelle les informations en provenance des lecteurs RFID « remontent » vers le middleware avant d'être envoyées vers l'automate. Pour qu'il soit pertinent que les informations du produit soient lues directement par l'automate, une communication directe entre ces deux équipements est requise.

Les premiers résultats de déploiement sont la possibilité de synchroniser l'état d'un agent logiciel avec l'état physique du produit. La principale limite mise en évidence par nos travaux est le manque d'intégration entre les équipements RFID et les automatismes. Au delà des contingences techniques, ce constat est révélateur du fossé qui sépare deux corps de métiers : l'automatisation, et l'ingénierie des systèmes d'information d'entreprise. Pour faciliter la suite du déploiement du pilotage par le produit, il nous faudra combler au mieux ce fossé.

5.6 Conclusion

Ce chapitre a d'abord présenté l'application de l'environnement d'évaluation à un cas industriel, puis l'utilisation de la plateforme d'essais dans le cadre de l'ERT TRACILog. L'utilisation conjointe de ces deux outils de test permet de couvrir l'ensemble des problématiques relatives à la mise en oeuvre industrielle d'un système de pilotage par le produit.

Les applications, tant au cas industriel présenté qu'à la plateforme d'essais, sont un travail en cours. Cependant, l'architecture semble jusque là capable de fournir les fonctions attendues pour la validation d'un pilotage par le produit : modélisation et évaluation des procédures de décision dans la phase de définition, modélisation et évaluation de l'architecture du système de pilotage en phase de développement, étude des aspects techniques à l'aide de la plateforme d'essais.

L'application du pilotage par le produit à la plateforme d'essais permettra de plus de faire la démonstration —à petite échelle— de fonctionnalités de traçabilité individuelle des ordres de fabrication, gestion d'une forte variété des produits, modification possible des commandes jusqu'a la transformation physique, etc...

Les travaux de modélisation et d'évaluation réalisés avec nos partenaires industriels vont continuer dans le cadre de l'ERT, conduisant à des modèles plus concrets, des protocoles et des résultats expérimentaux bien définis.

Nous prévoyons aussi de continuer les travaux de déploiement du pilotage par le produit à la plateforme, et nous espérons obtenir rapidement plus de résultats. Cette application dans une situation concrète ouvre sur de nombreuses expériences portant sur la manière de mettre en œuvre le contrôle par le produit : on peut par exemple mener des expériences sur l'impact du placement ou du nombre des lecteurs, sur l'impact de leur non-fiabilité en perturbant artificiellement les lectures, etc... Mais pour mener à bien ces expériences, il nous faudra d'abord définir un ensemble d'indicateurs de performance quantitatifs, ce qui peut ne pas être évident.

Conclusion

Cette thèse a proposé un environnement de modélisation et de test d'architectures de pilotage par le produit.

Le pilotage par le produit permet de répondre à la problématique industrielle de l'hybridation entre décisions centralisées et décisions distribuées. Cette hybridation est rendue nécessaire par le besoin de répondre à des exigences de traçabilité, de gestion de la variété des produits et de robustesse par rapport à des conditions de fonctionnement incertaines conduisant à une approche distribuée de la prise de décision, mais aussi à des exigences visant à inscrire le pilotage de manière cohérente et performante dans un système global de gestion conduisant à une approche centralisée de la prise de décision.

Le concept de contrôle par le produit permet d'articuler ces deux visions du pilotage, tout en fournissant un schéma d'organisation clair, facilitant la conception et la maintenance des systèmes d'information de l'entreprise dédiés au pilotage.

Se pose cependant la problématique de conception d'architectures concrètes de pilotage par le produit. En particulier, la modélisation et le test de ces architectures, afin d'étudier leur pertinence par rapport à des approches classiques de pilotage, ou de guider le concepteur dans le choix des modalités de mise en oeuvre du pilotage par le produit.

Pour répondre à cette problématique, nous nous sommes dans un premier temps munis d'un environnement de modélisation et d'évaluation du pilotage par le produit, que nous avons dans un second temps appliqué à des cas d'étude, notamment dans le contexte de l'ERT Tracilog.

La première partie de la thèse a donc présenté en deux chapitres la définition, le développement et la validation de cet environnement d'évaluation composé de :

- une bibliothèque de composants d'émulation supportant la création et l'exécution de modèles d'atelier virtuels, d'échelles réalistes.
- un ensemble de composants logiciels orientés-agents facilitant le développement et l'étude d'architectures de pilotage par le produit.

Ces deux éléments, utilisés conjointement ou séparément, permettent l'analyse des performances d'un système de pilotage par le produit et sa comparaison à d'autres systèmes de pilotage utilisés dans les approches classiques.

Dans le seconde partie, l'environnement de modélisation et de test a d'abord été appliqué à un cas industriel, choisi pour les problèmes de pilotage qu'il pose. Un plan d'expériences a permis l'étude des performances relatives de différentes modalités de pilotage par le produit et d'un pilotage centralisé, dans différents scénarios de perturbation. Ensuite, nous avons présenté la manière dont l'environnement d'évaluation, utilisé conjointement avec une plateforme d'essais, permettait d'assister au transfert du pilotage par le produit vers les industries du bois et des fibres, dans le cadre de l'équipe de recherche technologique TRACILog.

Ces expériences permettent d'éprouver et de valider la faisabilité du concept de pilotage par le produit en terme d'impact décisionnel (ré-ordonnancement local, cohérence entre l'état physique et informationnel du produit, ...) ainsi qu'en terme de contraintes techniques (fiabilité de l'identification produits, défauts du réseau de communication, ...).

Les travaux effectués dans le cadre de cette thèse ont permis de contribuer scientifiquement et techniquement à la définition, au développement et à l'évaluation du rôle du pilotage par le produit dans l'interaction centralisé/distribué.

D'abord, nous avons contribué à définir plus précisément le concept de pilotage par le produit, en le rattachant au mécanisme fondamental d'intelligence artificielle qu'est la stigmergie. Nous avons de plus présenté une manière dont ce concept pouvait être mis en œuvre dans le cadre des modèles logiques des normes FIPA, en particulier grâce au protocole d'abonnement, qui confère au produit un rôle relativement actif dans la prise de décision.

Ensuite, nous avons contribué à l'évaluation par émulation des systèmes distribués, en proposant une méthodologie de création des modèles d'émulation basée sur des primitives génériques de modélisation. Un protocole d'expérimentation dont les phases sont inspirées de la mécatronique (*software in the loop*, *hardware in the loop*) a, de plus, été défini. D'un point de vue plus technique, nous avons développé le prototype d'un logiciel permettant la modélisation et l'exécution des modèles d'émulation.

Enfin, les expériences que nous avons effectuées ont permis d'évaluer la faisabilité du contrôle par le produit. Le plan d'expériences mené sur un cas industriel a montré que le pilotage par le produit permet effectivement de combiner les avantages en terme de productivité des approches centralisées et distribuées. Ce plan d'expériences a donc permis d'évaluer la pertinence de ce concept de pilotage. Le déploiement du pilotage par le produit sur la plateforme d'essai de l'ERT Tracilog, bien qu'encore en cours, permettra pour sa part d'évaluer la faisabilité technique du pilotage par le produit.

Les limites de nos propositions s'expriment sur le plan de l'outillage, et sur le plan scientifique. L'un des objectifs de la thèse était de fournir un outil d'ingénierie pour la modélisation et l'évaluation. Nos travaux ont permis de définir et de développer des prototypes. Le développement de ces prototypes doit se poursuivre afin d'améliorer leur facilité d'utilisation et leur stabilité. De même, les travaux d'application doivent être complétés pour fournir des résultats expérimentaux plus complets et plus cohérents.

Du point de vue scientifique, nous avons adopté une approche par composants pour modéliser et tester les architectures de pilotage : nous disposons d'un ensemble de composants génériques, mais une démarche de modélisation plus approfondie nous permettrait de spécifier les méta-modèles qui sous-tendent notre approche. Les composants d'émulation, ont atteint une maturité suffisante pour qu'une telle démarche soit rapidement envisagée. Les composants multi-agents de modélisation des architectures décisionnelles nécessitent quant à eux encore une période de maturation, permettant de s'abstraire des aspects techniques, qui posent encore problème.

Enfin, nous nous sommes restrein dans cette thèse aux problématiques d'évaluation. La conception des architectures de contrôle par le produit reste un travail difficile, comme l'est en général la conception de systèmes de décision distribués. Les possibilités en matière d'évaluation et d'expérimentation facilitent la conception, mais on ne peut nier le besoin de spécifier des méthodes d'ingénieries permettant de guider la définition d'un pilotage par le produit.

Les perspectives de cette thèse peuvent être énoncées par rapports aux problèmes clés du domaine de la conception et de l'expérimentation des systèmes intelligents de production (*IMS modeling and experiments* [Morel *et al.*, 2007] :

- le manque d'outils et de plateformes d'échelles réalistes pour valider les développements des systèmes de pilotage intelligents,
- une meilleure maîtrise des problématiques de déploiement à grande échelle (*scalability*) et de robustesse grâce à l'utilisation de l'émergence et de l'auto-organisation,
- la gestion de l'information dans les systèmes intelligents, en particulier par rapport aux problématiques de traçabilité.

Nos travaux étaient principalement dirigés vers le premier point. Au delà de l'environnement d'évaluation que nous avons proposé et de ses améliorations futures, il nous faudra collecter un nombre suffisant de cas d'étude émulés, les sélectionner et les classer en fonction des problèmes de pilotage qu'ils comportent, afin de constituer une bibliothèque de benchmarking. Pour éviter la dispersion des efforts de modélisation, on peut envisager de définir un format neutre permettant l'échange de modèles entre les différentes plateformes d'émulation qui voient le jour.

Par rapport au deuxième point, nous avons dans cette thèse volontairement choisi de ne pas le traiter. Cependant, le manque de composants auto-organisables, ou de méthodes facilitant la création de tels composants s'est fait plusieurs fois sentir, nous contraignant à formuler « manuellement » des algorithmes de décision dont la qualité n'était pas forcément optimale. La définition de tels composants capables par intelligence artificielle d'auto-organisation constitue donc un champ de recherche primordial pour le développement des systèmes contrôlés par le produit. Une autre possibilité est de s'orienter vers une prise de décision anthropocentrique pour utiliser l'intelligence « naturelle » présente sur le terrain à travers les opérateurs.

De la même manière, bien que le paradigme de contrôle par le produit propose une organisation facilitant la traçabilité des produits, il reste à définir la nature des informations portés par le produit. Une telle définition, par exemple sous la forme d'une ontologie, est requise pour le développement futur du contrôle par le produit, en particulier pour qu'un système de pilotage par le produit puisse s'interfacer facilement avec les systèmes d'information de l'entreprise.

Enfin, dans le cadre de l'ERT Tracilog (qui a financé ma dernière année de thèse, et dans le cadre de laquelle je vais continuer mes travaux en tant qu'ingénieur de recherche) les perspectives de cette thèse s'axent autour de deux projets. D'abord, il s'agit de continuer le déploiement du pilotage par le produit sur la plateforme d'essais, et d'assister à la modélisation et l'évaluation du pilotage par le produit pour les partenaires industriels de L'ERT. Nous pourrons ainsi valider les approches proposées de pilotage par le produit, ou encore évaluer l'impact des contraintes techniques sur ces modèles. Le second projet porte sur l'amélioration de l'environnement d'évaluation par émulation. Le but est de rendre cet outil plus riche, plus robuste et plus facile d'utilisation, afin qu'il soit distribuable. Ainsi, une ouverture du développement pourrait potentiellement permettre de mutualiser les efforts, et permettre la constitution d'une communauté d'utilisateurs et de développeurs.

Bibliographie

[Albus et Barbera, 2005] Albus, J. S. et Barbera, A. J. (2005). Rcs : A cognitive architecture for intelligent multi-agent systems. *Annual Reviews in Control*, 29(1):87–99.

[Artigues *et al.*, 2005] Artigues, C., Billaut, J.-C. et Esswein, C. (2005). Maximization of solution flexibility for robust shop scheduling. *European Journal of Operational Research*, 165(2):314–328.

[Baina, 2006] Baina, S. (2006). *Interopérabilité dirigée par les modèles : une approche orientée produit pour l'interopérabilité des systèmes d'entreprise.* Thèse de doctorat, Université Henri Poincaré.

[Bajic et Chaxel, 2002] Bajic, E. et Chaxel, F. (2002). Auto-id mobile information system for vehicle life cycle data management. *In IEEE International Conference on Systems, Man and Cybernetics*, volume 4, pages 6 pp. vol.4+.

[Bellifemine *et al.*, 2001] Bellifemine, F., Poggi, A. et Rimassa, G. (2001). Developing multi-agent systems with a fipa-compliant agent framework. *Software : Practice and Experience*, 31(2):103–128.

[Bishop, 2005] Bishop, R. (2005). *Mechatronics : An Introduction.* CRC.

[Bitran et Hax, 1981] Bitran, G. R. et Hax, A. C. (1981). Disaggregation and resource allocation using convex knapsack problems with bounded variables. *Management Science*, 27(4):431–441.

[Blanc, 2006] Blanc, P. (2006). *Pilotage par l'approche holonique d'un système de production de vitres de sécurité feuilletées.* Thèse de doctorat, École Centrale de Nantes et l'Université de Nantes.

[Blanc *et al.*, 2006] Blanc, P., Demongodin, I. et Castagna, P. (2006). A holonic approach for manufacturing control : an industrial application. *In 12th IFAC Symposium on Information Control Problems in Manufacturing INCOM'2006.*

[Bongaerts *et al.*, 2000] Bongaerts, L., Monostori, L., Mcfarlane, D. et Kadar, B. (2000). Hierarchy in distributed shop ffoor control. *Computers in Industry*, 43(2):123–137.

[Bousbia, 2006] Bousbia (2006). *Proposition d'une architecture logique d'un système de pilotage hétérarchique évolutif par apprentissageBousbia-2006-.* Thèse de doctorat, Université de Valenciennes et du Hainaut-Cambrésis.

[Brennan, 2000] Brennan, R. W. (2000). Performance comparison and analysis of reactive and planning-based control architectures for manufacturing. *Robotics and Computer-Integrated Manufacturing*, 16(2-3):191–200.

[Brennan et Norrie, 2001] Brennan, R. W. et Norrie, D. H. (2001). Evaluating the performance of reactive control architectures for manufacturing production control. *Computers in Industry*, 46(3):235–245.

[Brennan et Norrie, 2003] Brennan, R. W. et Norrie, D. H. (2003). Metrics for evaluating distributed manufacturing control systems. *Computers in Industry*, 51(2):225–235.

[Bussmann et Schild, 2001] Bussmann, S. et Schild, K. (2001). An agent-based approach to the control of flexible production systems. *In Proc. of the 8th IEEE Int. Conf. on Emergent Technologies and Factory Automation (ETFA 2001)*, pages 481–488, Antibes Juan-les-pins, France.

[Cavalieri *et al.*, 2000] Cavalieri, S., Garetti, M., Macchi, M. et Taisch, M. (2000). An experimental benchmarking of two multi-agent architectures for production scheduling and control. *Computers in Industry*, 43(2):139–152.

[Cavalieri *et al.*, 2003] Cavalieri, S., Macchi, M. et P, V. (2003). Benchmarking the performance of manufacturing control system : design principles for a web based simulated testbed. *Journal of Intelligent Manufacturing*, 14(1):43–58.

[Cea, 2006] Cea, A. (2006). *Contribution à la Modélisation et à la Gestion des Interactions Produit-Processus dans la Chaîne Logistique par l'Approche Produits Communicants.* Thèse de doctorat, Université Henri Poincaré.

[Corbier, 1989] Corbier, F. (1989). *Modélisation et émulation de la partie opérative pour recette en plateforme d'équipement automatisés.* Thèse de doctorat, Université de Nancy I.

[Doumeingts, 1984] Doumeingts, G. (1984). Méthode grai, méthode de conception et de spéciflcation des systèmes de productique. Thèse de Docteur d'état.

[El Haouzi *et al.*, 2007] El Haouzi, H., Thomas, A. et Pétin, J.-F. (2007). Contribution to reusability and modularity of manufacturing systems simulation models : application to distributed control simulation within dft context. *International Journal of Production Economics.*

[Feliot, 1997] Feliot, C. (1997). *Modélisation de systèmes complexes : intégration et formalisation de modèles.* Thèse de doctorat, Université de Lille 1, Villeneuve-d'Ascq.

[FIPA, 2005] FIPA (2005). The foundation for intelligent physical agents. http :/www.fipa.org/repository/index.html.

[Fusaoka *et al.*, 1983] Fusaoka, A., Seki, H. et Takahashi, K. (1983). A description and reasoning of plant controllers in temporal logic. *In International Joint Conference on Artiflcial Intelligence*, pages 405–408.

[Goldin et Wegner, 2005] Goldin, D. et Wegner, P. (2005). The church-turing thesis : Breaking the myth. *Lecture Notes in Computer Science : New Computational Paradigms*, 3593:152–168.

[Gouyon, 2004] Gouyon, D. (2004). *Contrôle par le produit des systèmes d'exécution de la production : apport des techniques de synthèse.* Thèse de doctorat, Université Heri Poincaré, Nancy I.

[Grabot *et al.*, 2005] Grabot, B., Geneste, L., Reynoso-Castillo, G. et Verot, S. (2005). Integration of uncertain and imprecise orders in the mrp method. *Journal of Intelligent Manufacturing*, 16(2):215–234.

[Grassé, 1959] Grassé, P. (1959). La reconstruction du nid et les coordinations interindividuelles chez bellicositermes natalensis et cubitermes sp. la théorie de la stigmergie : Essai d'interprétation des termites constructeurs. *Insectes Sociaux*, 6:41–84.

[Génin, 2003] Génin, P. (2003). *Planification tactique robuste avec usage d'un APS. Proposition d'un mode de gestion par plan de référence.* Thèse de doctorat, Ecole des mines de Paris.

[Hadeli *et al.*, 2005] Hadeli, K., Valckenaers, P., Germain, S. B., Verstraete, P., Zamfl-rescu, C. B. et Van Brussel, H. (2005). Emergent forecasting using a stigmergy approach in manufacturing coordination and control. *Engineering Self-Organising Systems : Methodologies and Applications*, 3464.

[ISO/IEC, 2004] ISO/IEC (2004). 62264, "enterprise–control system integration", part 1. models and terminology, part 2. object models attributes, part 3. activity models of manufacturing operations management, tc184/sc5.

[Iung *et al.*, 2001] Iung, B., Neunreuther, É. et Morel, G. (2001). Engineering process of integrated - distributed shop floor architecture based on interoperable field components. *International Journal of Computer Integrated Manufacturing*, 14(3):246–262.

[Karkkainen, 2003] Karkkainen, M. (2003). Increasing efficency in the supply chain for short shelf life goods using rfid tagging. *International Journal of Retail & Distribution Management*, 31(10):529–536.

[Klein et Thomas, 2006] Klein, T. et Thomas, A. (2006). Développement d'un modèle de simulation pour l'évaluation des systèmes de pilotage distribués. *In 6ème Conférence Francophone de Modélisation et Simulation Modélisation, MOSIM'06, Rabat, Maroc.*

[Koestler, 1976] Koestler, A. (1976). *The Ghost in the Machine.* Hutchinson & Co Ltd., London, UK.

[Kumar *et al.*, 2004] Kumar, V., Leonard, N. et Morse, A. S. (2004). *Cooperative Control : A Post-Workshop Volume : 2003 Block Island Workshop on Cooperative Control (Lecture Notes in Control and Information Sciences).* Springer.

[Le Gallou et Bouchon-Meunier, 1992] Le Gallou, F. et Bouchon-Meunier, B. (1992). *Systemique : theorie et applications.* Tech.& Doc., Lavoisier.

[Le Moigne, 1977] Le Moigne, J. L. (1977). *La théorie du système général. Théorie de la modélisation.* Presse Universitaires de France. réédition de 1994. ISBN 2130384838.

[Leitão et Restivo, 2005] Leitão, P. et Restivo, F. (2005). Experimental validation of adacor holonic control system. *Lecture Notes in Computer science : Holonic and Multi-Agent Systems for Manufacturing*, 3593:121–132.

[Marik et Lazansky, 2007] Marik, V. et Lazansky, J. (2007). Industrial applications of agent technologies. *Control Engineering Practice*, In Press, Corrected Proof.

[Marshall *et al.*, 2006] Marshall, J. A., Broucke, M. E. et Francis, B. A. (2006). Pursuit formations of unicycles. *Automatica*, 42(1):3–12.

[Mcfarlane *et al.*, 2003] Mcfarlane, D., Sarma, S., Chirn, J. L., Wong, C. Y. et Ashton, K. (2003). Auto id systems and intelligent manufacturing control. *Engineering Applications of Artificial Intelligence*, 16(4):365–376.

[Monch, 2006] Monch, L. (2006). Simulation-based benchmarking of production control schemes for complex manufacturing systems. *Control Engineering Practice*, In Press, Corrected Proof.

[Morel, 2006] Morel, G. (2006). Cours du Master EEAPR-IS de l'université Henri Poincaré, Nancy 1.

[Morel *et al.*, 2007] Morel, G., Valckenaers, P., Faure, J.-M., Pereira, C. E. et Dietrich, C. (2007). Manufacturing plan control chanllenges and isssues. à paraitre.

[Ortiz-Araya, 2005] Ortiz-Araya, V. (2005). *Proposition d'un modèle de désagrégation pour un plan tactique stable dans le contexte des chaînes logistiques et de l'usage d'un APS.* Thèse de doctorat, Université Henry Poincaré.

[Pannequin et Thomas, 2004] Pannequin, R. et Thomas, A. (2004). Centralized versus distributed decision, an industrial case. *In 11th IFAC Symposium on Information and Control Problem in Manufacturing INCOM'2004*.

[Pannequin et Thomas, 2006] Pannequin, R. et Thomas, A. (2006). Cooperation between business and holonic manufacturing decision systems. *In 12th IFAC Symposium on Information Control Problems in Manufacturing INCOM'2006*, pages 423–428.

[Parlikad et Mcfarlane, 2007] Parlikad, A. K. et Mcfarlane, D. (2007). Rfid-based product information in end-of-life decision making. *Control Engineering Practice*, In Press, Corrected Proof.

[Parsopoulos et Vrahatis, 2002] Parsopoulos, K. E. et Vrahatis, M. N. (2002). Recent approaches to global optimization problems through particle swarm optimization. *Natural Computing*, V1(2):235–306.

[Penalva, 1997] Penalva, J.-M. (1997). *La modélisation par les systèmes en situations complexes*. Thèse de doctorat, Université Paris Sud, UFR scientifique d'Orsay.

[Pfeiffer *et al.*, 2003] Pfeiffer, A., Kádár, B. et Monostori, L. (2003). Evaluating and improving production control systems by using emulation. *In Applied Simulation and Modelling*.

[Plossl, 1985] Plossl, G. W. (1985). *Production and Inventory Control : Principles and Techniques (2nd Edition)*. Prentice Hall.

[Pétin *et al.*, 2006] Pétin, J.-F., Morel, G. et Panetto, H. (2006). Formal specification method for systems automation. *European Journal of Control*, 12(2):115–130.

[Rennard et Mange, 2002] Rennard, J.-P. et Mange, D. (2002). *Introduction à la vie artiflcielle*. Vuibert.

[Sahin, 2005] Sahin, e. (2005). *A qualitative and quantitative analysis of the impact of RFID technology on the performance of supply chains*. Thèse de doctorat, École Centrale Paris.

[Saint Germain *et al.*, 2003] Saint Germain, B., Valckenaers, P., Zamflrescu, C., Bochmann, O. et Vanbrussel, H. (2003). Benchmarking of manufacturing control systems in simulation. *In Proc. of the 3rd Int'l Workshop on Performance Measurement (IflP WG5.7), Bergamo*, pages 357–369.

[Simao, 2005] Simao, J. (2005). *Contribution au Développement d'un Outil de Simulation de Systèmes Holoniques de Production et Proposition d'un Meta-Modèle de Contrôle Holonique*. Thèse de doctorat, Université Henri Poincaré & Centro Federal de Educação Tecnológica do Paraná (CEFET-PR) - Brasil.

[Simao *et al.*, 2006] Simao, J. M., Stadzisz, P. C. et Morel, G. (2006). Manufacturing execution systems for customized production. *Journal of Materials Processing Technology*, 179(1-3):268–275.

[Simon, 1996] Simon, H. A. (1996). *The Sciences of the Artiflcial - 3rd Edition*. The MIT Press.

[Simon, 1997] Simon, H. A. (1997). *Administrative Behavior, 4th Edition*. Free Press.

[Smith, 1980] Smith, R. G. (1980). The contract net protocol : High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, 29(12):1104–1113.

[Souquières et Thuan Truong, 2006] Souquières, J. et Thuan Truong, N. (2006). Verification of uml model elements using b. *Journal of Information Science and Engineering*, 22:357–373.

[Tixador, 1989] Tixador, J.-M. (1989). *Une contribution au génie automatique : la spécifIcation exécutable des machines et systèmes automatisés de production*. Thèse de doctorat, Université de Nancy I.

[Tranvouez *et al.*, 2006] Tranvouez, E., Ferrarini, A. et Espinasse, B. (2006). Cooperative disruption management in industrial systems : a multiagent approach. *In* Dolgui, A., Morel, G. et Pereira, C. E., éditeurs : *12th IFAC Symposium on Information Control Problems in Manufacturing, St Etienne, France, 17–19 Mai*.

[Valckenaers, 2001] Valckenaers, P. (2001). Editorial of the special issue on holonic manufacturing systems. *Computers in Industry*, 46(3):233–234.

[Valckenaers *et al.*, 2006a] Valckenaers, P., Cavalieri, S., Germain, B., Verstraete, P., Hadeli, Bandinelli, R., Terzi, S. et Brussel, H. (2006a). A benchmarking service for the manufacturing control research community. *Journal of Intelligent Manufacturing*, 17(6):667–679.

[Valckenaers *et al.*, 2006b] Valckenaers, P., Hadeli, Saint Germain, B., Verstraete, P. et Van Brussel, H. (2006b). Emergent short-term forecasting through ant colony engineering in coordination and control systems. *Advanced Engineering Informatics*, 20(3):261–278.

[van Brussel *et al.*, 1998] van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L. et Peeters, P. (1998). Reference architecture for holonic manufacturing systems : Prosa. *Computers in Industry*, 37(3):255–274.

[van der Zee, 2006] van der Zee, D. J. (2006). Modeling decision making and control in manufacturing simulation. *International Journal of Production Economics*, 100(1):155–167.

[Van Dyke Parunak, 1993] Van Dyke Parunak, H. (1993). Mascot : A virtual factory for research and development in manufacturing scheduling and control. Rapport technique, ITI Tech Memo 93-02.

[Vollmann *et al.*, 1997] Vollmann, T. E., Berry, W. L. et Whybark, C. D. (1997). *Manufacturing Planning and Control Systems*. McGraw-Hill.

[Wegner, 1997] Wegner, P. (1997). Why interaction is more powerful than algorithms. *Communications of the ACM*, 40(5):80–91.

[Wong *et al.*, 2002] Wong, C., McFarlane, D., Ahmad Zaharudin, A. et Agarwal, V. (2002). The intelligent product driven supply chain. *In IEEE int. conference on System Man and Cybernetic SMC'02*.

Annexe A : Modèles d'implémentation des composants d'émulation

Cette annexe présente les modèles d'implémentation des modules Arena. Le définition d'un modules avec Arena se fait selon plusieurs vues :

- la vue *opérandes*, dans laquelle l'interface de paramétrage du module est définie,
- la vue *logique*, qui défini le comportement du module, grâce à un réseau de bloc SIMAN,
- la vue *interrupteurs* qui permet de définir des variables booléennes servant à définir plusieurs mode de fonctionnement du module,
- les vues *user view* et *panel icon* qui servent à éditer l'aspect graphique du module.

Les opérandes du modules sont représentés dans la vue logique entre backquotes (par exemple ‘Name‘), tandis que les interrupteurs sont représenté entre crochets.

Nous présentons dans cette annexe les modèle d'opérandes et les modèles logiques pour les modules principaux :

- modules de création (figures 1 et 2),
- module de transformation de temps (figures 3 et 4, table 1),
- module de transformation d'espace (figures 5 et 6),
- module de transformation de forme (figures 7 et 8),

Enfin, nous présenterons un diagramme UML de classes présentant l'implémentation de la bibliothèque de fonctions (DLL) accompagnant les modules Arena.

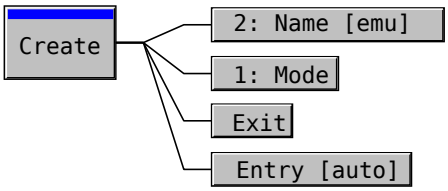


Fig. 1 – Modules Arena de création : vue opérandes.

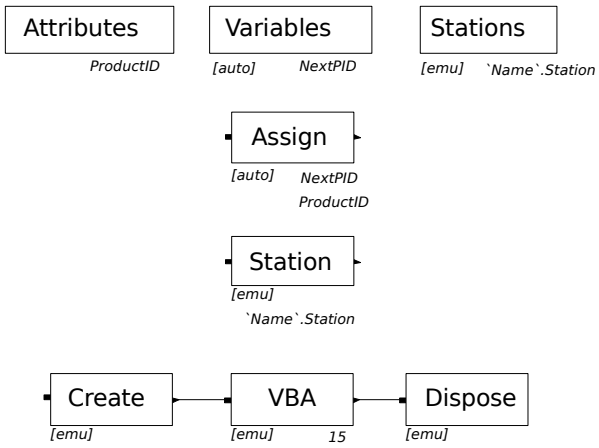


FIG. 2 – Modules Arena de création : vue logique.

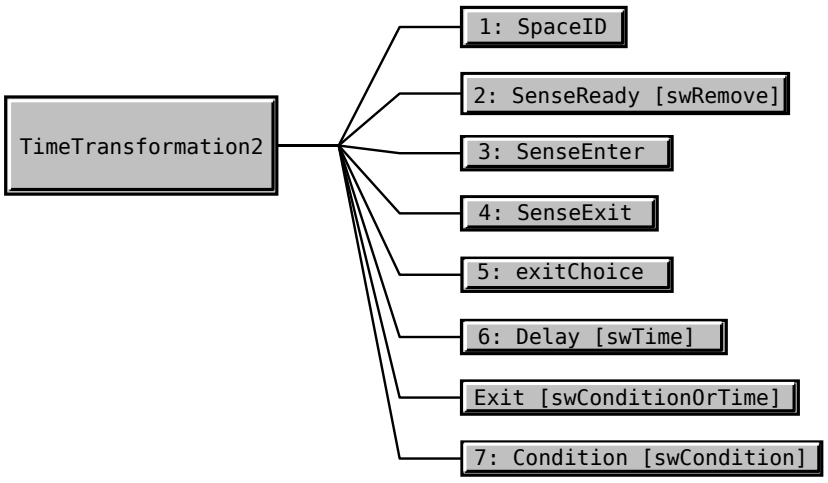


FIG. 3 – Module Arena de transformation de temps : opérandes

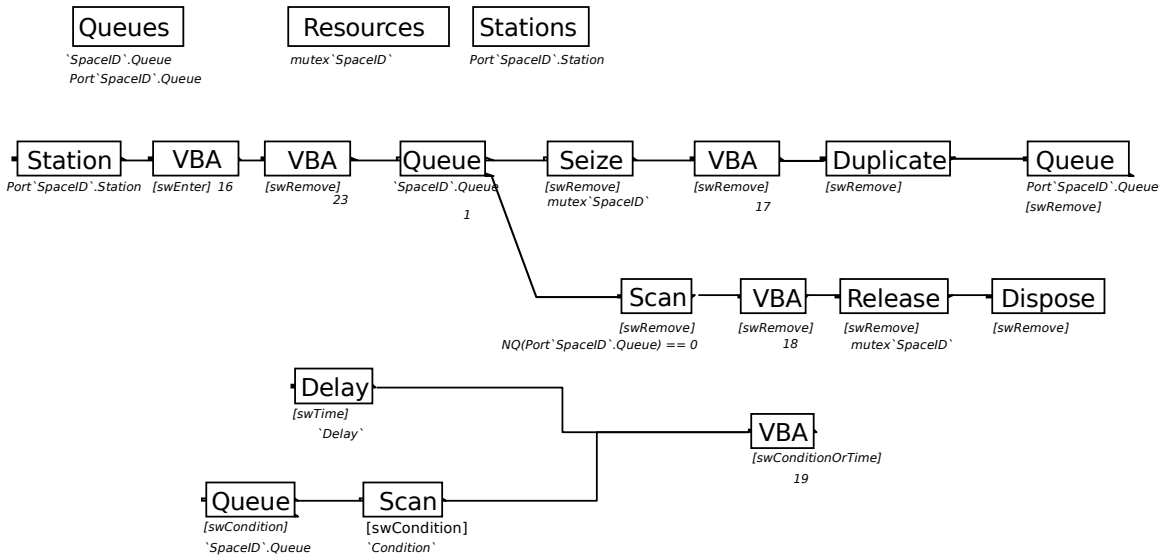


FIG. 4 – Module Arena de transformation de temps : modèle logique.

```
Private Sub VBA_Block_16_Fire()
'product enter in TimeTransform 2
'generated from emulation template
'do not edit following code

    'send Product event
    sendProductEvent "enter‘SpaceID‘"

End Sub

Private Sub VBA_Block_17_Fire()
'ready event in time transform 2
'generated from emulation template
'do not edit following code

    'send Product event
    sendProductEvent "ready‘SpaceID‘"
End Sub

Private Sub VBA_Block_18_Fire()
'exit event in timetransform 2
'generated from emulation template
'do not edit following code

    'send Product event
    sendProductEvent "exit‘SpaceID‘"

    'send TimeTransform Event
    On Error Resume Next
    Main.emu.TimeTransformEvent "‘SpaceID‘", "exit"
    On Error Resume Next
    Main.TimeTransformEvent "‘SpaceID‘", "exit"
End Sub
```

```
Private Sub VBA_Block_19_Fire()
'exit event in timetransform 2
'generated from emulation template
'do not edit following code

    'send Product event
    sendProductEvent "exit‘SpaceID‘"

    'send TimeTransform Event
    On Error Resume Next
    Main.emu.TimeTransformEvent "‘SpaceID‘", "exit"
    On Error Resume Next
    Main.TimeTransformEvent "‘SpaceID‘", "exit"
End Sub

Private Sub VBA_Block_20_Fire()
'ready event in time transform 2
'generated from emulation template
'do not edit following code

    'send Product event
    sendProductEvent "ready‘SpaceID‘"
End Sub

'enter event on time Transformation 2
Private Sub VBA_Block_23_Fire()
    'generated from emulation template
    'do not edit following code

    'send TimeTransform Event
    On Error Resume Next
    Main.emu.TimeTransformEvent "‘SpaceID‘", "enter"
    On Error Resume Next
    Main.TimeTransformEvent "‘SpaceID‘", "enter"
End Sub
```

TAB. 1 – Fonctions associées au blocs VBA (en Visual Basic 6) du module de transformation de temps

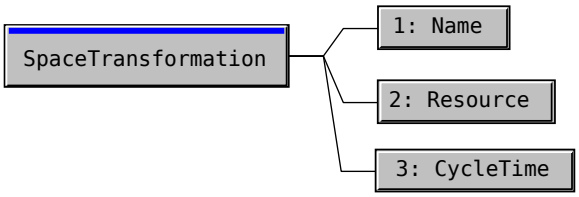


FIG. 5 – Module Arena de transformation d'espace : opérandes

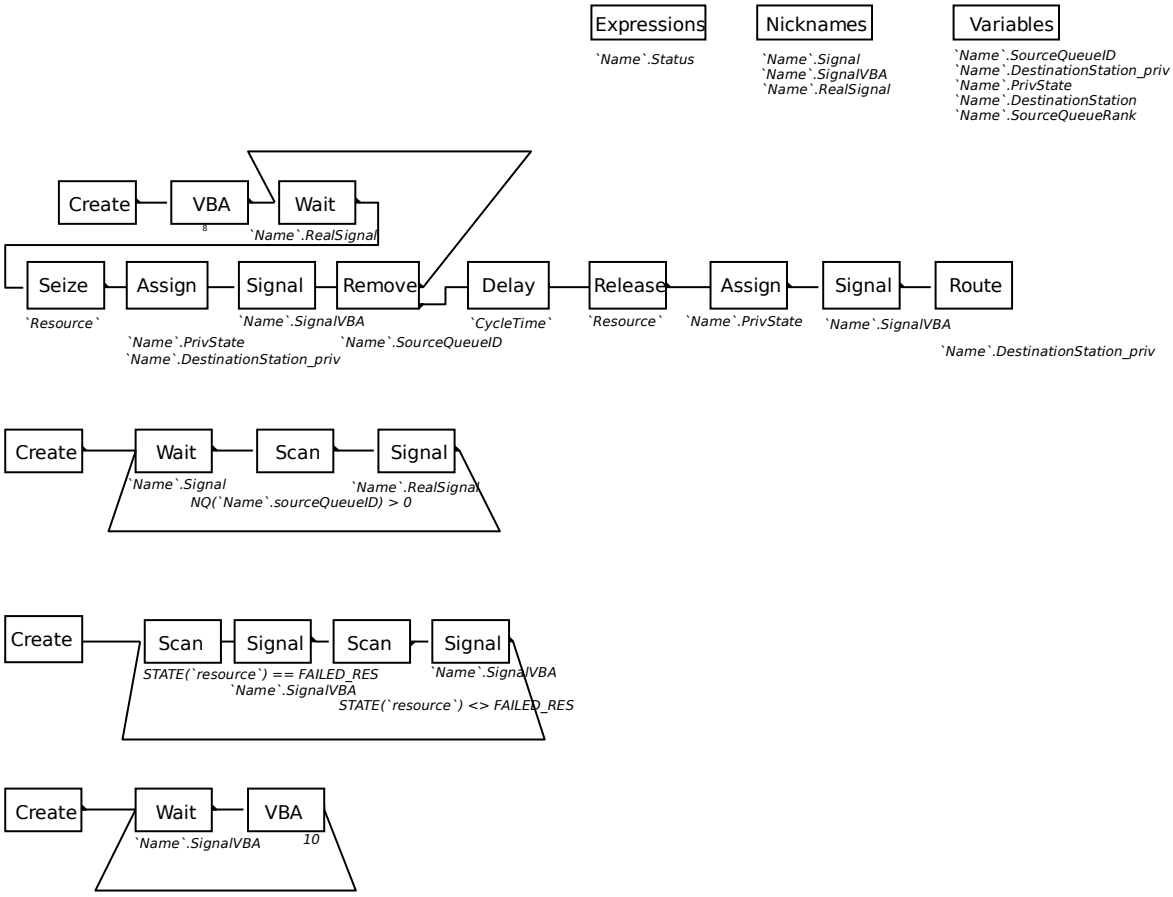


FIG. 6 – Module Arena de transformation d'espace : modèle logique.

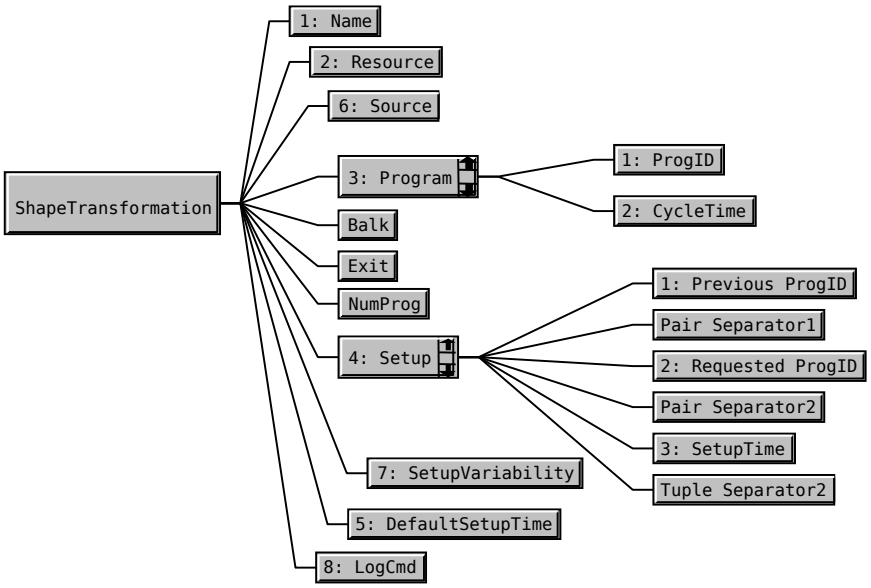


FIG. 7 – Module Arena de transformation d'espace : opérandes

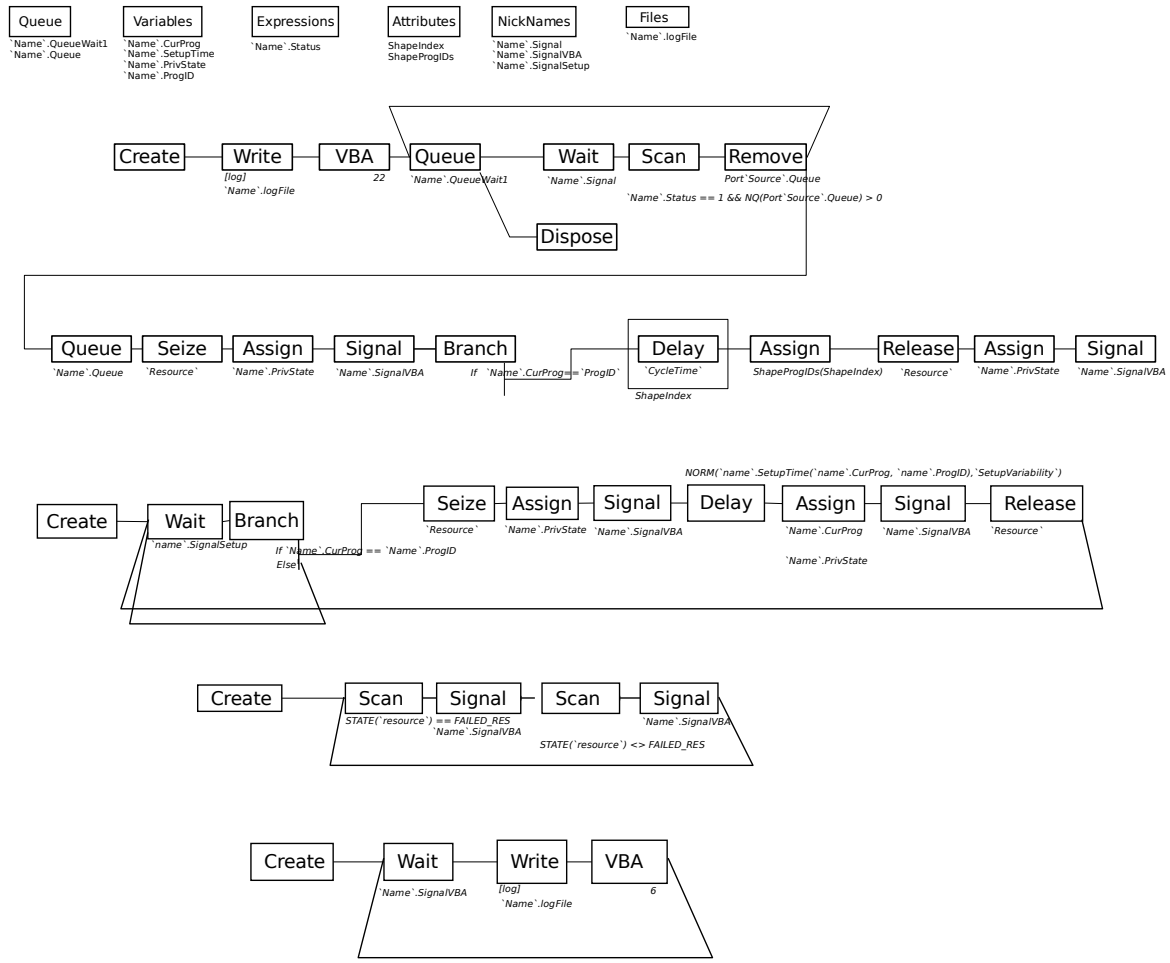


FIG. 8 – Module Arena de transformation de forme : modèle logique

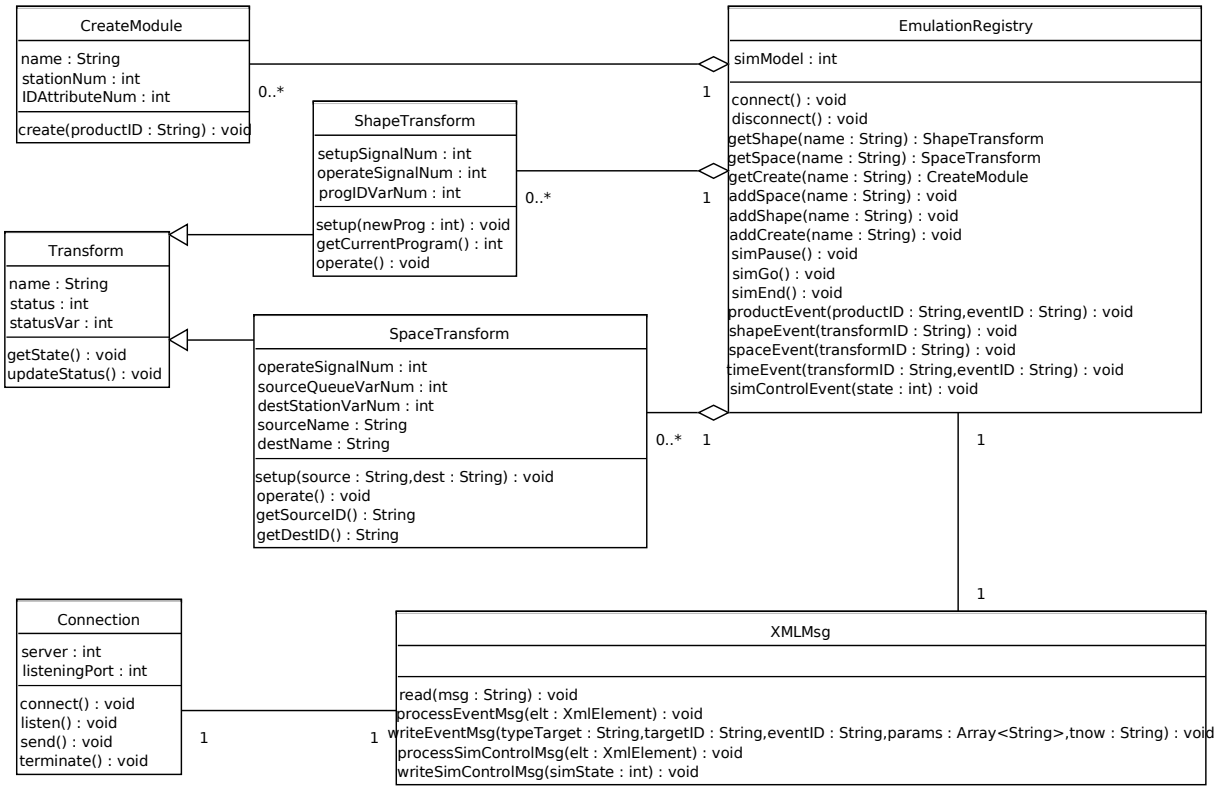


FIG. 9 – Diagramme de classes UML : structure de la bibliothèque de fonctions (DLL) adjointe aux modules Arena

Résumé

Le développement des technologies infotroniques, qui permettent de faire porter numériquement des données par le produit afin de lui conférer un rôle actif dans la boucle cybernétique, conduisent à remettre en question l'organisation conventionnelle des systèmes de pilotage, pour s'orienter vers un pilotage par le produit. Il existe un large consensus sur l'intérêt de cette approche dans la prise de décision tant centralisée que distribuée. Cependant, peu de travaux portent sur l'évaluation de l'efficience du pilotage par le produit dans l'interaction entre des systèmes d'information centralisés de niveau business (ERP) et des systèmes distribués de niveaux process (MES).

Notre contribution porte sur un outil de modélisation et de simulation de systèmes de pilotage contrôlés par le produit afin d'évaluer différentes topologies d'organisation combinant décisions centralisées et/ou distribuées en comparant certains critères de productivité.

Nous présentons d'abord la définition, le développement et la validation d'un environnement d'évaluation orienté composants, basé sur un outil d'émulation et un système multi-agents, permettant d'analyser les performances d'un système de pilotage par le produit et de le comparer avec des approches classiques.

Nous présentons ensuite l'application du pilotage par le produit à partir d'une série d'expériences réalisées à l'aide de l'environnement développé. Ces expériences, menées sur un cas industriel ainsi que sur une plateforme d'expérimentation de laboratoire permettent d'éprouver et de valider la faisabilité du concept de pilotage par le produit en terme d'impact décisionnel et en terme de contraintes techniques.

Mots-clés: système de production intelligents, contrôle par le produit, simulation, évaluation de performance, systèmes holoniques de production, système multi-agents.

Abstract

New developpments of infotronics technologies, that enable the product to carry data in order to have an active role in the cybernetic loop, drive to consider new product-driven architectures for manufacturing execusion control systems. There is indeed a large consensus on this approach that combine product instrumentation and either centralised or distributed control architecture. Nevertheless few works focus on evaluating the pertinence of product-driven control as a way to solve interaction issues between centralised business-level control (ERP) and distributed manufacturing-level control (MES).

This thesis contribution is a benchmarking environment able to model and test various product-driven organisations in order to analyse different ways to combine centralised and distributed decisions by comparing productivity performance indicators.

First, the definition, development and validation of a component-based modeling and testing environment is presented. It is composed of an emulation tool able to build industry-scale shop floor models, and of a multi-agent system to model either traditionnal or product-driven control architecture.

Then, this benchmarking environement is applied to an industrial case-study and on a laboratory experimental plateform. These experiments enable to study the feasability of product-driven control in terms of both decisional impact and technical constraints.

Keywords: intelligent production system, product-driven control, simulation, benchmarking, holonic manufacturing systems, multi-agent systems.