Proposition d'une plateforme d'expérimentation sur le contrôle par le produit des flux de production

Rémi Pannequin, André Thomas

PROPOSITION D'UNE PLATEFORME D'EXPÉRIMENTATION SUR LE CONTRÔLE PAR LE PRODUIT DES FLUX DE PRODUCTION

Rémi PANNEQUIN, André THOMAS

Centre de Recherche en Automatique de Nancy, CNRS UMR 7039
Université Henri Poincaré
faculté des sciences, BP 209, 54500 VANDOEUVRE, FRANCE
{Remi.Pannequin, Andre.Thomas}@cran.uhp-nancy.fr

RÉSUMÉ : *Le développement d'approches novatrices du contrôle de la production se heurte à la difficulté d'évaluer la qualité de ces propositions. Pour valider les approches de contrôle par le produit des flux de production, nous avons besoin d'une infrastructure d'expérimentation. Nous proposons ici une plateforme basée sur l'émulation du système opérant. Ainsi, il nous est possible de représenter des systèmes complexes, d'échelle réaliste. Nous introduirons des primitives de modélisation permettant de représenter les aspects spécifiques au contrôle par le produit. Puis nous spécifierons la manière dont un système de contrôle (éventuellement multi-agents) peut être intégré à notre plateforme. Enfin, nous appliquerons notre démarche au développement du modèle d'émulation d'un site industriel.*

MOTS-CLÉS : contrôle holonique, émulation

1 INTRODUCTION

On assiste depuis quelques années au développement d'approches novatrices relatives au pilotage des systèmes de production. Parallèlement à l'amélioration des système de gestion de production classiques, basés sur MRP2 (Vollman *et al.*, 1997) et mis en œuvre par les systèmes de planification avancés (*Advanced Planing Systems*, APS), de nouveaux paradigmes de contrôle sont explorés. Ceux ci sont souvent basé sur la distribution de "l'intelligence technique" dans des entités décisionnelles autonomes et coopérantes. En particulier, le paradigme holonique (Valckenaers, 2001) vise à associer une entité informationnelle, voire décisionnelle, à chaque élément du système de production (recettes et ordres de fabrication, ressources, produits, etc...).

Parallèlement, les avancées des technologies d'identification par radio-fréquences (*Radio Frequencies Identification* - RFID) permettent, en associant à chaque produit une étiquette électronique (*tag*) de le doter de capacités de communication, de traitement et de stockage de l'information. Cette mise en réseau des produits ("*Internet of Things*") (McFarlane et al., 2003), permet de mieux intégrer les produits aux systèmes d'information de l'entreprise, et rendent possible des approches alternatives de contrôle de la production.

Ainsi, on peut envisager de faire du produit l'acteur de sa production. C'est ce paradigme de contrôle que l'équipe « systèmes contrôlés par le produit » du centre de recherche en automatique de Nancy (CRAN) cherche à explorer (Morel *et al.*, 2003). Les bénéfices attendus concernent la flexibilité et la robustesse du contrôle, mais aussi la possibilité de produire en masse des produits personnalisés (*mass-customisation*), ou encore d'améliorer la traçabilité tout au long de la chaîne logistique.

Ces propositions novatrices se concrétisent souvent sous la forme d'*architecture de référence*. Une architecture de référence constitue une abstraction d'une architecture générique, définissant une terminologie unifiée, le type des composants du système, leurs responsabilités, leurs dépendances, leurs interfaces. Les architectures PROSA (Van Brussel *et al.* 1998) ou HCBA (Chirn *et al.* 2000) en sont deux exemples.

Mais comment évaluer la qualité d'une telle architecture de référence ? D'une part, il s'agit de vérifier le bon fonctionnement des mécanismes de pilotage ; d'autre part de comparer la proposition aux solutions « classiques ».

La démonstration d'une unique application pratique n'est pas véritablement probante, car trop partielle. De plus, pour des raisons techniques et économiques, cette mise en œuvre est parfois réalisée sur des systèmes réels de petite taille, ou à l'aide de simulateurs. L'utilisation d'un système trop simple par rapport à l'échelle du problème censé être résolu conduit à des résultats discutables. En

effet, l'un des problèmes auquel on fait face est la complexité même du système à contrôler. De même, l'utilisation de la simulation (qui permet d'utiliser des modèles de taille réaliste), n'est pas sans poser des problèmes. En effet, le modèle de simulation ne représente que certains aspects de la réalité. De plus, on ne peut pas assurer la fiabilité de cette représentation partielle. Une validation utilisant ce modèle partiel et potentiellement erroné est donc sujette à caution.

Cependant, des travaux ont été effectués sur la comparaison de concepts organisationnels antagonistes. Ainsi, (Cavalieri *et al.* 2000) a développé un système multi-agents effectuant l'ordonnancement et le pilotage de la production d'un atelier virtuel simple. Cet atelier comporte quatre machines, des équipements de transport, un stock tampon de capacité infinie. Ce système a servi à comparer les performances d'une organisation de type « marché », avec une coordination hiérarchique. De la même manière, (Brennan, 2000) utilise une émulation d'un atelier de production assez simple lui aussi, pour comparer une architecture de contrôle réactive à une architecture fondée sur une planification.

On voit donc petit à petit apparaître des cas de test. Mais ceux-ci sont difficilement réutilisables. En effet, ils sont spécifiques à un atelier virtuel donné, dont l'échelle est plutôt celle du laboratoire que de l'industrie. De même ils sont spécifiques (d'un point de vue technique) à une architecture de contrôle. Pour obtenir un outil de test plus exploitable, le groupe d'intérêt spécifique 4 (SIG4) du réseau d'excellence IMS-NoE (IMS, 2005) a vu le jour. Il est consacré au développement d'un service de benchmarking mettant à la disposition de la communauté un ensemble de cas industriels virtuels (Cavalieri *et al.*, 2003).

Dans le même esprit, nous proposons ici une infrastructure expérimentale pour l'étude des systèmes contrôlés par le produit. Cet ensemble d'outils permet, d'une part, la conduite effective des expériences, par l'émulation d'un système opérant, et fournit, d'autre part, des outils de modélisation permettant de guider le développement de ces modèles d'émulation. L'objectif final est de constituer une bibliothèque de cas de tests réutilisables.

Après avoir précisé les notions d'émulation de simulation, nous examinerons les fonctionnalités devant être remplies par notre plateforme, d'abord d'un point de vue général, puis relatif au contrôle par le produit. Ensuite, nous présenterons les éléments de sa mise en œuvre, basée sur une émulation du système industriel à contrôler. Enfin, à partir d'une application, nous discuterons des choix technologiques, et des performances du système.

2 CARACTÉRISATION D'UN SYSTÈME D'EXPÉRIMENTATION

2.1 Qualité d'une architecture de référence

L'étude de la qualité d'une architecture de référence comporte deux aspects :

- Les propriétés *structurelles* de l'architecture, comme par exemple la nature des informations échangées entre ses éléments, sa facilité de mise en œuvre, sa ré-utilisabilité, ou sa capacité à évoluer ;
- Les performances *opérationnelles* de cette architecture, par exemple la capacité à maintenir un niveau de production élevé malgré des aléas, ou à assurer une production conforme à la demande, dans un contexte de haute variété des produits.

Les aspects structurels peuvent être étudiés sur l'architecture en tant que telle, hors de toute application, à l'aide de métriques (Hollocks *et al.*, 1997), (Brennan & Norrie, 2003). Par contre, les propriétés opérationnelles ne peuvent pas être mesurées "abstraitement", mais nécessitent une mise en œuvre concrète. Ainsi, les qualités opérationnelles générales d'une architecture de référence ne peuvent être obtenues que par induction, à partir de plusieurs cas d'application (figure 1).

On voit donc la nécessité de constituer une bibliothèque de cas d'application virtuels.

L'évaluation des performances d'un cas particulier, dérivé de l'architecture de référence, peut être illustrée par le prédicat de l'automatisation de Fusaoka (Fusaoka *et al.*, 1983) :

$$\textit{Système de contrôle} \wedge \textit{Système opérant} \supset \textit{Spécifications}$$

Pour résoudre cette équation, on peut procéder par vérification, d'une part, en prouvant formellement que quelque soit la situation, une certaine propriété sera vraie. Cette approche requiert un modèle formel des sous-systèmes opérant et de contrôle. Dans le cas du contrôle par le produit, dans lequel on se place, cette modélisation est difficile, et la complexité du problème peut dépasser les capacités des logiciels de preuve. En effet, le comportement global *émerge* des interactions entre des entités régies par des comportements élémentaires. On ne peut représenter un tel système d'une manière simple (Wegner, 1997).

On peut, d'autre part, procéder par validation, en observant le comportement du système (système opérant et système de contrôle) dans une situation donnée. À cause des aspects stochastiques, de nombreuses expériences sont nécessaires pour être probantes. Cette approche « de Monte-Carlo » a pour avantage sa simplicité de mise en œuvre, car elle n'exige pas de formaliser le système de contrôle. Ainsi, il est possible d'étudier des systèmes complexes, comprenant par exemple une grande variété de produits, comprenant divers flux de matière, et différents types de perturbations.

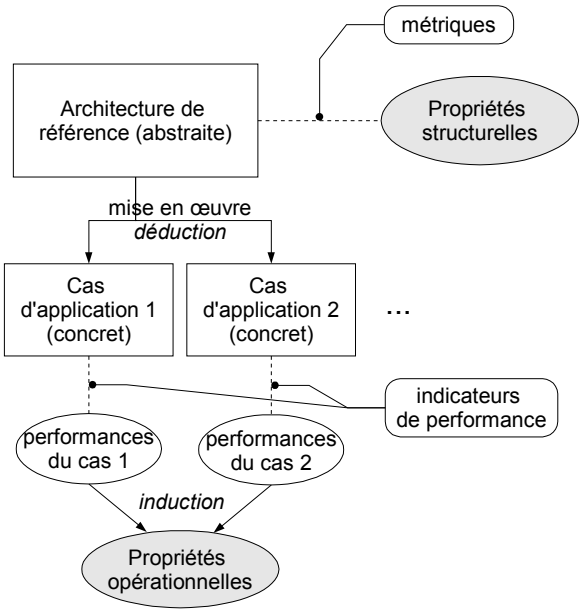

Figure 1 : Principe d'obtention des propriétés structurelles et opérationnelles

Pour effectuer ces expériences facilement, tout en respectant notre objectif de constituer une bibliothèque informatique de cas industriels de test, nous avons décidé d'*émuler* le système opérant.

2.2 Émulation et simulation

Il convient avant d'aller plus loin d'expliciter ces deux termes. 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 (Corbier, 1989). En électronique, l'émulation d'un microprocesseur permettra d'exécuter des programmes conçus pour ce composant, sans pour autant en disposer physiquement.

S'il s'agit dans tous les cas de « singer » un comportement, la distinction fondamentale repose en fait sur le but recherché : dans une démarche de simulation, on cherche à observer l'évolution des états internes du modèle placé dans une situation *prédéfinie*. Dans une démarche d'émulation, on cherche à reproduire l'interaction du système 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.

Dans le domaine de l'étude des systèmes de production, la simulation peut par exemple servir au dimensionnement d'une nouvelle installation, ou à l'évaluation de la faisabilité d'un plan de production. Cette simulation se fait grâce à un modèle dynamique du système industriel. Pour que la simulation soit fidèle, on doit lui adjoindre des éléments du système de pilotage (par exemple des règles de lotissement ou de changement de série) (Van der Zee, 2005). Néanmoins, cette intégration est délicate, et les règles de décision sont souvent simplifiées.

A défaut de *simuler* le système *complet* (système opérant et système de contrôle), on peut *émuler* le système opérant, en le connectant à un système de contrôle externe. Dans cette configuration, on pourra facilement passer d'un système de contrôle à un autre. Faire l'émulation du système opérant nous permet donc d'atteindre notre objectif de modularité.

Puisqu'il doit échanger avec son environnement, l'émulateur devra nécessairement être doté de moyens de communication avec celui-ci. Un logiciel « de simulation », qui permet d'animer un modèle dynamique, que l'on dotera de capacités de communication pourra donc servir de support à un émulateur (Brennan, 2000). Les outils qui permettent de construire et d'animer les modèles de simulation et d'émulation peuvent donc être les mêmes, ce qui contribue à une certaine confusion.

En conclusion, l'émulation se caractérise par une interaction permanente avec son environnement. C'est la restitution fidèle de ces échanges qui est le but de la démarche d'émulation, l'émulateur devenant un composant d'un système plus large. A contrario, une simulation reste isolée pendant l'expérience, à l'exception de la phase de paramétrage, et de la lecture des résultats.

2.3 Modularité des modèles d'émulation

De manière générale, une plateforme d'expérimentation comporte deux sous-ensembles :
- l'environnement de modélisation, qui permet de construire un modèle du système opérant, accompagné de ses paramètres ;
- l'environnement d'expérimentation, dans lequel doivent s'interfacer l'émulation du système opérant, et le système de contrôle.

Un schéma de principe de la plateforme est présenté figure 2.

Chaque cas de test doit pourvoir interagir avec un système de contrôle quelconque, afin de faire facilement des comparaisons. Pour atteindre cet objectif de modularité, on peut choisir d'utiliser une interface universelle, à laquelle les modèles devront se conformer.

L'environnement de modélisation devra donc permettre de construire facilement de tels modèles.

Cette interface universelle se fonde sur la distinction entre système opérant et système de contrôle. De plus, l'universalité de l'interface repose sur une délimitation qui doit être la même pour chaque modèle. Ce « niveau de coupure » entre système contrôlé et système de contrôle repose d'abord sur le niveau d'agrégation utilisé pour représenter les équipements de l'atelier. Ces « atomes » de représentation peuvent être un atelier, une cellule, une machine ou un actionneur... Ce choix conditionne ensuite le niveau d'agrégation des instructions de contrôle reçues, et des comptes-rendus émis.

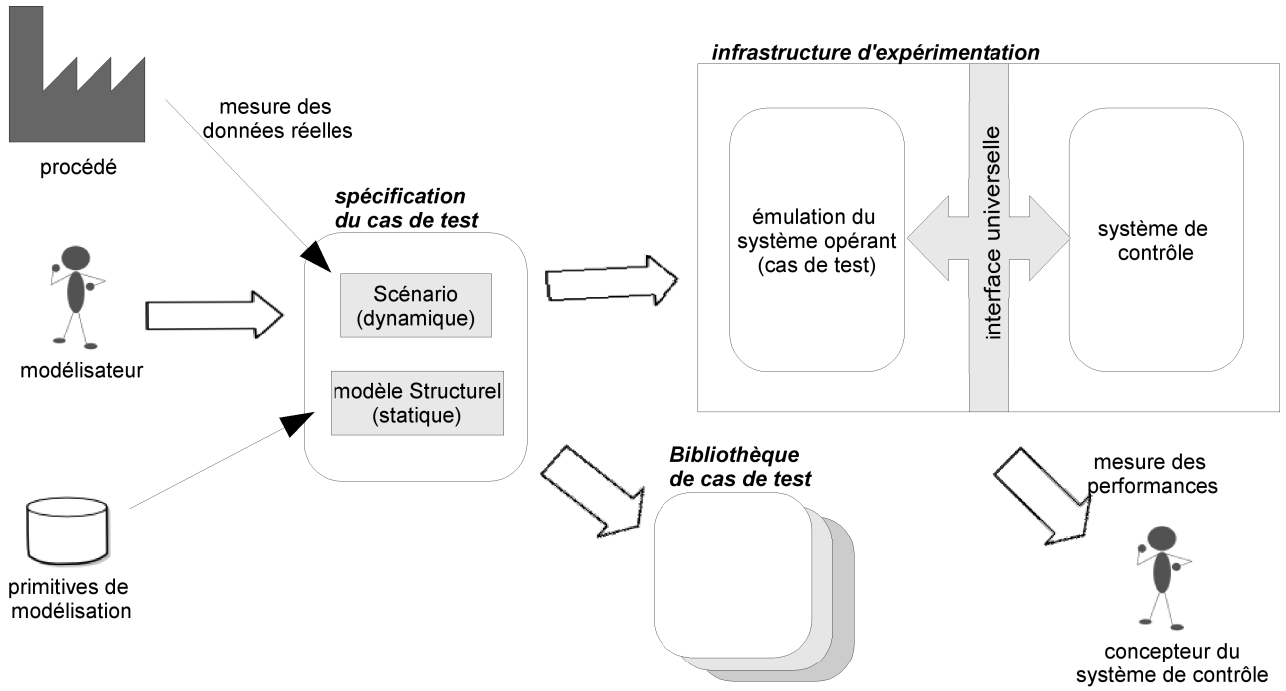

Figure 2 : principe de la plateforme d'expérimentation

Ces « briques » élémentaires constituent les primitives utilisées pour constituer les modèles d'émulation. La définition de chaque primitive de modélisation, ainsi que de son interface permet de définir l'interface du modèle, comme l'union de chaque interface élémentaire. Avant de définir notre choix du niveau de coupure (section 3.2), examinons les travaux présentés dans la littérature.

Les travaux du SIG4 de l'IMS-NoE ont conduit à définir des primitives de modélisation inspirées de composants physiques (Cavalieri *et al.*, 2003) :
- Station de travail, et plus précisément d'assemblage, de chargement/déchargement, d'usinage,
- zones de stockage, comme des tampons d'entrée ou de sortie,
- équipements de transport, tels des convoyeurs ou des véhicules autonomes.

L'interaction se fait alors entre un élément physique et son homologue du système de contrôle à travers une communication inter-processus, utilisant les technologies JAVA.

L'outil de simulation MAST (Marik & Vrba, 2005), développé par Rockwell Automation distingue lui aussi trois types d'éléments devant être représentés :
- cellules de production, jouant le rôle d'une machine d'assemblage, ou d'usinage, mais ayant aussi une importance dans le routage des pièces.
- 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 (CDCA) (Fletcher et al., 2003). Ainsi, les lecteurs RFID ont été introduits, et d'autres éléments génériques ont été modifiés pour correspondre aux spécificités du cas.

On constate donc que le niveau de granularité habituellement choisi est celui de la machine à commande numérique. Le choix des éléments à émuler provient des objets techniques existants effectivement dans l'atelier (convoyeur. robots, machine C/N). Reste à savoir si ce choix comporte une abstraction de modélisation suffisante pour représenter de manière homogène différents cas, et si il est compatible avec l'étude du contrôle par le produit.

2.4 « Honnêteté » des expériences

On peut enfin se poser la question de la validité des expériences menées sur le système émulé. En effet, le résultat d'une expérience est déterminé par le système de contrôle, mais aussi par la manière de modéliser le système opérant.

Lors de la modélisation du système opérant, deux phénomènes interviennent :
- le choix du langage et des outils de modélisation, ainsi que la subjectivité de la personne effectuant la modélisation, détermine une abstraction particulière de la réalité. Ainsi, certains aspects de la réalité n'apparaîtront pas dans le modèle ;
- la réalité à modéliser pourra être traduite dans le modèle de manière erronée.

La modélisation n'est donc pas neutre, et seules certaines dimensions de la réalité sont représentées dans le modèle, à la manière d'une projection (figure 3). On cherchera donc autant que possible à minimiser l'impact de cette projection sur les expériences menées.

Pour atteindre ce but, le modèle d'émulation devra rendre potentiellement possible *l'échec* d'une expérience. Cette affirmation se fonde sur les travaux en épistémologie de Popper. D'après lui, une proposition scientifique est une proposition réfutable et non encore réfutée. On voit donc l'importance de la réfutation dans l'analyse critique d'une architecture de référence. Une expérience montrant la « réussite » d'un système de contrôle ne sera valable que si cette expérience aurait pu échouer.

Il est donc important dans le modèle d'émulation de ne pas escamoter les aspects pouvant conduire à l'échec d'une expérience de contrôle.

Pour cela, on peut par exemple dissocier la conception du système de pilotage de la modélisation du système

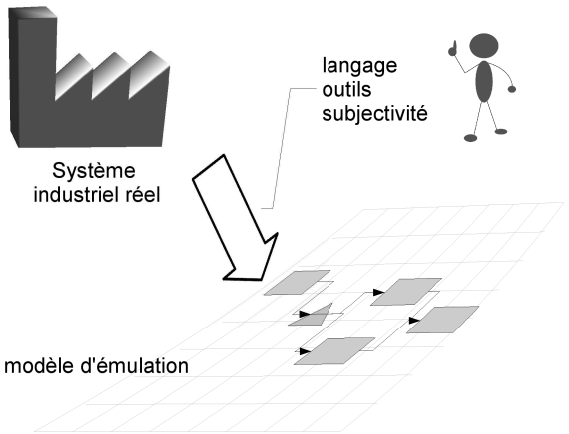

Figure 3 : projection du système réel dans un modèle. Seuls certains aspects sont retenus.

opérant. En effet, si une seule et même personne effectue la spécification du problème industriel devant être résolu (le modèle d'émulation) et de sa solution (le système de contrôle), on voit peu de raisons, autres que des problèmes techniques, pour que l'expérience échoue. Ceci peut être nuancé lorsque le modèle est vu par d'autres personnes.

La propriété principale que doit avoir l'émulateur sera donc de conserver les aspects de la réalité pouvant poser des problèmes de contrôle de la production. En particulier, les problèmes industriels sur lesquels le contrôle par le produit peut avoir un effet se devront d'être scrupuleusement transcrits dans le modèle d'émulation.

3 PROPOSITION D'UNE PLATEFORME EXPÉRIMENTATION

3.1 Fonctionnalités requises

Nous allons spécifier les fonctions devant être remplies par l'émulateur, pour qu'il soit utile à la validation du système de contrôle par le produit. C'est à dire quels aspects du système réel doivent être conservés. La figure 4 présente les fonctions spécifiques au contrôle par le produit devant être présentes dans notre outil, en plus des fonctions traditionnelles des émulateurs.

D'une part, le contrôle par le produit nous amène à séparer les flux de natures différentes. En effet, dans le cadre d'une production en gros volumes et grande variété, la synchronisation des flux physiques et des flux d'informations pose problème.

Par exemple, dans l'automobile, le flux de véhicules à peindre doit être synchronisé avec le flux informationnel de teintes à donner, afin que chaque véhicule ait une teinte conforme aux demandes du client. La synchronisation des flux est encore plus importante dans le contexte du juste-à-temps et de l'approvisionnement synchrone (j*ust in sequence*). L'un des bénéfices attendus du contrôle par le produit est une « synchronisation naturelle » des objets physiques (les produits) avec leurs images informationnelles.

Pour évaluer l'impact de cette synchronisation, ces flux de différentes natures doivent être séparables. Cependant, il est habituel lorsque l'on construit des modèles de simulation de ne pas séparer ces flux. En effet, les outils de simulation à événements discrets reprennent naturellement une modélisation inspirée de la théorie des files d'attente. Celle ci met en avant les notions de serveur et de queue. Dans cette modélisation, une entité arrivant à un serveur représente un produit physique, mais aussi l'événement déclenchant le traitement, et elle peut porter des informations relatives à la manière de la traiter.

Cette confusion *a priori* de flux de natures différentes ne permet pas de tester l'action du flux d'information sur le flux physique.

Une classification basée sur la nature des flux nous permet de distinguer trois types :
- les flux de matière, comme par exemple les produits entrant et sortant d'un processus ;
- les flux d'événements, tels des ordres à effectuer (début cycle), ou des compte-rendus émis (fin de cycle) ;
- les flux d'information qui paramètrent la tâche à effectuer, ou représentent l'état du processus.

Nous devrons donc développer l'émulateur de manière à obtenir une séparation *a priori* de ces flux.

Parallèlement au problème de la synchronisation des différents types de flux, se pose le problème du routage des produits dans l'atelier. Par exemple, il s'agit du choix entre deux machines parallèles, ou bien du choix d'un emplacement de stockage. Comme précédemment, ces aspects sont souvent ignorés dans les modèles de simulation. En effet, le soucis du concepteur du modèle sera de faire suivre aux produits simulés la trajectoire souhaitée, qui est habituellement prédéfinie, puis importée depuis un outil extérieur. Comment dans ces conditions évaluer un choix par le produit lui-même de sa trajectoire ? Il faudra donc modéliser tous les routages possibles des produits dans l'atelier.

D'autre part le contrôle par le produit conditionne le « niveau de coupure », entre émulation et système de contrôle (voir section 2.3).

En effet, il nous faut étudier le dialogue entre le produit et son environnement. Il est donc naturel de modéliser le système opérant tel qu'il est « vu par le produit ». D'abord, le niveau d'agrégation des échanges portera naturellement sur une instance de produit, et non sur un lot. Le produit va donc voir une série d'opérations qu'il subit. Le niveau de détail que nous considérons dans le

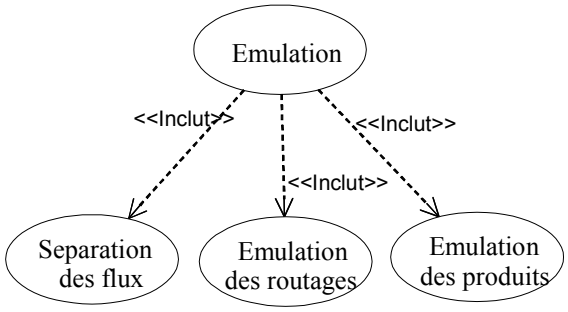

Figure 4 : Cas d'utilisations requis dans le contexte du contrôle par le produit

découpage de ces opérations est celui de la cellule de production. En effet, aucune décision de routage n'est possible à l'intérieur d'une cellule. Elle constitue de ce fait un élément indivisible dans les décisions prise par le produit.

3.2 Primitives de modélisation du système opérant

Comme évoqué en section 2.3, un élément important de la plate forme est constitué par une bibliothèque de primitives de modélisation, qui donnent un cadre au développement du modèle d'émulation.

Plutôt que de nous appuyer sur les objets technologiques de l'atelier, nous nous sommes inspirés d'une classification systémique, pour déterminer nos primitives. Ainsi celles-ci se déclinent en transformations de temps, d'espace, et de forme (Le Moigne, 1977). Dans l'atelier modélisé, chaque produit subit une succession de transformations de forme, et d'espace, *dans* le temps. En effet, les changements de forme (par exemple un perçage) ou d'espace (un passage sur un convoyeur) durent un certain temps. Tenir compte de ces durées nous permet de représenter la dynamique du système.

De plus, ces transformations de forme-temps ou d'espace-temps sont directement connectées au système contrôlant.

D'une part, elles reçoivent des informations qui conditionnent le traitement qu'elles appliquent aux produits. Dans le cas d'une transformation de forme, on considère que les traitements pouvant être appliqués au produit sont en nombres finis. En effet, dans le cas d'une machine à commande numérique, on peut associer un identifiant à chaque programme enregistré. Chaque traitement peut donc être désigné par son identifiant. Par ailleurs, les transformations d'espace sont paramétrées par deux variables identifiant les emplacements de départ et d'arrivée. Elles émettent en retour la valeur effective de ces variables au système de contrôle. On constate donc que l'on a effectué une discrétisation de l'évolution du produit dans sa forme, ainsi que dans l'espace.

D'autre part, elles reçoivent des événements. Nous considérons deux événements, l'un pour déclencher un cycle de fabrication, l'autre pour un changement de série. Parallèlement les transformations émettent des événements correspondants à leurs changements d'états. Ceux ci sont « en marche », « indisponible », « libre ».

Enfin, chaque transformation est paramétrée par des tableaux de réels, représentant respectivement les durées associées à chaque programme, et aux changements de série.

La séquence des traitements appliqués aux produits est par ailleurs conservée. Elle permet de connaître la route suivie par un produit, et surtout, de vérifier que le produit a effectivement reçu les traitements qu'il devait recevoir (i.e. qu'il soit conforme aux spécifications).

Enfin, comme les produits eux-mêmes peuvent interagir avec le système d'information, il nous faut une autre catégorie de primitives, relatives au cycle de vie du produit, pour représenter :
- sa création et sa destruction,
- l'assemblage ou le déassemblage de deux produits,

les événements de sa « vie », déclenchés par le passage dans un lieu donné.

Ces primitives sont elles aussi connectées au système de contrôle, dans la mesure où elles fournissent des événements.

3.3 Modélisation du système de décision

Pour modéliser le système de contrôle, nous avons choisi d'utiliser une technologie multi-agents. Celle-ci sert d'une part à modéliser les produits, et d'autre part, à faire effectivement fonctionner le système de contrôle.

Nous définissons donc deux types d'agents :
- les agents produits, qui représentent les capacités de stockage et de décision associées aux produits
- les agents de contrôle, qui constituent le système de contrôle à proprement parler.

L'utilisation d'agents pour représenter les produits est naturelle, puisque chaque produit intelligent s'exécute de manière autonome, et dispose de ses propres informations. D'autre part, seul un système multi-agents peut simuler un autre système multi-agents (Marik & Vrba, 2005).

De plus, l'utilisation d'agents nécessite la définition explicite des interactions en terme de protocole, et de structure de message entre les agents. Dans l'outil JADE,

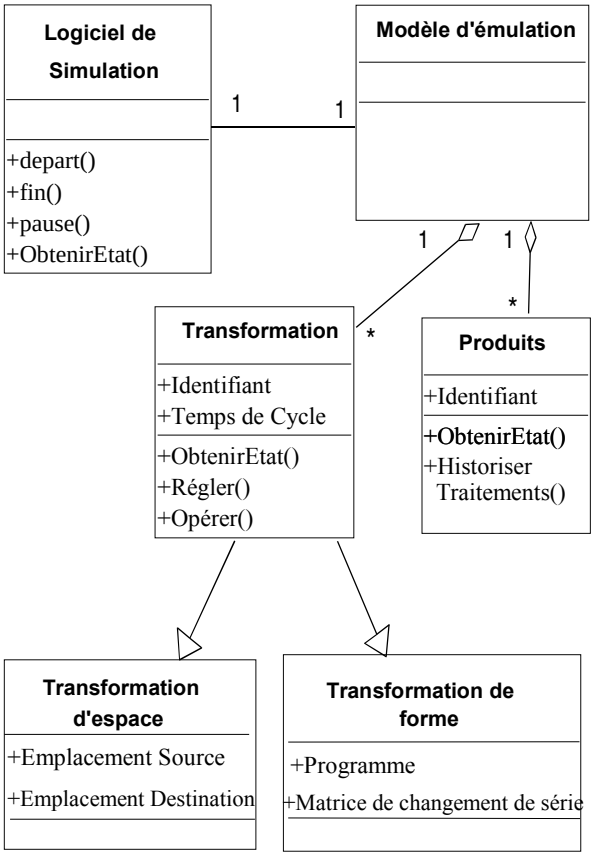

Figure 5 : Diagramme de classe de l'émulateur

les interactions sont définies grâce à la théorie des actes de langage (performatives) (Searle, 69). L'outil conditionne donc une définition très claire, et une étude facilitée des échanges du produit avec son environnement.

Quant au système de contrôle, une plateforme multi-agents permet de représenter tous les types d'organisation, quelles soient hétérarchiques ou hiérarchiques, centralisées ou décentralisées. Il suffit pour cela de changer les modalités d'interaction entre agents, en « rigidifiant » les relations pour hiérarchiser le système.

En fait, ces agents constituent le système de contrôle réel, et pourraient être utilisés par la suite en production.

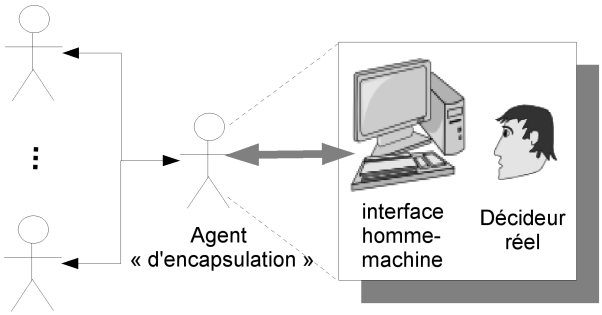

Figure 6 : un agent du système décisionnel peut encapsuler un acteur externe, par exemple un opérateur pouvant prendre des décisions

Cependant, il est parfois nécessaire de tenir compte du comportement d'acteurs externes, comme par exemple un logiciel d'ordonnancement, ou un opérateur auquel est conféré une certaine capacité de décision (figure 6). Dans ce cas, on pourra « encapsuler » cet acteur dans un agent, en développant une interface logicielle s'il s'agit d'une application, ou bien en offrant une interface homme-machine si c'est un humain. Il est aussi possible de modéliser les grandes lignes du comportement de l'acteur (le plus souvent humain) dont il est question, et d'utiliser ce modèle plutôt que l'acteur réel dans l'agent.

3.4 Interactions entre éléments : structures des échanges

Les échanges entre l'émulateur et le système de contrôle nécessitent la mise au point d'une infrastructure technique de communication, mais aussi la définition de la syntaxe et de la sémantique des informations échangées.

Les problèmes de communication peuvent être résolus par l'utilisation d'une couche d'interconnexion (*middleware*). Ce type de logiciel permet d'exécuter des méthodes d'un programme distant, ou assure le partage d'*objets* entre applications géographiquement distribuées.

Grâce à ces outils, l'interaction entre émulateur et contrôleur peut prendre la forme d'un système de « tableau noir » (*blackboard*). Il s'agit d'un objet partagé, dont la valeur des attributs est accessible par les deux acteurs. L'échange se fait alors en modifiant les variables du tableau.

Un autre mode d'interaction est l'utilisation de messages. Celui-ci a pour avantage une meilleure visibilité des échanges, qui peuvent être facilement tracés, et enregistrés dans un historique. De plus ce mode assure une plus grande ouverture de l'émulateur, puisque tout logiciel capable d'interpréter ce message pourra être utilisé. En contre-partie, il demande une définition explicite et partagée de la structure des messages.

Nous avons donc choisi une communication basée sur l'échange de messages. Ceux-ci sont écrit dans le langage XML (*eXtended Markup Langage* – langage à balises extensible), ce qui permet de valider simplement leur syntaxe et de faciliter leur codage et décodage.

La syntaxe est définie dans une DTD (*Document Type Definition* – définition de type de document), ce qui assure l'ouverture du programme. Nous avons défini deux types principaux de messages :

- « événement » qui met en relation éléments émulés, et éléments de contrôle. La structure de ces messages est donnée figure 7.
- « gestion de la simulation », qui permet de piloter l'outil d'émulation (lancement, arrêt, pause)

De plus, un type « découverte » devrait être ajouté, pour permettre au système de contrôle de connaître la structure et les paramètres des éléments émulés.

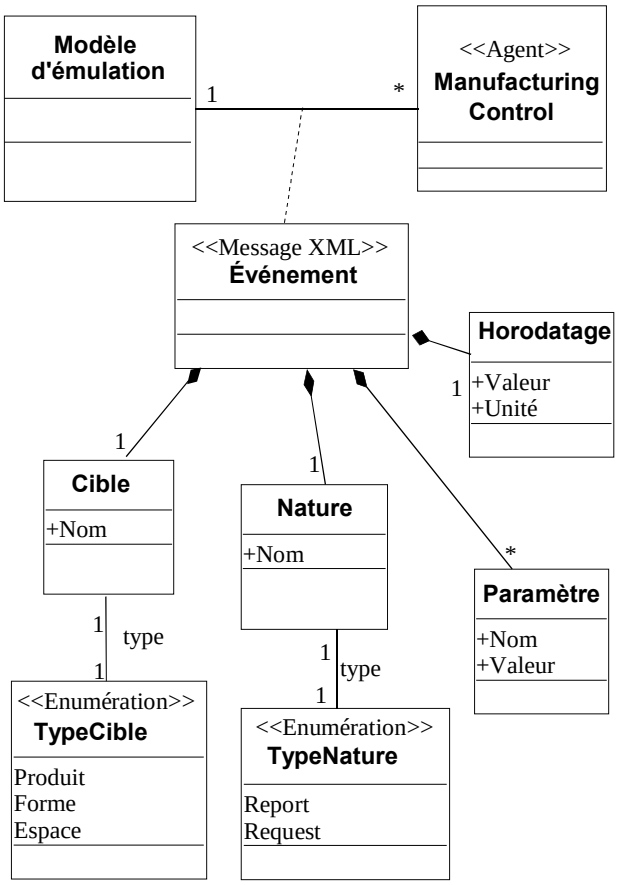

Figure 7 : Structure des messages échangé entre l'émulateur et le système contrôle

Chaque « événement » est constitué de plusieurs éléments :
- la cible, qui est l'objet émulé émetteur (ou récepteur) de l'événement. Elle est désignée par son nom, et son type est mentionné.
- la nature de l'événement. Chaque type d'événement est identifié par son nom. Ceux-ci ont été définis en section 3.2 pour le cas d'une transformation. Les noms utilisés pour identifier les événements relatifs aux produits sont définis dans le modèle d'émulation.
- un ensemble de paramètres qui servent à transmettre des informations aux éléments émulés (par exemple l'identifiant du programme a exécuter)
- un horodatage de l'événement.

4 APPLICATION ET DISCUSSION

4.1 Différentes approches de développement des modèles

Le développement effectif d'un modèle d'émulation peut être fait de deux manières :
- on peut développer à partir de rien un modèle dédié à l'émulation,
- on peut modifier un modèle de simulation qui existe déjà, développé pour un autre but.

Le développement « complet » du modèle d'émulation est une solution qui permet d'obtenir un modèle bien adapté à l'émulation, mais cette démarche de modélisation reste un travail de grande envergure. De plus, le développement de ce modèle dédié est souvent fait par son utilisateur direct, c'est à dire celui qui met au point et désire tester le système de contrôle. Il y a alors le risque de biaiser les expériences, comme indiqué en section 2.4.

Pour résoudre ce problème, et pour alléger la charge de développement, de paramétrage et de validation du modèle d'émulation, il est judicieux de ré-utiliser, lorsqu'il existe, un modèle de simulation du système.

Dans ce modèle global, on pourra ensuite délimiter une zone dans laquelle les composants classiques pourront être remplacés par les composants « émulables ». Cela permet de donner un contexte réaliste à cette zone de test. En effet, un paramètre important dans l'étude d'une décision dé-centralisée (et donc locale) est son impact sur le système de production global.

4.2 Cas d'un fabriquant de meubles

Nous nous intéressons au cas d'un fabricant de meubles en kit (à monter soi même). La production des meubles se découpe en trois étapes principales :
- la découpe des panneaux de particules,
- le perçage, et d'éventuelles opérations complémentaires (pose de miroirs, usinage d'un arrondi, etc...)
- l'emballage des différentes pièces d'un meuble, y compris la quincaillerie et la documentation.

la figure 8 schématise les flux dans cet atelier.

Entre deux étapes d'usinage, les pièces sont empilées sur un porteur, et transportées sur des voies. Les piles de pièces ne peuvent pas se doubler dans une voie. Chaque voie constitue donc une queue FIFO (premier entré, premier sorti), ou éventuellement LIFO (dernier entré, premier sorti).

Une ou plusieurs voies sont placées en entrée et en sortie des cellules d'usinage. Les stocks d'en-cours et de pièces finies sont constitués de plusieurs voies parallèles. Des transbordeurs, qui se déplacent perpendiculairement aux voies, permettent de déplacer une pile de pièces d'une voie à une autre. Les stocks de matière premières (grands panneaux de particules, principalement) et de produits finis (colis) ont une topologie plus classique.

Au cours d'une semaine, plusieurs milliers de porteurs chargés de pièces circulent dans l'atelier.

4.3 Mise en oeuvre

Nous disposions donc d'un modèle effectuant la simulation d'un plan de travail ordonnancé. Ce modèle avait été réalisé à l'aide du logiciel de simulation à événements discrets Arena, de Rockwell Software.

Les composants à émuler ont été implantés dans une bibliothèque d'Arena. Pour faciliter la modélisation,

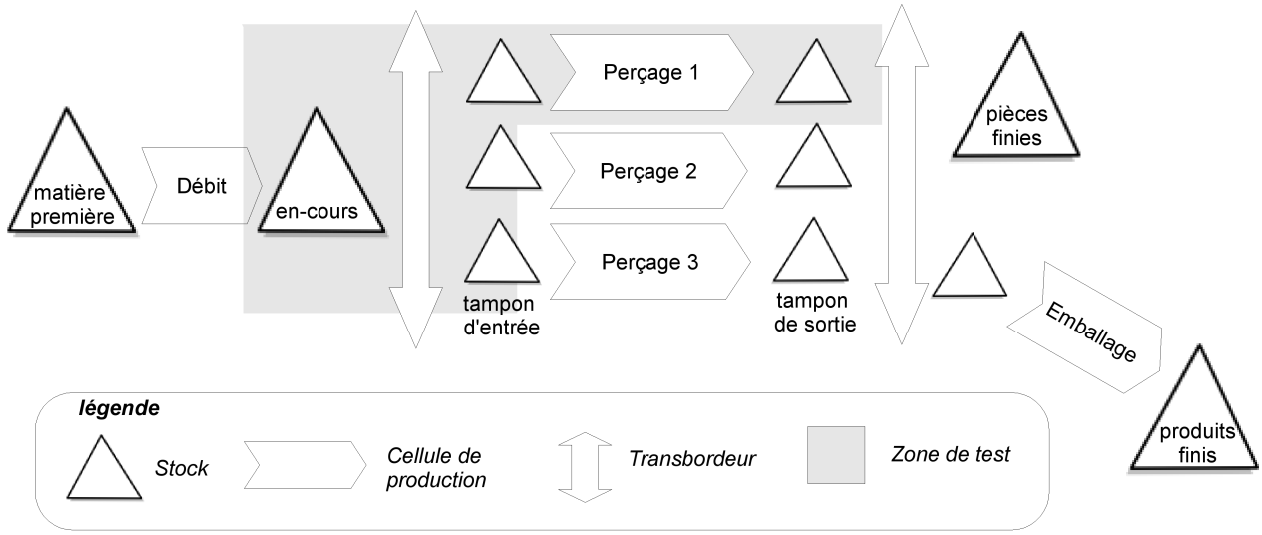

Figure 8 : modèle global du cas, et périmètre de la zone de test.

d'autres macro-composants spécifiques à ce cas (les voies) ont été développés.

Les éléments servant à simuler la zone de test ont donc été supprimés du modèle, et remplacés par les blocs d'émulation. La cellule de perçage a été modélisée par une transformation de forme, et le transbordeur par une transformation d'espace. Les problèmes qui se sont posés dans cet étape sont le manque de données relatives au terrain. En effet, les temps utilisés pour paramétrer les machines dans le modèle de simulation étaient directement déduits du plan ordonnancé. Le modèle d'émulation nécessite des données moins agrégées (temps méthodes, ou réels, relatifs à chaque programme).

4.4 Discussion

Au final, la création du modèle d'émulation se révèle assez facile. En effet, il n'est plus nécessaire de chercher à intégrer à la simulation les décisions prises sur le terrain. Il en résulte un modèle souvent moins abstrait, et plus proche de la réalité. Toutefois, une connaissance précise des paramètres de fonctionnement des équipements reste nécessaire.

Néanmoins, la conception du système de contrôle en est d'autant alourdie. En particulier, l'approche multi-agent n'est pas « gratuite », et requiert des compétences spécifiques, et un haut niveau d'expertise.

De la même manière, les primitives forme-temps et espace-temps, si elles sont générales, restent abstraites, ce qui rend leur usage difficile pour un non-spécialiste.

Quand à la validation du modèle, elle repose essentiellement sur la reproduction de la structure réelle de l'atelier. Le modèle étant plus concret, cette reproduction est facilitée. Par ailleurs, le modèle de départ provient de l'industriel, qui l'a validé. De plus, notre but n'est pas de prédire la situation future de l'atelier, mais d'obtenir un cas de test réaliste. L'exactitude des paramètres du modèle de l'atelier requiert donc moins d'importance qu'en simulation.

Ensuite, le choix de Arena comme logiciel de simulation est discutable. Le moteur de simulation à événements discrets basé sur le langage SIMAN est éprouvé par de nombreux usages industriels, et l'interface de modélisation est de type graphique, ce qui facilite la communication en milieu industriel. En revanche, l'utilisation d'Arena apporte également quelques problèmes, avec notamment la difficulté et la lourdeur de la communication avec d'autres applications, ainsi que des incompatibilités entre différentes versions et un coût de licence élevé.

Enfin, un problème difficile que nous avons rencontré concerne la gestion de l'avancement temporel de la simulation. Le mode d'avancement « à événements discrets » n'est pas compatible avec le fonctionnement « temps réel » du système de pilotage. Ce problème avait déjà été identifié dans (Cavalieri et al., 2003). Une solution consiste à utiliser Arena en mode temps-réel. Cependant, cela augmente la durées de simulation.

5 CONCLUSION

Disposer d'un outil d'expérimentation va nous permettre de valider des approches de contrôle de la production par le produit. Cet outil est indispensable, dans la mesure ou il va servir de support à la concrétisation de nos idées, tout en nous permettant d'obtenir des mesures quantitatives de leurs performances. En permettant l'expérience, et donc la réfutation, il est indissociable de la démarche scientifique.

L'apport de notre travail repose d'abord sur l'utilisation des transformations systémiques de temps, d'espace et de forme pour développer les modèles d'émulation. Cela donne à nos primitives de modélisation un haut niveau de généricité, et leur confère une bonne expressivité (par exemple du routage des produits). Ensuite, la spécification de l'interface entre émulateur et système de

contrôle, permet de rendre notre plateforme ouverte et modulaire. Il est par exemple possible d'adapter un outil de simulation autre que Arena, qui pourra assurer le même service. Enfin, nous avons intégré dans le modèle du système opérant et dans le système de contrôle les aspects permettant de prendre en compte le contrôle par le produit. Celui-ci est représenté sous la forme d'une partie physique et d'un partie informationnelle, ce qui permet d'étudier la synchronisation entre les flux de matière de l'atelier et les flux d'informations des systèmes d'information d'entreprise.

En perspectives, on peut d'abord évoquer l'amélioration des aspects techniques de notre plateforme. En particulier, il nous faudra mettre en place un système pour gérer correctement l'avancement temporel de l'émulation. L'utilisation de HLA (High Level Architecture, norme ISO 1516) pour fédérer émulateur et système de contrôle est une voie à explorer. On pourra aussi améliorer la communication entre émulateur et contrôle, en mettant au point un web-service d'émulation, ou en « encapsulant » le logiciel supportant l'émulation dans un agent.

Enfin, nous prévoyons de continuer le travail de modélisation, afin d'enrichir notre bibliothèque de cas de tests. Il sera aussi nécessaire d'intégrer à la plateforme un système d'indicateurs de performance. Mais il nous faudra surtout mettre en œuvre dans le système de contrôle les modèles de contrôle par le produit que nous envisageons.

6 RÉFÉRENCES

Brennan, R. W., 2000. Performance comparison and analysis of reactive and planning-based control architectures for manufacturing. *Robotics and Computer Integrated Manufacturing*, 16, pp 191–200.

Brennan, R. W., D. H. Norrie, 2003. Metrics for evaluating distributed manufacturing control systems. *Computer in Industry*, **51**, pp 225-235.

Cavalieri S., M. Macchi, P. Valckenaers, 2003. Benchmarking the performance of manufacturing control system: design principles for a web based simulated testbed. *Journal of Intelligent Manufacturing*, **14**, 43-58.

Cavalieri S., M. Garetti, M. Macchi, M. Taisch, 2000. An experimental benschmarkin,g of two multi-agent architectures for production scheduling and control. *Computers in Industry*, **43**, pp. 139-152.

Corbier, F, 1989. Modélisation et émulation de la partie opérative pour la recette en plateforme d'équipement automatisés. Thèse de l'université Henri Poincaré.

Chirn, J. and D. McFarlane, 2000. Application of the Holonic Component Based Approach to the Control of a Robotic Assembly Cell, *Proc. IEEE Conference on Robotics and Automation*, San Francisco.

Fletcher, M., D. McFarlane, A. Lucas, J. Brusey and D. Jarvis, 2003. The Cambridge Packing Cell – A Holonic Enterprise Demonstrator. In: Multi-Agent Systems and Application III (V. Marik, j Müller, M. Petroucek (Eds)), pp 533-543. LNAI 2691, Springer-Verlag, Berlin, Heidelberg.

Fusaoka A., H. Seki, K. Takahashi, 1983. A description and reasoning of plant controlers in temporal logic. in Proceedings of the 8th International conference on artificial intelligence, Karlsruhe, Germany.

Hollocks, B. W., H. T. Goranson, D. N. Shorter, F. B. Verdana, 1997. Assessing Enterprise Integration for Competitive Advantage, in proceedings of ICEIMT'97, International conference on Enterprise modelling and modelling technology, pp 96-107, Springer-Verlag, Berlin, ISBN 3-540-63402-9.

IMS, 2004. Intelligent Manufacturing Systems, Network of Excellence. www.ims-noe.org.

Le Moigne J.-L., 1977. *La théorie du système général*, Presses Universitaires de France.

Morel G., H. Panetto, M. Zaremba and F. Mayer, 2003. Manufacturing Enterprise Control and Management System Engineering: paradigms and open issues. *Annual Reviews in Control,* **27** (2), pp 199-209.

Marik, V., P. Vrba, 2005, Simulation in agent-based control systems: MAST case study. In *Proceedings of the IFAC World Congress,* Vienna.

McFarlane, D., S. Sarma, J. L. Chirn, C. Y. Wong, k. Ashton, 2003. Auto ID Systems and intelligent manufacturing control. *Engineering Applications of Artificial Intelligence*, **16**, pp 293-305.

Searle J.R., 1969. *Speech Acts*, Cambridge University Press.

Valckenaers P. (ed), 2001. Special issue on holonic Manufacturing systems. *Computer in Industry* **46**, pp 233-331.

Van Brussel H., J. Wyns, P. Valckenaers, L. Bongaerts, P. Peeters, 1998. Reference architecture for holonic manufacturing systems: PROSA, *Computer in Industry*, **37**, pp. 255-274.

Van der Zee, à paraître en 2005. Modeling decision making and control in manufacturing simulation. *International Journal of Production Economics*.

Vollman, T. E., W. L. Berry, D. C. Whybark, 1997. *Manufacturing Planning and Control Systems*, Fourth edn, Irwin.

Wegner, P., 1997. Why interaction is more powerful than algorithms. *Communication of the ACM*, **40** (5), pp 80-91.