ORCA : architecture hybride pour le contrôle de la myopie dans le cadre du pilotage des systèmes flexibles de production

Cyrille Pach

N°Ordre : 13/37

Thèse de doctorat

Pour obtenir le grade de Docteur de l'Université de

VALENCIENNES ET DU HAINAUT-CAMBRESIS

Spécialité Automatique et Génie Informatique

Présentée et soutenue par Cyrille PACH

le 10/12/2013, à Valenciennes.

Ecole doctorale :
Sciences Pour l'Ingénieur (SPI)

Equipe de recherche, Laboratoire :
Laboratoire : Thermique Ecoulement Mécanique Matériaux Mise en Forme Production (TEMPO)
Equipe : Production, Services, Information (PSI)

ORCA : Architecture hybride pour le contrôle de la myopie dans le cadre du pilotage des Systèmes Flexibles de Production

JURY

Président du jury

THOMAS, André	Professeur à l'Ecole Nationale Supérieure des Technologies et Industries du Bois de Nancy

Rapporteurs

CASTAGNA, Pierre	Professeur à l'Institut Universitaire de Technologie de Nantes
GLEIZES, Marie Pierre	Professeur à l'Université Paul Sabatier de Toulouse
LOPEZ, Pierre	Directeur de Recherche CNRS au LAAS de Toulouse

Examinateurs

THOMAS, André	Professeur à l'Ecole Nationale Supérieure des Technologies et Industries du Bois de Nancy
VALCKENAERS, Paul	Professeur à l'Université Catholique de Louvain (Belgique)

Directeur de thèse

TRENTESAUX, Damien	Professeur à l'Université de Valenciennes et du Hainaut Cambrésis

Co-encadrants

ADAM, Emmanuel	Maître de conférences, Université de Valenciennes et du Hainaut Cambrésis
BERGER, Thierry	Maître de conférences, Université de Valenciennes et du Hainaut Cambrésis

Remerciements

Je tiens à remercier au travers de cette page tous ceux qui ont contribué de près ou de loin à l'accomplissement de ce travail de thèse.

Je tiens particulièrement à remercier mes encadrants. Tout d'abord, je remercie Thierry Berger, pour son encadrement quasi journalier durant ces trois années de thèse. Il est celui qui était là au quotidien pour m'aider à avancer et sans lui cette expérience de trois années n'aurait pas été la même. Je remercie ensuite Damien Trentesaux, mon directeur de thèse, qui m'a accueilli dans son équipe de recherche, m'a fait confiance en me proposant ce sujet de thèse et m'a soutenu pendant ces trois ans. Tout a commencé grâce à lui puisque, sans lui, je n'aurais probablement jamais mis les pieds dans un laboratoire de recherche à la sortie de l'école d'ingénieur. Je remercie ensuite Emmanuel Adam qui m'a apporté son expérience dans le domaine multi-agent et qui a su s'adapter afin de retranscrire nos travaux dans le domaine agent. Finalement je remercie Yves Sallez qui, bien que ne faisant pas partie de la liste de mes encadrants, en a fait plus pour moi que beaucoup d'encadrants n'en auraient fait. Sa place est donc dans ce paragraphe.

Je remercie ensuite l'ensemble des membres de l'équipe TEMPO-PSI qui m'ont accueillis pendant cette thèse et qui m'ont permis de travailler dans une ambiance de travail exceptionnelle. Je remercie en particulier Gabriel Zambrano qui a commencé sa thèse en même temps que moi et avec qui j'ai collaboré de façon étroite pendant trois ans. Je remercie ensuite Thérèse Bonte, qui a participé à tout ce qui est simulation et sans qui le simulateur ne serait surement pas là. Je remercie aussi Abdelghani Bekrar qui m'a permis d'intégrer de la recherche opérationnelle dans cette thèse, alors que je partais de loin dans cette spécialité. Je tiens à remercier aussi dans ce paragraphe les membres de l'AIP PRIMECA de Valenciennes qui ont permis la réalisation de toutes les expérimentations sur la cellule réelle.

Mes remerciements vont aussi aux personnes ayant accepté de rapporter ce manuscrit de thèse. Je remercie tout d'abord Pierre Castagna, qui était souvent présent lors des présentations concernant cette thèse et qui par ses retours m'a beaucoup aidé. Je remercie ensuite Pierre Lopez et Marie Pierre Gleizes qui ont accepté de relire cette thèse afin d'apporter leur expertise dans les domaines de la recherche opérationnelle et des systèmes multi-agents. Je tiens aussi à remercier André Thomas et Paul Valckenaers d'avoir fait parti de mon jury de thèse. De part leur présence dans le jury, ils donnent de la valeur à cette thèse.

Finalement mes derniers remerciements vont à ma famille qui m'a soutenu tout au long de cette thèse. Merci à la femme de ma vie que j'ai rencontrée pendant cette thèse et qui me supporte depuis deux ans. Merci aussi à mes parents et mes frères qui sont toujours là pour moi, ainsi que mes grands parents sans qui je n'aurais surement pas pu commencer cette thèse dans les meilleures conditions.

Table des Matières

Table des Figures

Table des Tableaux

Introduction générale

De nouvelles contraintes appliquées aux systèmes de production tels que le phénomène de *mass customization* ou la prise en compte de l'énergie sont désormais à considérer dans la gestion de la production. Afin d'intégrer ces contraintes, les architectures de pilotage permettant de piloter ces systèmes de production ont besoin d'évoluer. Ainsi être efficace lorsque l'environnement est déterministe ne suffit plus et un système doit pouvoir réagir à l'incertitude et aux perturbations (ordre de fabrication urgent, panne d'une ressource, baisse de la quantité d'énergie disponible…). Cela nécessite une architecture de pilotage plus complexe et un changement dans le mode de pensée des concepteurs et utilisateurs des architectures de pilotage. Afin d'accompagner ce changement de mode de pensée, des applications concrètes doivent illustrer les propositions, assorties d'une méthodologie claire et précise facilitant la mise en œuvre des architectures proposées.

Cette thèse s'inscrit dans le contexte d'étude et développement de nouvelles architectures de pilotage. Les thématiques abordées dans ce travail sont centrées autour de la dualité hiérarchie / hétérarchie des entités d'une architecture de pilotage et notamment de la myopie qui influence la rapidité et l'optimalité des décisions prises par ces entités. Ce travail de thèse aboutit sur la présentation d'une nouvelle architecture nommée ORCA-FMS basée sur une structure à deux niveaux associant une méthode exacte et une méthode approchée réactive. Une implémentation réelle des concepts, basée sur une méthodologie holonique, est proposée. Ces éléments seront validés (i.e. leur validité sera étudiée, puisqu'il est impossible de valider une approche définitivement sur un seul cas d'étude) en simulation puis sur une cellule existante.

Ce mémoire est composé de six chapitres organisés selon le plan suivant :

Le premier chapitre présente le contexte de l'étude et les conditions particulières dans lesquelles évoluent les systèmes de production actuels. Tout d'abord les caractéristiques primordiales que devront posséder les systèmes de production dans un futur proche et à moyen terme sont définies. Le concept clé qui permet de jouer sur la réactivité et l'optimalité d'un système, la myopie, est ensuite identifié, défini puis illustré. Différentes solutions sont ensuite proposées afin de contrôler cette myopie. Une étude des architectures de pilotage est ensuite réalisée et des architectures de pilotage hiérarchique puis hétérarchique existantes sont présentées. Finalement, le pilotage hybride qui apparaît comme la possibilité la plus prometteuse est introduit.

Le deuxième chapitre détaille les architectures de pilotage hybride. Un état de l'art des architectures de pilotage hybride est tout d'abord présenté au regard de la myopie. Cet état de l'art propose quatre sous-classes d'architectures, classées selon le dynamisme et l'homogénéité du pilotage. Cette classification permet de mettre en valeur les deux sous-classes correspondant

à un pilotage dit « dynamique », comme les plus adaptées au contexte présenté chapitre I. Un deuxième état de l'art présente ensuite les mécanismes de basculement des architectures de pilotage hybride de ces deux sous-classes, qui définissent les performances globales et la réactivité de ces architectures.

Le troisième chapitre présente l'architecture générique ORCA qui est proposée dans ce travail de thèse. Cette **A**rchitecture permet un **C**ontrôle **O**ptimisé et **R**éactif d'un système. Le fonctionnement global de cette architecture applicable à de nombreux domaines est d'abord présenté. Les deux modes de fonctionnement sont ensuite présentés ainsi que le basculement permettant de passer d'un mode à l'autre. La composition des différentes couches de cette architecture est ensuite détaillée. La fin de ce troisième chapitre permet de proposer un modèle de mise en œuvre des concepts d'ORCA via le concept d'holon. Les caractéristiques du nouveau modèle holonique proposé sont présentées et le lien entre entités d'ORCA et le modèle holonique est mis en évidence.

Le quatrième chapitre présente l'application de l'architecture générique ORCA à l'ordonnancement réactif et optimisé d'un système flexible de production. Après que le problème à résoudre ait été introduit, le fonctionnement global de cette architecture nommée ORCA-FMS est présenté. Les deux couches d'ORCA-FMS sont ensuite détaillées. Un modèle linéaire résolu par un ILP (*Integer Linear Programming*) compose la couche haute alors qu'une approche basée sur les champs de potentiel compose la couche basse. Le mécanisme de basculement entre ces deux couches est par la suite détaillé et justifié. Finalement, la représentation d'ORCA-FMS avec le modèle holonique proposé est présentée.

Le cinquième chapitre valide l'architecture ORCA-FMS en simulation. Après avoir présenté l'environnement de simulation utilisé, le cas applicatif et le modèle du simulateur, des simulations permettent de valider les différents éléments d'ORCA-FMS. Les performances individuelles des optimiseurs globaux (ILP) et locaux (basés sur les champs de potentiel) sont étudiés au niveau de l'efficacité (notamment le temps total de réalisation de la production) et de la capacité à réagir (temps de résolution de l'ILP par exemple). L'architecture ORCA-FMS est ensuite validée avec deux niveaux d'hybridation : un premier niveau dans lequel l'optimiseur global ne fournit que l'ordre d'introduction des produits aux optimiseurs locaux et un second dans lequel la séquence des machines à atteindre est fournie.

Le sixième chapitre valide l'architecture ORCA-FMS grâce à des expérimentations réelles. Les éléments physiques de la cellule flexible de l'AIP PRIMECA de Valenciennes sont tout d'abord présentés. La composition des entités utilisées est ensuite détaillée notamment les produits et ressources actifs ainsi que le réseau de communication. Les mises en œuvre du concept de champs de potentiel, du comportement du produit actif et de l'ILP sont ensuite présentées. Finalement, le protocole expérimental et les résultats expérimentaux permettant de confirmer les résultats obtenus en simulation via des expérimentations réelles sont détaillés.

Ce mémoire se termine par la présentation des conclusions du travail sur l'approche hybride proposée et sur la proposition de perspectives de recherche.

Chapitre I : Contrôle de la myopie des systèmes de production via un pilotage hybride

1. Introduction

De nos jours, les systèmes de production sont soumis à de fortes contraintes liées au contexte dans lequel évoluent les entreprises (Grabot, 2012, 2011). Ce contexte, changeant, est par exemple lié à des impératifs de développement durable (Mancebo, 2010), l'instabilité financière (Ministère de l'Industrie, 2010), ou encore la raréfaction des matières premières (De Guillebon et Bihouix, 2012 ; Lesourne, 2011).

1.1. Contexte industriel et scientifique

Dans un récent rapport, ayant pour objectif de définir une feuille de route pour la recherche concernant les futurs Systèmes Manufacturiers Intelligent (*Intelligent Manufacturing Systems* ou IMS), la communauté IMS2020 confirme cette évolution et met en avant un ensemble de trois prescriptions, impactant directement les différents niveaux de contrôle des IMS (IMS2020, 2013) :

- ***Production rapide et adaptative centrée sur les utilisateurs.*** Les systèmes de production doivent pouvoir répondre aux besoins (des clients), qui s'expriment en termes de personnalisation, avec un minimum de délais. Cela implique une capacité d'adaptation et de réactivité du système de production ainsi que des cycles de vie des produits.

- ***Chaîne de valeur hautement flexible et auto organisée.*** Cette prescription impose que les systèmes de production puissent s'organiser (au niveau de l'architecture de pilotage et au niveau de l'infrastructure physique (positionnement des machines, système de transport des produits)) afin de permettre la fabrication d'éventuels nouveaux types de produits. Ainsi les futurs systèmes de production doivent être flexibles (ce sont les SFP ou *Flexible Manufacturing Systems* (*FMS*)), reconfigurables (nommés *Reconfigurable Manufacturing Systems* ou RMS) et intelligents (appelés IMS).

- ***Production durable permettant de prendre en compte les changements culturels des individus et des sociétés.*** À l'heure actuelle cela concerne surtout l'énergie et le rapport entre l'industrie et l'environnement.

Il s'ensuit que les IMS du futur devront obligatoirement disposer de capacités d'adaptation et de réactivité tout en conservant des performances globales optimisées. Afin d'atteindre ces capacités et performances, des sujets de recherche sont proposés dans le rapport prospectif IMS2020 notamment aux sections 3.1 et 3.2. Certains de ces sujets de recherche sont cités ci-dessous à titre d'exemple :

- *Control for Adaptability,*
- *Build-to-Order – New Production Planning and Control Models for Complex Individualized Products,*
- *High Performance,*
- *Dealing with unpredictability.*

Ainsi la notion de contrôle est au cœur des prospectives IMS2020. Selon ce rapport le contrôle des systèmes doit être capable de « s'adapter » à des « produits de plus en plus complexes » et potentiellement personnalisables. Cela correspond notamment à la production dite de *mass customization* avec des produits dont on ne connaît pas à l'avance la configuration finale choisie par le client. Le contrôle doit aussi gérer « l'incertitude » en réagissant dans des délais courts (en termes de secondes ou minutes) aux perturbations internes (pannes, maintenances) et externes (fluctuation de l'énergie disponible, commandes clients variables) au système.

Deux exemples illustrent ces propos. Tout d'abord la *mass customization* (et dans une moindre mesure la *mass personalization*) consistant à produire un produit de consommation courante personnalisé par client (ou par catégorie de clients dans le cas de la *mass personalization* par exemple pays, catégorie sociale, malvoyants…). Cette pratique résulte en une connaissance très tardive de la forme définitive du produit par le système de production et donc nécessite une réactivité importante (plus d'informations sur la *mass customization* sont disponibles en **annexe 1.1)**. Le second exemple est lié à la modification du comportement des industriels face à la consommation d'énergie. En effet, la raréfaction des énergies fossiles, l'utilisation d'énergies renouvelables et la sensibilité des consommateurs sur l'empreinte énergétique d'une entreprise, font de l'énergie une contrainte de plus en plus forte à intégrer dans les systèmes de production (cette contrainte est détaillée et illustrée dans l'**annexe 1.2)**.

Ces deux exemples illustrent bien la nécessité pour le système d'avoir des capacités de réaction aux évènements inattendus et d'adaptation aux changements. Le premier implique des commandes clients non connues à l'avance et une combinatoire de produits réalisables très importante (incertitude au niveau des ordres de fabrication et complexité de la production). Le second implique une source d'énergie variable et donc une capacité à intégrer la consommation énergétique du système comme nouveau critère dans les décisions de production classiques (allocation et routage par exemple). Les systèmes permettant de répondre à ces besoins sont appelés « systèmes réactifs » dans la suite de ce chapitre.

1.2. Problématique scientifique

Afin de réagir aux évènements inattendus, une solution couramment proposée consiste à répartir les décisions et l'information entre plusieurs entités composant le système réactif. Le

problème est que cette distribution des décisions et de l'information peut nuire à la performance globale de ces systèmes à cause de décisions basées sur une information incomplète. Ce problème est appelé myopie et est au cœur de la problématique scientifique de la thèse. En effet, les systèmes de production évoluant dans un contexte concurrentiel, ils se doivent de conserver de bonnes performances globales mais rester réactifs. La problématique scientifique est donc de contrôler la myopie pour améliorer les performances globales tout en laissant les entités suffisamment myopes pour disposer d'autonomie décisionnelle afin de réagir aux évènements inattendus.

La résolution de cette problématique a été identifiée comme une des clés pour l'adoption des systèmes de production réactifs dans l'industrie (Leitão et al., 2010), au même titre que, par exemple, la synchronisation entre les flux du système physique et les données virtuelles le représentant (Thomas et Thomas, 2011). Bien entendu les systèmes réactifs doivent aussi être conçus afin de proposer aux industriels une maîtrise des systèmes de contrôle distribué. Ces difficultés sont liées à la nécessité d'adopter un mode de pensée différent (de « centralisé » à « distribué ») (Leitão, 2009a; Pechoucek et Marík, 2008), de démontrer l'efficacité des systèmes de contrôle distribué/réactif (Duffie et Prabhu, 1994; Pechoucek et Marík, 2008) ou encore d'assurer l'évolution des systèmes conventionnels aux systèmes réactifs (Marik et Lazanskı, 2007; Vrba et al., 2011). Nous nous proposons dans le cadre de cette thèse d'étudier le phénomène de myopie et d'apporter des éléments de solution.

2. Le phénomène de myopie

Les premiers éléments de réflexion à propos du phénomène de myopie, dans le domaine manufacturier, sont présentés dans les travaux de Leitão et Trentesaux. Dans ces travaux, les entités[1] d'un système de production réactif sont décrites comme « autonomes, possédant leurs propres objectifs et n'ayant pas de vision globale du système » (Leitão, 2009a). Les travaux de Trentesaux appellent ce phénomène « myopie » et le caractérise comme étant « les conditions qui génèrent un manque de visibilité précise des états futurs » (Trentesaux, 2009). Ce phénomène peut être considéré à l'origine du manque de performances globales des systèmes réactifs. Afin de montrer l'importance du phénomène de myopie et ensuite proposer une définition générale adaptée à notre contexte, une analyse d'autres domaines scientifiques, touchés par ce phénomène, a été réalisée dans la sous-section suivante.

2.1. La myopie dans d'autres domaines scientifiques

Le terme de myopie est issu du **domaine médical (**Larousse[2]) mais a été utilisé dans de nombreux autres domaines. Ainsi la myopie est rencontrée dans :

- **le domaine économique** (Langer et Weber, 2005),

[1] « Entité » : terme générique faisant référence à une unité autonome capable de communiquer, prendre des décisions et agir (Trentesaux, 2009).

[2] http://www.larousse.fr/dictionnaires/francais/myopie/53567

- **l'étude du comportement organisationnel et humain** (Reb et Connolly, 2009),
- **le marketing** (Johnston, 2009),
- **la robotique mobile** (Mataric, 1992),
- **la recherche opérationnelle et plus particulièrement, en programmation dynamique** (Puterman, 1987)

La table 1.I, inspirée de (Zambrano et al., 2013) illustre la myopie en donnant une définition tirée de chacun de ces domaines. La première colonne présente le domaine d'où est tirée la définition, la seconde l'auteur de celle-ci et la troisième la définition relative à la myopie en elle-même.

Table 1.I– La myopie dans des domaines scientifiques autres que celui de la production

Domaine	Auteurs	Définition
Médecine	Larousse	Anomalie de la réfraction oculaire entraînant une mauvaise vue des objets éloignés sans toucher la vision de près.
Economie	T. Langer et M. Weber	*Myopic loss aversion* fait référence au fait que certains décideurs essaient à tout prix d'éviter les pertes financières lors de leurs choix.
Analyse du comportement humain	J. Reb et T. Connolly	*Myopic regret avoidance* concerne les décideurs qui refusent d'avoir un retour d'informations sur les choix qu'ils n'ont pas faits pour éviter d'avoir des regrets.
Marketing	K. Johnston	*Capability myopia* est liée à l'incapacité d'une entreprise à se reconfigurer pour proposer de nouvelles solutions lors de son adaptation à un environnement changeant. *Boundary myopia* caractérise une entreprise en marketing qui a tendance à toujours communiquer et travailler avec le même groupe d'entités en ignorant les nouveaux groupes qui se créent.
Robotique mobile	M. J. Mataric	La myopie représente l'incapacité de prévoir des cycles (répétition d'un ensemble d'actions) ou des oscillations lors de trajectoires ou déplacements de robots mobiles.
RO et programmation dynamique	M.L. Puterman	*Myopic Policy* représente une politique d'optimisation dans laquelle chaque règle de décision ignore les conséquences futures de la décision et choisit l'action qui maximise la récompense immédiate.

Dans tous ces domaines la myopie engendre une perte de performance qu'il faut s'attendre à retrouver dans un système réactif. Une définition de cette myopie, adaptée aux systèmes de production, est proposée dans la section suivante.

2.2. Définition de la myopie des systèmes de production

Les différents travaux de l'équipe Production, Services, Information du Laboratoire TEMPO ont permis de définir la myopie dans les systèmes de production (Pach et al., 2011b; Zambrano et al., 2012). Sur la base de ces travaux, la définition suivante est retenue :

La myopie représente le manque d'information d'une entité sur son futur et celui des autres entités du système.

Dans les systèmes réactifs considérés les décisions sont prises localement. Ainsi ce manque d'information conduit fréquemment à des décisions non satisfaisantes globalement pour le système (Adam et al., 2011). De cette définition deux types de myopie peuvent être dégagés : la myopie individuelle et la myopie collective.

La myopie individuelle représente le manque d'information que possède chaque entité sur son propre futur.

Un exemple de cette myopie est présenté en **annexe 2.1**.

La myopie collective est significative du manque d'information d'une entité sur le futur des autres entités du système.

Un exemple de cette myopie est illustré en **annexe 2.2**.

La myopie est présente à partir du moment où un système est constitué d'entités ayant une certaine autonomie décisionnelle et cherchant à atteindre leurs objectifs à partir d'informations de leur environnement proche. Ainsi lorsque les entités prennent des décisions localement, basées sur un ensemble réduit d'informations locales induisant une performance globalement non optimisée, elles présentent un niveau de myopie élevé. Cependant, elles peuvent réagir rapidement puisqu'elles traitent un faible nombre d'informations. À l'inverse lorsque qu'elles disposent d'une quantité d'informations importante sur le système, elles prennent des décisions optimisées globalement, leur niveau de myopie étant par conséquent très faible, mais sont incapables de réagir rapidement. Supprimer totalement la myopie impliquerait donc une perte de réactivité. Le terme « contrôle de la myopie » est donc utilisé plutôt que correction ou annulation car l'objectif est d'en atténuer les effets indésirables (de façon temporaire ou permanente) tout en conservant la réactivité qu'elle permet (Pach et Zambrano, 2011). Une démarche intégrale permettant le contrôle générique de la myopie contiendrait plusieurs étapes avant le contrôle comme l'observation et la quantification. Cette thèse étant un des premiers travaux sur la myopie elle se concentre surtout sur les moyens de contrôler la myopie et sur l'impact de la myopie et de son contrôle sur les performances du système. Ainsi la quantification de la myopie n'est pas détaillée dans cette thèse même si des pistes sont proposées (par exemple l'indice d'autonomie proposé dans la section 2.2 du chapitre III).

La section suivante traite de la façon dont le système lui-même va être impacté par la myopie.

3. Impact des prescriptions et de la myopie sur le système

Les systèmes de production doivent donc satisfaire deux « contraintes ». Premièrement, ils doivent s'adapter aux nouvelles prescriptions présentées dans la section 1 de ce chapitre. Et deuxièmement, ils doivent conserver une bonne performance globale qui implique un contrôle de la myopie de leurs entités. Ces deux objectifs ont un impact important sur les différents niveaux de contrôle du système de production.

Les prescriptions de la section 1 (*mass customization*, gestion de l'énergie, etc.) forcent l'entreprise à modifier ses stratégies que ce soit en termes de gestion financière, de planification, de prévision voire de ressources humaines. Ces modifications au niveau stratégique ont des répercussions sur les fonctions des niveaux de décision inférieurs tels que le pilotage, le contrôle/commande et le système physique. La figure 1.1 illustre et synthétise ces propos en positionnant les niveaux de contrôle de la norme ISA 95 (cf. (Bolton et Tyler, 2008)) et les interactions entre ceux-ci.

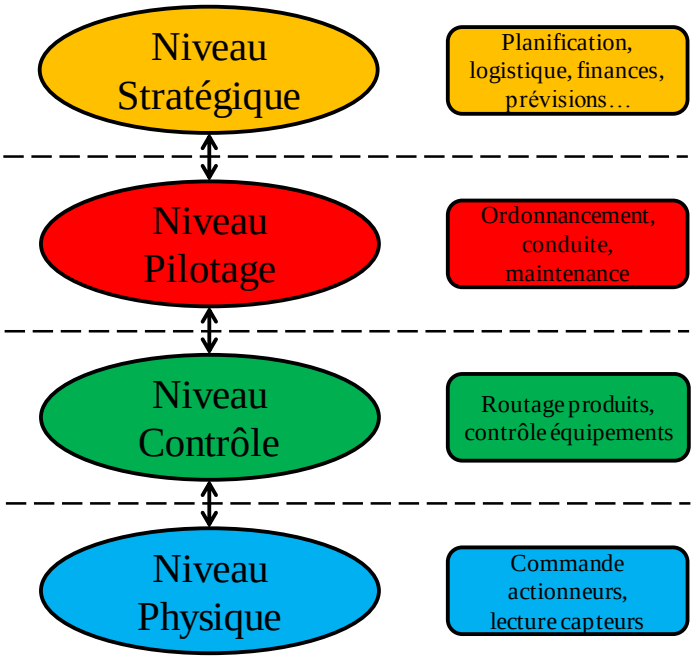


Figure 1.1– Niveaux de contrôle d'un système de production

Ce travail de thèse s'intéresse plus particulièrement au niveau pilotage (qui est le premier niveau de contrôle du système *online)* c'est-à-dire à la fonction pilotage et plus particulièrement aux architectures de pilotage qui supportent cette fonction. Il est donc nécessaire d'étudier les architectures de pilotage existantes et de vérifier leur compatibilité avec d'une part la myopie et d'autre part les prescriptions de la section 1. La section suivante définit rapidement le pilotage des systèmes selon un point de vue fonctionnel avant que ne soient présentées les architectures de pilotages existantes (section 3.2).

3.1. Définition du pilotage des systèmes : vue fonctionnelle

La notion de pilotage a été définie par de nombreux auteurs, notamment (Pujo et Kieffer 2002; Leitão 2004; Wyns 1999). La définition retenue ici est celle de Trentesaux (Trentesaux, 2002) en l'adaptant au contexte actuel :

> Le pilotage consiste à décider dynamiquement des commandes pertinentes à donner à un système soumis à l'incertitude pour atteindre un objectif donné décrit en termes de maîtrise de performances.

En effet, dans la définition initiale de Trentesaux le terme « perturbation » était utilisé à la place d'« incertitude ». Le terme « perturbation » correspond à une perte totale ou partielle de moyens de fabrication subie (comme la panne d'un robot, ou un ralentissement sur le système de convoyage). Ce terme n'est désormais plus assez générique. En effet, la non-connaissance des ordres de fabrication en cas de *mass customization* ou la variation de l'énergie disponible ne sont pas, à proprement parler, des perturbations mais des paramètres (ou contraintes) à considérer dans le pilotage du système de production. Le terme « incertitude » est donc désormais plus adapté et permet d'intégrer à la fois les perturbations et les différentes variations à considérer.

Dans la définition, le terme « système » concerne à la fois des êtres biologiques (organisation hiérarchique ou en communauté d'une société d'individus par exemple) et/ou artificiels (par exemple un réseau d'ordinateurs, un groupe de programmes informatiques répartis dans un *cloud*, un ensemble de robots, machines). Ainsi le pilotage est étudié dans de nombreuses disciplines que ce soit la sociologie, l'économie, la robotique, l'intelligence artificielle ou l'automatique. Dans cette thèse, le pilotage est étudié dans le domaine de l'automatique et plus particulièrement celui des systèmes de production de biens et de services.

Selon les niveaux décisionnels présentés figure 1.1, le pilotage fait le lien entre les stratégies de l'entreprise et le système piloté (niveau contrôle et niveau physique). Par exemple, dans la norme ANSI/ISA95 (Brandl, 2012) jusqu'à 11 fonctions sont différenciées dans le pilotage effectué par un *Manufacturing Execution System* (MES). Ces fonctions sont supportées par des logiciels centrés autour d'une base de données. La fonction considérée comme centrale dans la vue fonctionnelle illustrée figure 1.2 est la conduite (*Production Control* en anglais). Elle fait le lien entre les autres fonctions du système telles que l'ordonnancement de la production, l'approvisionnement, la maintenance ou encore le contrôle qualité et le système piloté. La figure 1.2 illustre ces propos en présentant les liens entre les différentes fonctions d'un MES.

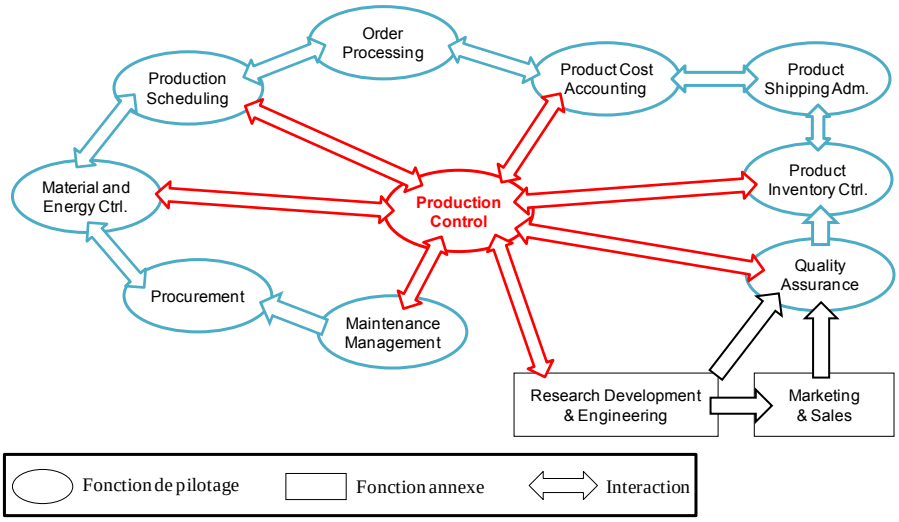


Figure 1.2– Vue fonctionnelle d'un MES (norme isa95)

L'organisation fonctionnelle d'un système de production de type SFP (Système Flexible de Production) diffère de celle d'un MES. Dans l'exemple présenté figure 1.3, les fonctions « conduite » et « ordonnancement » ne sont plus supportées comme dans un MES classique (logiciels autour d'une base de données) mais par des éléments externes au MES plus proches du système physique comme des produits ou ressources actifs (Sallez et al., 2010). Ainsi deux niveaux de pilotage sont distingués sur la figure 1.3 : le niveau pilotage 1 intégrant les fonctions ordonnancement et conduite, et le niveau pilotage 2 contenant les autres fonctions de pilotage supportées de manière dite classique.

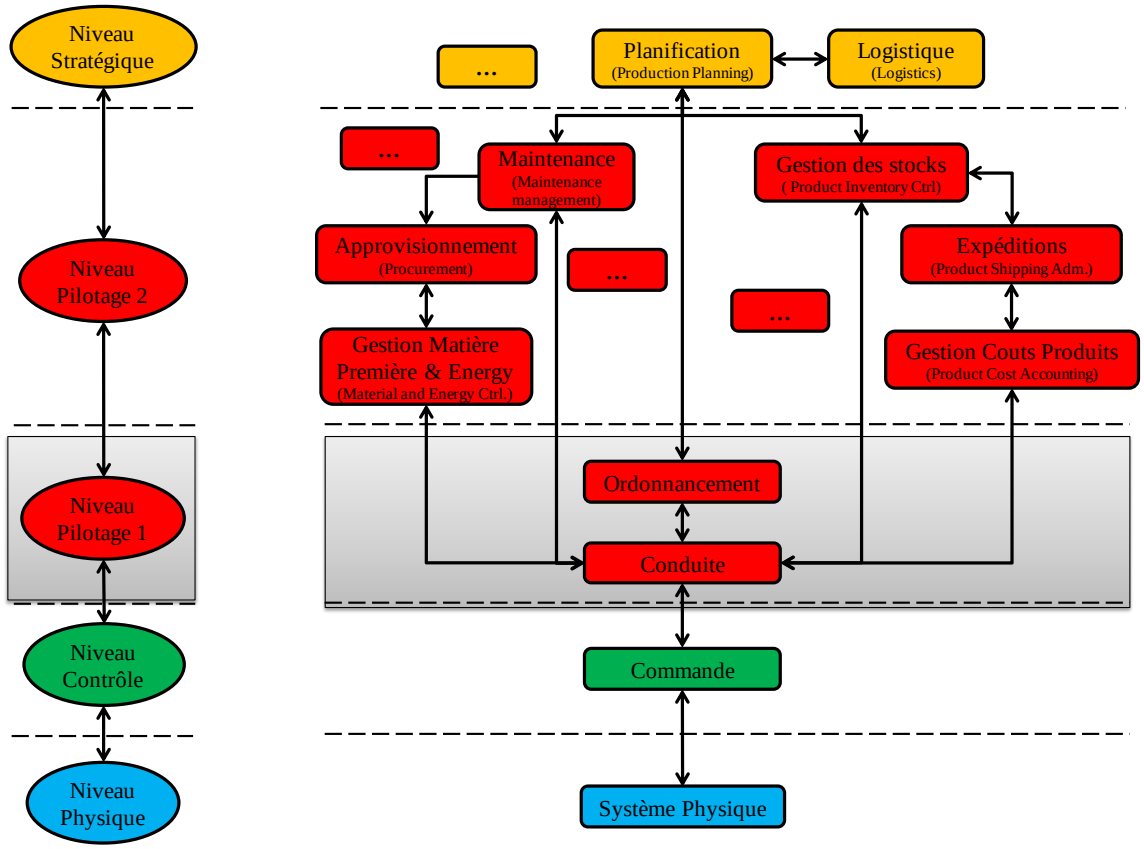


Figure 1.3– Division du niveau pilotage dans un SFP

La zone grisée de la figure 1.3 positionne les fonctions étudiées dans ce travail de thèse. Il s'agit des fonctions de pilotage directement liées à la performance et à la réactivité du système à savoir : **l'ordonnancement** et la **conduite**. Ces deux fonctions forment la colonne vertébrale (détaillée dans l'**annexe 3**) permettant le contrôle du système physique selon les ordres générés en amont par le niveau stratégique (par exemple par un ERP, *Enterprise Resource Planning*). Les niveaux contrôle et physique seront touchés indirectement par les modifications effectuées au niveau pilotage 1. Dans la suite du travail de thèse, les « décisions de pilotage » correspondent donc aux décisions permettant l'ordonnancement et la conduite (niveau pilotage 1 de la figure 1.3). Les « décisions opérationnelles » correspondent au contrôle du système physique par la fonction commande (niveau contrôle, figure 1.3).

La partie suivante présente les différentes architectures de pilotage intégrant les prises de décisions de pilotage (niveau pilotage 1) et les prises de décisions opérationnelles.

3.2. Les architectures de pilotage des systèmes de production

> Une architecture de pilotage est une description de la composition et de la structure d'un système de pilotage.
>
> Senehi et Kramer, 1998

C'est selon ces termes qu'est définie une architecture de pilotage par (Senehi et Kramer, 1998). D'autres définitions comme celles de (Bongaerts et al., 1995; Leitão, 2004) confirment ces notions de « composition » et de « structure ». Dans les SFP, le pilotage (ici ordonnancement et conduite) n'est généralement pas réalisé de façon centralisée (c'est-à-dire par un seul être biologique ou artificiel) mais est distribué et rapproché du système piloté (rapprochement décision-matière). Ainsi, les architectures de pilotage doivent/peuvent être composées d'êtres biologiques ou artificiels possédant une certaine « intelligence » et des capacités de communication/interaction. Ces êtres correspondent à des « entités » selon la définition de Trentesaux (Trentesaux, 2009).

> Le terme entité est un terme générique faisant référence à une unité autonome capable de communiquer, prendre des décisions et agir.
>
> Trentesaux, 2009

Une architecture de pilotage regroupe donc l'ensemble des entités du système et les relations entre ces entités. Dans cette architecture les entités de niveau *i* participent au pilotage de leur niveau et sont soumises au pilotage de niveau *i+1*. Comme vu précédemment, ce terme peut désigner des êtres biologiques (opérateur humain, responsable de production, etc.) ou artificiels. Lorsqu'ils sont artificiels, ils peuvent être par exemple modélisés par les modèles Multi-Agent (Ferber et al., 2004), Holonique (Koestler, 1967), Acteur (Mbobi et Boulanger, 2006), Fractal (Ryu et Jung, 2003) ou Modelon (Okino, 1989). Ces modèles sont présentés et brièvement comparés dans l'**annexe 4**.

La figure 1.4 présente un exemple d'architecture de pilotage d'un système de production générique composée d'entités artificielles contrôlant chacune un élément physique via des décisions opérationnelles. Cet ensemble d'entités est lui-même contrôlé par les fonctions ordonnancement et conduite via des décisions de pilotage.

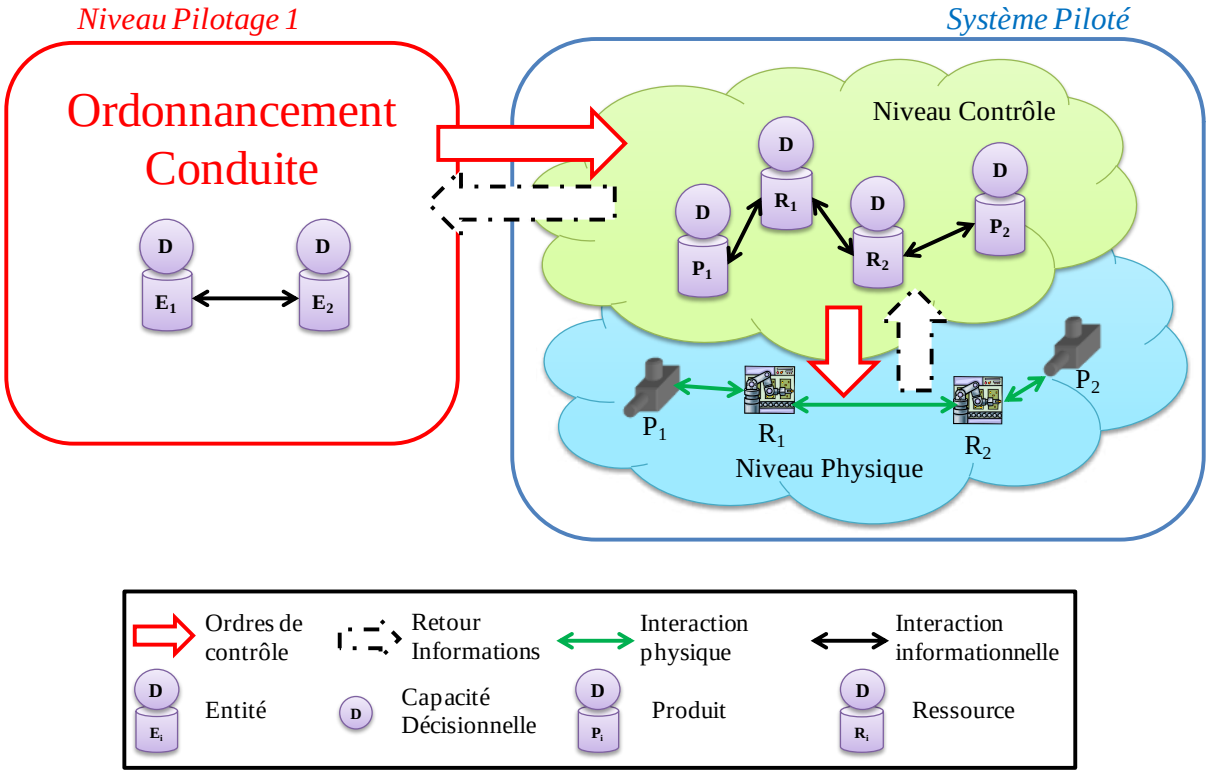


Figure 1.4– Exemple d'architecture de pilotage (niveau pilotage 1 et système piloté)

Les architectures de pilotage ont été divisées en trois classes par Trentesaux comme illustré dans la figure 1.5 extraite de (Trentesaux, 2009). La **classe I** représente les architectures **hiérarchiques** dans lesquelles la relation maître–esclave qui lie les entités est définitive (cf. section 3.2.1). La **classe III** regroupe les architectures **hétérarchiques**. Dans celles-ci, aucune relation maître–esclave n'est statique et toutes les entités peuvent être représentées sur un seul niveau (cf. section 3.2.2). La **classe II** regroupe quant à elle les architectures **hybrides** contenant des entités pilotées de façon hiérarchique et d'autres de façon hétérarchique (cf. section 3.2.4). Ces trois classes sont détaillées dans la suite de cette section.

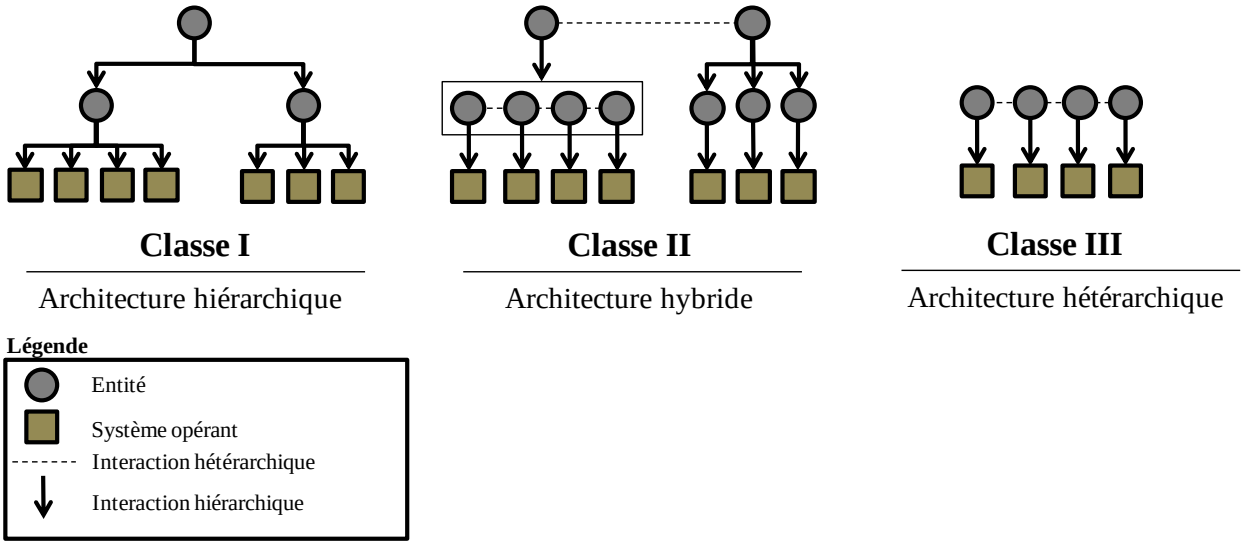


Figure 1.5– Les trois classes d'architectures de pilotage selon Trentesaux

3.2.1. Les architectures de pilotage de classe I (hiérarchique)

Dans la seconde moitié du vingtième siècle, les architectures de pilotage de classe I ont été largement acceptées et déployées par les industriels. Ainsi le concept de CIM (*Computer Integrated Manufacturing*) est largement répandu dans l'industrie depuis les années 1970 et est toujours utilisé de nos jours (Nagalingam et Lin, 2008). Dans ce type d'architecture les décisions de pilotage (telles que l'allocation des produits aux ressources ou le routage des produits) sont prises par l'entité de plus haut niveau hiérarchique qui les propagent ensuite aux entités de niveau inférieur. Ces entités (généralement associées aux produits et ressources physiques) vont appliquer les décisions qui les concernent puis propager les ordres au niveau inférieur et ainsi de suite jusqu'au niveau le plus bas. Quand une décision est appliquée, un retour d'information sur l'état du système est remonté vers l'entité de niveau supérieur. Ainsi dans la classe I, l'entité de plus haut niveau possède (et doit posséder) toutes les informations nécessaires à la prise de décisions globales ce qui garantit une bonne performance globale du système. Les autres entités exécutent les ordres donnés par cette entité supérieure. Un exemple de CIM toujours d'actualité est la « pyramide » ERP-MES-Shop Floor qui est détaillée en **annexe 5**.

Les architectures de classe I sont présentées plus en détails dans les analyses de la littérature proposées par (Buzacott et Yao, 1986; Kenneth et al., 1995; Scattolini, 2009).

3.2.2. Les architectures de pilotage de classe III (hétérarchique)

Plus récemment les chercheurs ont commencé à étudier des architectures de pilotage différentes (Trentesaux et al., 2000) et notamment les architectures de pilotage de classe III comme par exemple PROSIS ou le *flat holonic* (Ounnar et Pujo, 2010; Ounnar et al., 2008). Dans une architecture de classe III chaque entité peut éventuellement influencer les autres entités. Ainsi, si à un instant *t* une entité prend une décision de contrôle ou pilotage, elle peut la transmettre aux autres entités sous forme d'ordres à exécuter. À l'instant *t+1*, il se peut qu'une autre entité fasse de même et transmette sa décision sous forme d'ordres à la première entité. Finalement, toutes les entités se trouvent sur le même niveau de hiérarchie et chacune

peut prendre des décisions sans attendre un ordre d'un niveau strictement supérieur. Ainsi la relation maitre–esclave qui existe dans la classe I est ici volatile dans le temps et chaque entité peut être dans la position du décideur (maître) à un instant *t*, et en position de l'exécutant (esclave) à un instant *t+1*.

Pour éviter que les entités ne prennent des décisions contradictoires et que des conflits (qui doivent alors être résolus) apparaissent, des notions comme la coopération et la négociation ont été définies (Camalot, 2000; Taghezout et Zaraté, 2008). De nombreuses architectures de classe III ont été proposées, basées sur différentes approches de coopération/négociation comme le protocole *Contract Net* (Kadera et Tichy, 2009), les approches agents de résolutions de approches agents de résolution de CSP (*Constraint Satisfaction Problems*) distribués comme DAmasCOP (Clair et al., 2008), la stigmergie (Leitão, 2009a; Sallez et al., 2009) ou encore les champs de potentiel (Zbib et al., 2012). Enfin, de l'information supplémentaire peut être fournie aux entités via des mécanismes de simulation (Cardin et Castagna, 2009) ou de partage d'information (Cavalieri et al., 2000). Cette classe III a été étudiée en détails dans plusieurs analyses de la littérature (Babiceanu et Chen, 2006; Lee et Kim, 2008; Shen et al., 2006; Tharumarajah, 1996).

3.2.3. Synthèse sur les architectures de classe I et III

Les nouvelles prescriptions (i.e. la *mass customization*, les courts cycles de vie, l'utilisation d'énergies variables) présentées au début de ce chapitre (section 1) posent des problèmes aux architectures de classe I. En effet, ces architectures sont faites pour optimiser une production dans un environnement déterministe (peu sujet aux variations non prévues). Dans ce contexte déterministe, la myopie de ce type d'architectures est très faible puisque tout est prévu et optimisé. En contrepartie, elles manquent de réactivité et d'adaptation dès lors qu'un évènement inattendu se produit (Borangiu et al., 2012; Bussmann et McFarlane, 1999; Marík et McFarlane, 2005). En effet, l'optimisation étant effectuée par le niveau supérieur, les entités de bas niveau doivent remonter l'intégralité de l'état du système. Une fois toute l'information remontée, le niveau supérieur doit trouver une solution pour l'ensemble du système à partir d'une grande quantité d'information. Lorsque le système est de grande taille ou complexe, le problème à résoudre est difficile et la résolution prend du temps. Le système physique continuant d'évoluer pendant cette résolution, il peut arriver que son état ait changé au moment où la solution est trouvée. Ainsi les systèmes soumis à l'incertitude (perturbations internes et/ou externes fréquentes) deviennent très difficiles, voire impossibles à contrôler par une architecture de classe I. La communauté des chercheurs du domaine de la Recherche Opérationnelle, utilisent généralement des architectures de classe I. Ce problème étant de nature NP-difficile (en anglais *NP-Hard*, cf. (Mati et Xiaolan Xie, 2003)), ces chercheurs le considèrent comme un des plus compliqués à résoudre.

Pour pallier ce problème, les architectures de classe III ont été proposées. Le fait que chaque entité puisse prendre ses décisions de manière autonome permet au système d'être très réactif puisque les décisions sont prises localement, là où elles doivent être appliquées. Ainsi les architectures de classe III permettent par exemple de réallouer instantanément les produits se dirigeant vers une ressource qui tombe en panne ou de s'adapter à de nouveaux produits

(Dilts et al., 1991; Duffie et Prabhu, 1994; Leitão, 2009a). Ces caractéristiques correspondent aux besoins des systèmes de production actuels (Duffie, 1996; Morel et al., 2007; Sauer, 2008; Skobelev, 2011). Cependant, cette réactivité est rendue possible grâce à une prise de décision locale qui utilise peu d'information : les entités sont donc myopes. Par conséquent, les performances globales de ce type d'architecture sont réduites en régime normal et n'atteignent plus celles des architectures de classe I (Leitão, 2009a).

À la lumière de ce raisonnement, les chercheurs ont proposé des architectures hybrides (classe II). Ces architectures sont présentées dans la section suivante.

3.2.4. Les architectures de pilotage de classe II (hybride)

Les architectures de pilotage de classe II sont conçues pour combiner les approches de pilotage de classe I et de classe III afin de bénéficier de l'optimisation globale de la classe I et de la réactivité de la classe III sans toutefois souffrir de la myopie de la classe III (Heragu et al., 2002). La figure 1.6 positionne les trois classes d'architectures selon leurs performances en régime perturbé et non perturbé.

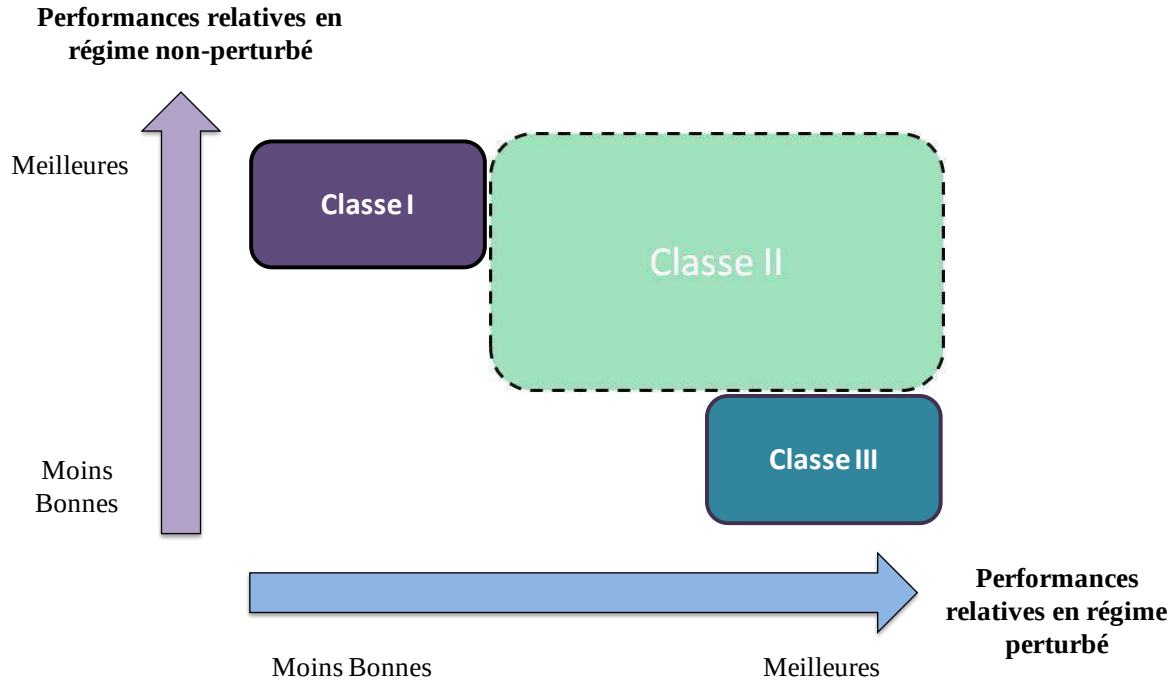


Figure 1.6– Positionnement des classes d'architectures selon leurs performances

Les architectures de classe II sont ainsi *a priori* très prometteuses. Étant donné que notre contribution relève de ce cadre, elles sont étudiées en détails dans le chapitre 2.

4. Conclusion

Le contexte scientifique de la thèse peut être synthétisé par les 3 points suivants :

- De nos jours, l'évolution du contexte concurrentiel de l'industrie manufacturière oblige les industriels à rechercher de nouveaux moyens pour piloter leurs systèmes de

production afin de rendre leurs systèmes de production à la fois performants sur le long terme mais aussi de plus en plus réactifs sur le court terme.

- En rupture avec la classe I qui trouve rapidement ses limites dans ce contexte, la communauté scientifique a cherché à développer des approches de classe III afin de satisfaire ces nouveaux besoins.

- Ces approches de classe III présentent de nombreux inconvénients et les contributions ne sont pas encore « mûres » pour être proposées aux industriels en particulier du fait de la myopie. En effet, l'adoption d'approches à la fois réactives et performantes ne pourra se faire qu'à condition de savoir contrôler correctement la myopie des entités décisionnelles.

L'objectif de ce chapitre était ainsi de présenter tout d'abord ce contexte scientifique. Compte tenu des évolutions scientifiques, technologiques, industrielles, sociétales, reprisent dans les nouvelles prescriptions (ex. *mass customization*, énergie variable) pesant sur les systèmes de production à venir, les architectures de pilotages doivent permettre de piloter un système de manière optimisée et réactive. Les limites connues des classes I et III des architectures de pilotage ayant été présentées, la classe II, celle des architectures hybrides, a alors été introduite. Notre contribution relevant de cette classe, le chapitre suivant étudie en détails ces architectures.

Chapitre II : Typologie et état de l'art des approches de pilotage hybride

1. Introduction

Le chapitre I a souligné l'intérêt potentiel des approches de pilotage hybride pour s'adapter à l'évolution du contexte industriel et contrôler la myopie des systèmes de production.

Le chapitre II présente tout d'abord un état de l'art sur les architectures de pilotage hybride. Une typologie des architectures hybrides est proposée et permet de déterminer quatre sous-classes d'architectures de pilotage hybride. Parmi ces sous-classes, deux sembleront particulièrement adaptées pour contrôler le problème de myopie tout en assurant les performances globales du système.

Ces deux sous-classes spécifiques, qualifiées de dynamiques, sont basées sur un mécanisme de basculement entre plusieurs modes de fonctionnement. Une seconde typologie, dédiée à ces mécanismes de basculement, est donc ensuite proposée afin de positionner les différentes propositions de la littérature.

2. État de l'art des architectures de pilotage hybride au regard de la myopie

Afin de positionner les différentes contributions issues de la littérature, deux critères sont proposés. Le premier critère est appelé « dynamisme du pilotage » et représente la possibilité pour l'architecture de pilotage d'évoluer (e.g. passer d'un pilotage hiérarchique à hétérarchique à un moment donné). Si cette évolution est possible le pilotage est dans ce cas qualifié de « dynamique », sinon il est qualifié de « statique ». Le second critère concerne l'« homogénéité du pilotage ». Si le pilotage s'applique de la même manière pour toutes les entités du système, il est qualifié d' « homogène », sinon il est « hétérogène ». Avec ces deux critères (dynamisme et homogénéité), quatre sous-classes de la classe II peuvent être identifiées. La table 2.I présente ces quatre sous-classes et les architectures les composant. La colonne #1 présente les sous-classes trouvées dans la littérature. La colonne #2 indique si le dynamisme du pilotage est statique ou dynamique. La colonne #3 indique l'homogénéité du pilotage : homogène ou hétérogène. La colonne #4 présente le mécanisme d'optimisation globale permettant le contrôle de la myopie, alors que la colonne #5 présente le mécanisme myope permettant une optimisation locale avec une quantité d'information limitée. Finalement, la dernière colonne précise les références associées aux exemples présentés.

Table 2.I– Les différentes sous-classes hybrides et références représentatives

Sous classes	Dynamisme du pilotage	Homogénéité du pilotage	Entité ou mécanisme d'optimisation globale contrôlant la myopie	Mécanisme myope d'optimisation locale	Références
II-SHo	Statique	Homogène	Ordonnanceur de niveau haut	Demande de re-calcul des contrôleurs de sous-niveau	Tawegoum et al. 1994
			Agent superviseur	Demande de re-calcul des ressources	Ottaway & Burns 2000
			Contrôleur de niveau usine	Enchère des contrôleurs de cellule	Ou-Yang & Lin 1998
			Algorithme de synergie	Plan des agents de niveau bas	Cox & Durfee 2003
			Agent coordinateur	Optimisation à base de colonie de fourmis	Böhnlein et al. 2011
			Ordonnancement basé sur l'exploration d'arbres	Négociation entre agents	Chu et al. 2013
II-SHe	Statique	Hétérogène	Ordonnanceur centralisé	Négociation du niveau bas	Parunak et al. 1985
			Agent médiateur	Négociation des ressources	Maturana et al. 1999
			Agent maitre	Mécanisme d'apprentissage des agents de niveau bas	Yang et al. 2008
			Ordonnanceur de niveau haut + coordinateur	Modification du plan par les entités de bas niveau	Heragu et al. 2002
			Ordonnancement de l'agent ordre	Modification du plan par les agents ressources	Rolòn et Martìnez 2012
			Ordonnancement des ressources goulets	Allocation hétérarchique des autres ressources	Trentesaux et al. 1998
			Ordonnancement d'un groupe d'opérations	Flexibilité séquentielle des opérations dans un groupe	Cardin et al. 2013
			Séquencement de lots de production	Division éventuelle des lots par les agents produits	Herrera, 2011
II-DHo	Dynamique	Homogène	Holon staff	Delegate MAS	Novas et al. 2013
			Holon staff	Contract Net de niveau bas	Raileanu et al. 2012
II-DHe	Dynamique	Hétérogène	Holon superviseur	Stigmergie de niveau bas	Barbosa et al. 2011
			Ordonnancement basé sur l'exploration d'arbres	Approche par champs de potentiel	Zambrano et al. 2011
			Ordonnancement de haut niveau	Agents fourmis d'exploration et d'intention	Valckenaers et al. 2007

2.1. La sous-classe II-SHo (Statique et Homogène)

Cette première sous-classe regroupe les architectures dont le pilotage est **Statique** et **Homogène** (dénoté SHo). Dans cette sous-classe un optimiseur de haut niveau réalise l'ordonnancement de la production et l'envoie aux entités de bas niveau. La différence avec les architectures de classe I réside dans le fait que les entités de bas niveau sont actives et peuvent prendre des décisions et communiquer. Ainsi elles prennent des décisions conformes à l'ordonnancement du niveau supérieur mais peuvent lui demander de recalculer un nouvel ordonnancement, si elles considèrent l'ordonnancement initial dépassé ou inapproprié. Un exemple caractéristique de ce type d'architectures est présenté dans (Tawegoum et al., 1994). Des contrôleurs (i.e. entités) de bas niveau peuvent détecter des perturbations quand elles apparaissent et déclencher un re-calcul auprès du niveau supérieur. Un autre exemple assez

similaire est proposé dans (Ottaway et Burns, 2000). Dans ces travaux les ressources (par exemple les robots) sont capables d'évaluer leurs performances. Si elles considèrent qu'elles ne sont pas utilisées correctement, elles demandent un nouvel ordonnancement à un agent superviseur. Ce type d'architectures est également utilisé dans (Ou-Yang et Lin, 1998). Ce travail propose une architecture hybride dans laquelle la répartition des produits est réalisée grâce à un mécanisme d'enchères intégré dans les entités de bas niveau (*cell controllers*). Lorsqu'un ordre de fabrication est lancé dans la cellule, le contrôleur de haut niveau (*shop floor controller*) démarre un appel d'offres. Chaque contrôleur de bas niveau enchérit selon sa qualité de service et le contrôleur de haut niveau choisit le plus approprié. Dans (Cox et Durfee, 2003), un algorithme de synergie est utilisé pour fusionner les plans d'agents de bas niveau en un plan global considéré comme le meilleur pour le système. Ainsi les entités sont intégrées dans la fabrication du plan final mais elles doivent suivre le plan choisi par le niveau supérieur. Dans (Böhnlein et al., 2011) des agents fourmis sont responsables de la construction de leur plan de route en utilisant un mécanisme d'optimisation de colonie de fourmis. Une fois ce plan réalisé, les agents fourmis contactent un agent coordinateur qui évalue ce plan. Si le plan est acceptable d'un point de vue global, l'agent fourmi est autorisé à l'exécuter, sinon il doit en recalculer un. Finalement (Chu et al., 2014) proposent un exemple d'architecture de sous-classe II-SHo appliquée à des lots de production complexes. Dans cette architecture, une simulation à base d'agents permet d'obtenir une première simulation grâce à des protocoles de négociation entre agents. Puis une heuristique basée sur une exploration d'arbre utilisant de l'information globale permet d'affiner cette solution tout en évitant les solutions myopes proposées par les agents.

Dans la sous-classe II-SHo, une entité de haut niveau prend les décisions finales pour réaliser l'ordonnancement global en réduisant au maximum la myopie. Les capacités des entités de bas niveau sont utilisées pour détecter les perturbations ou construire des ordonnancements locaux ensuite utilisés pour construire le plan global. La hiérarchie est forte et appliquée à toutes les entités. L'occurrence d'une perturbation implique toujours un re-calcul du haut niveau, qui peut prendre des heures à cause de la complexité du problème à résoudre. L'avantage principal est que la myopie est très faible dans ces systèmes, ce qui conduit à un comportement optimal forcé des entités. L'inconvénient principal est le manque de réactivité du système à cause des échanges d'informations (décisions, état du système) importants entre les niveaux de hiérarchie, ce qui est source de retard et le temps que met le niveau supérieur à calculer un plan global.

2.2. La sous-classe II-SHe (Statique et Hétérogène)

Cette sous-classe regroupe les architectures dont le pilotage est **Statique** et **Hétérogène** (SHe). Dans cette sous-classe, l'ordonnancement principal est réalisé par un optimiseur central, mais les entités de bas niveaux sont capables de le moduler jusqu'à un certain point. Ainsi lorsqu'une perturbation apparaît, elles peuvent proposer des changements dans l'ordonnancement, qui doivent ensuite être validés par un niveau supérieur. Par exemple (Parunak et al., 1985) a proposé une architecture nommée *Yet Another Manufacturing System* (YAMS). Cette architecture est composée de plusieurs couches de contrôle et de deux types d'ordonnancement. Tout d'abord un ordonnanceur central avec une vue globale du système

créé un macro-ordonnancement. Ensuite, les entités d'un niveau de contrôle plus bas négocient un ordonnancement détaillé entre elles. Toutes les entités sont soumises aux décisions du contrôleur de haut niveau mais peuvent modifier légèrement celles-ci. Plus tard, (Maturana et al., 1999) ont proposé METAMORPH, une architecture multi-agent hybride. Dans cette architecture, les agents ressources sont capables de communiquer et de négocier un ordonnancement, pendant qu'un optimiseur global prend soin de l'optimisation globale et de la coordination entre agents. L'architecture multi-robot proposée dans (Yang et al., 2008) appartient aussi à la sous-classe II-SHe. L'agent de bas niveau joue un rôle donné par l'agent maître (haut niveau). Une fois un rôle obtenu, chaque agent suit les ordres de haut niveau pour atteindre ses objectifs mais intègre un algorithme de raisonnement. Cet algorithme basé sur des techniques d'apprentissage utilisant un réseau de neurones, permet à l'agent de modifier son comportement et de réagir aux perturbations tant qu'il continue de satisfaire au rôle défini par l'agent maitre. Si l'agent nécessite une modification de comportement importante, il doit faire une demande de changement de rôle. Une autre architecture II-SHe composée de trois niveaux est proposée dans (Heragu et al., 2002). Le niveau haut propose un ordonnancement global qui est envoyé aux niveaux inférieurs. Le niveau le plus bas peut modifier cet ordonnancement pour améliorer ses performances en cours de fabrication après acceptation du niveau intermédiaire qui est responsable de la coordination entre les entités de bas niveau. Ici il y a une hiérarchie permanente entre ces trois niveaux mais le niveau le plus bas peut légèrement dévier de son ordonnancement initial tant qu'il reste conforme aux limites fixées par le niveau intermédiaire. Un autre exemple est décrit dans (Rolón et Martínez, 2012). Ce travail présente le concept de @MES basé sur des agents ordres et ressources utilisant des propriétés d'informatique autonome (*autonomic computing)*. L'agent ordre est responsable de la réalisation d'un ordonnancement global qui est envoyé aux agents ressources. Si un produit semi-fini arrive plus tôt ou plus tard que prévu, les agents ressources peuvent légèrement modifier l'ordonnancement tant que celui-ci ne perturbe pas les autres tâches allouées. Si une perturbation importante apparaît, les agents ressources peuvent demander un réordonnancement à l'agent ordre responsable. Un autre type d'architecture II-SHe est proposée dans (Trentesaux et al., 1998). Ici une entité de niveau supérieur réalise un ordonnancement uniquement pour les ressources goulets alors que les autres entités sont gérées hétérarchiquement. Le pilotage est effectué différemment en fonction des ressources qui sont donc contrôlées de manière hétérogène. (Cardin et al., 2013) propose une architecture II-SHe basée sur le principe d'ordonnancement de groupe. Dans cette architecture un ordonnancement prédictif est tout d'abord réalisé afin d'optimiser la production en ordonnançant des groupes d'opérations. Lors de l'exécution, les opérations au sein de chaque groupe peuvent être interverties ou décalées de manière réactive afin de pouvoir absorber les aléas de production. Finalement, (Herrera, 2011) propose une architecture dans laquelle un plan directeur de production permet de définir a priori des lots de production à lancer dans le système. Lorsque ces lots sont en production, des holons produits peuvent évaluer la validité du plan proposé et décider de fractionner ce lot pour faire face à différentes perturbations.

Dans la sous classe II-SHe, les entités de bas niveau ne sont pas toutes contrôlées de la même façon. Elles disposent d'une marge de manœuvre pour prendre des décisions même si elles doivent toujours respecter le plan global. Elles peuvent légèrement modifier le plan qui leur

est fourni sous acceptation du coordinateur, en ne considérant que certaines informations (comme la séquence des machines ou l'ordonnancement des machines goulets). Ainsi si une perturbation est localisée et ne concerne que peu d'entités de bas niveau, elle peut être gérée localement par les entités concernées. Cette sous-classe est donc plus réactive que la sous-classe II-SHo. L'inconvénient principal de cette sous-classe est que la validation obligatoire par l'entité de haut niveau, des modifications proposées par les entités de bas niveau, qui ne peuvent donc pas réagir librement, nécessite des communications permanentes entre les niveaux. Cette sous-classe peut aussi difficilement gérer les perturbations qui touchent tout ou une grande partie du système puisqu'il faut recalculer l'ordonnancement global.

2.3. La sous-classe II-DHo (Dynamique et Homogène)

Dans cette sous-classe le pilotage est **Dynamique** et **Homogène** (DHo). Les entités de bas niveau subissent une hiérarchie forte (contrôlée par un optimiseur central) tant qu'il n'y a pas de perturbation. Dès qu'une perturbation apparaît, toutes les entités basculent dans un pilotage hétérarchique et réagissent sans intervention de l'optimiseur central. Dans une des récentes mises en œuvre de PROSA, (Novas et al., 2013) présentent une structure collaborative entre un système d'ordonnancement centralisé et une implémentation de PROSA utilisant le concept de système multi-agent délégué (D-MAS). L'ordonnanceur commence par créer un macro-ordonnancement qui satisfait les contraintes globales du système, et l'architecture PROSA affine cet ordonnancement pour créer un ordonnancement temps réel pour les contraintes locales. Un « niveau de conseil » est défini dès le départ et permet aux entités du D-MAS de s'éloigner plus ou moins de l'ordonnancement global. Les entités suivent les ordres de l'ordonnanceur global mais peuvent basculer dans un mode « D-MAS uniquement » lorsque l'ordonnanceur fournit de mauvaises indications au regard de la situation courante. Dans (Raileanu et al., 2012) le système peut basculer entre trois structures différentes de pilotage : Hiérarchique (Hi), Hétérarchique Négocié (HN) et Hétérarchique Non Négocié (HNN). Le système commence en pilotage Hi et bascule en HN si une perturbation intervient. Si la situation continue de se dégrader, le système bascule en HNN. À chaque basculement la garantie d'optimalité de la production diminue mais la réactivité des entités augmente. Si la situation du système s'améliore le système rebascule en HN puis Hi.

La sous-classe II-DHo est très différente des deux précédentes. Au lieu de laisser la décision finale aux entités de haut niveau, l'architecture peut complètement changer en cas de perturbation. En effet, le mécanisme permettant le changement de structure est global, après une perturbation, toutes les entités basculent en hétérarchie. Dans ce cas, les entités de bas niveau ignorent les plans définis par le haut niveau et prennent des décisions autonomes mais beaucoup plus myopes. Le mécanisme permettant le changement de structure est global donc après une perturbation, toutes les entités basculent en hétérarchie. Ainsi ces architectures sont robustes et peuvent réagir aux perturbations rapidement, même si le système entier est concerné. Le problème de ces architectures est le basculement global. En cas de perturbation locale, les entités non concernées sont aussi obligées de passer en pilotage hétérarchique. Comme les performances globales d'un ensemble d'entités myopes sont moins bonnes qu'en suivant le plan global, la performance du système va diminuer.

2.4. La sous-classe II-DHe (Dynamique et Hétérogène)

Cette dernière sous-classe concerne donc les architectures dont le pilotage est **Dynamique** et **Hétérogène**. Dans (Barbosa et al., 2011), l'architecture ADACOR proposée correspond à cette sous-classe. Une fonctionnalité d'optimisation globale est intégrée dans le holon superviseur alors que la réactivité du système repose sur une coordination par stigmergie. Lorsqu'une perturbation apparaît les entités de bas niveau peuvent peut passer du mode hiérarchique au mode hétérarchique. Lorsque l'organisation suit une architecture hétérarchique, les entités prennent les décisions localement sans se référer à l'holon superviseur de haut niveau, afin de rester réactives. L'architecture proposée dans (Zambrano et al., 2011) correspond aussi à cette sous-classe. Dans ces travaux, un ordonnancement est réalisé offline par un mécanisme d'exploration sous forme d'arbre et est envoyé aux entités décisionnelles (en l'occurrence des « produits actifs »). Celles-ci sont capables de basculer dans un mode réactif basé sur des champs de potentiel si une perturbation apparaît. Le basculement permet de rendre certaines entités myopes afin qu'elles puissent prendre des décisions locales rapidement. Ce basculement étant déclenché par chaque entité, celles qui ne sont pas concernées par la perturbation continuent de suivre le plan initial qui leur a été fourni et donc conservent leur performance optimale. Ainsi la myopie est différente selon les entités, ce qui permet un comportement adapté à la perturbation. Le retour en mode normal n'a pas été étudié dans ces travaux. La dernière architecture qui correspond à cette sous-classe est basée sur PROSA et présentée dans (Valckenaers et al., 2007). Dans ces travaux, les holons ordres exécutent un ordonnancement fourni par une entité de haut niveau. Si une perturbation apparaît, les holons ordres deviennent autonomes et réalisent leur propre ordonnancement en utilisant des agents-fourmis d'exploration et d'intention. Ils utilisent la simulation et les intentions pour collaborer et finir les produits qui leur sont associés malgré la perturbation. Le mécanisme de retour en mode exécution n'est pas non plus présenté dans ce papier.

2.5. Synthèse

Les sous-classes II-DHo et II-DHe permettant de changer dynamiquement d'architecture de pilotage correspondent le mieux au contexte de réactivité des systèmes de production actuels (présenté chapitre I section 1). En effet, puisque leur structure est dynamique, le système peut réagir en évitant les communications entre niveaux de hiérarchie et les temps d'attentes liés aux re-calculs de nouveaux plans globaux. La sous-classe II-DHe, qui possède un contrôle hétérogène, permet de garantir un fonctionnement optimal pour les entités non concernées par les évènements inattendus. Ainsi, en cas d'évènement inattendu, ayant une portée locale, seules certaines entités sont rendues myopes ce qui limite la chute de performances globales du système. Cependant, peu de travaux ont été réalisés pour concevoir des architectures de sous-classe II-DHe.

La sous-classe II-DHe apparaît ainsi comme un axe de recherche prometteur. C'est pourquoi, les travaux menés dans cette thèse tentent d'apporter des éléments de solutions pour le contrôle de la myopie des systèmes de production à l'aide d'une architecture hybride de sous-classe II-DHe. Comme indiqué précédemment, les architectures dynamiques (et donc la sous-classe II-DHe) sont basées sur un mécanisme propre qui est le basculement dynamique entre

plusieurs modes de fonctionnement. Les performances et la réactivité du système dépendent du mécanisme de basculement choisi. La section suivante présente donc une typologie des mécanismes de basculement existants.

3. Étude des mécanismes de basculement

Cette section commence par reprendre les architectures hybrides et dynamiques de la typologie (sous-classes II-DHo et II-DHe) proposée dans la section 2 pour étudier les mécanismes de basculement qu'elles utilisent. Des critères de différentiation sont ensuite déterminés et une typologie est proposée afin de faciliter le choix d'un mécanisme de basculement.

3.1. Les mécanismes de basculement de la littérature

Cette section présente en détails les mécanismes de basculement de chacune des architectures hybrides et dynamiques présentés dans la section 2.

Dans l'architecture présentée dans (Novas et al., 2013) et introduite précédemment, un système d'ordonnancement centralisé (CSS) réalise un ordonnancement global qui optimise la performance globale. Cet ordonnancement est ensuite affiné et exécuté par un MES décentralisé (DMES). Si une perturbation apparaît le DMES devient autonome (ie. bascule en mode de fonctionnement autonome) et prend ses propres décisions sans suivre l'ordonnancement du CSS. Après ce basculement, le DMES demande un ré-ordonnancement au CSS et rebascule en mode exécution dès que cet ordonnancement est prêt.

Dans (Raileanu et al., 2012), deux niveaux de basculements différents sont considérés. Tout d'abord, un contrôle hiérarchique (CH) est appliqué lors d'un fonctionnement normal. Si un évènement inattendu apparaît, le système bascule dans un mode de contrôle hétérarchique (appelé hétérarchique négocié (HN)) qui est plus réactif (mais moins optimal) que le mode CH. Si le système gère correctement l'évènement inattendu ou que l'entité de niveau haut arrive à fournir un nouvel ordonnancement prenant en compte cet évènement, le système retourne dans le mode CH. Cependant, le contrôle hétérarchique est lui aussi divisé en deux modes de fonctionnement. Si la situation empire (l'évènement inattendu n'est pas géré ou un deuxième évènement inattendu survient), le système bascule depuis HN dans un contrôle hétérarchique non négocié (HNN) totalement réactif.

Dans (Barbosa et al., 2011), l'architecture intègre une capacité d'optimisation globale incluse dans un holon *supervisor*. Des holons *task* contiennent l'information relative à l'ordre de production et sont responsables de l'exécution des plans fournis par le holon *supervisor*. Si un évènement inattendu apparaît, le holon *task* concerné bascule dans un mode réactif pour gérer l'évènement inattendu. Des phéromones sont déposées et propagées pour informer tous les holons *task* de ce basculement. Lorsque les phéromones ont disparu (par évaporation), le système revient dans le mode initial de contrôle hiérarchique mais ce basculement « retour » n'est pas détaillé.

Dans (Zambrano et al., 2011), une entité de niveau haut réalise un ordonnancement grâce à une exploration d'arbre. Des produits actifs (entités de bas niveau) sont responsables de l'exécution de cet ordonnancement dans un mode de fonctionnement normal. Lors de l'exécution de l'ordonnancement, les produits actifs vérifient que l'état du système correspond à celui prévu au moment de réaliser l'ordonnancement. Si ce n'est pas le cas, ils deviennent autonomes, réalisent l'allocation des tâches par eux même et ignorent l'ordonnancement initial. Dans cette architecture, un produit actif est lié à un produit à fabriquer. Ainsi chaque produit actif est responsable du basculement du produit (à fabriquer) qui lui est associé et le basculement est local. Aucun re-calcul n'est déclenché par l'entité de niveau haut et donc le retour à un « mode exécutant » n'est pas présenté dans ces travaux.

Finalement, dans (Valckenaers et al., 2007), les holons *order* exécutent un ordonnancement fourni par une entité de haut niveau. Si une perturbation apparaît, les holons *order* deviennent autonomes et réalisent leur propre ordonnancement en utilisant des agents-fourmis d'exploration et d'intention intégrés dans un *holonic MES*. Ils utilisent la simulation et les intentions pour collaborer et finir les produits qui leurs sont associés malgré la perturbation. Le mécanisme de retour en mode exécution n'est pas non plus présenté dans ce papier.

Afin de positionner tous ces mécanismes, une typologie est proposée dans la section 3.2.

3.2. Proposition d'une typologie

Les contributions présentées dans la sous-section précédente utilisent toutes des basculements entre différents modes de fonctionnements. Trois critères permettent cependant de les différencier. Premièrement le processus décisionnel aboutissant au basculement (ou non) peut être déclenché de plusieurs façons. Le déclenchement peut être intégré dans le comportement normal de l'entité et donc déclenché périodiquement, ou vu comme une interruption de ce comportement normal lorsqu'un évènement particulier survient, voire combiner ces deux modes de déclenchement. Deuxièmement, le processus décisionnel de basculement peut être localisé soit au sein des entités du niveau de décision haut de l'architecture, soit au sein des entités du niveau de décision bas. Troisièmement, le basculement d'une architecture peut être homogène ou hétérogène et donc concerner l'ensemble ou seulement une partie des entités. Ces critères sont détaillés ci-dessous.

Étant donné que chaque approche intègre plusieurs basculements (par exemple passage d'un mode de pilotage hiérarchique à un mode de pilotage hétérarchique), les critères doivent être appliqués à chaque basculement (transition entre deux modes).

Déclenchement du processus décisionnel de basculement

Le premier critère détermine la façon dont le processus décisionnel décidant du basculement (ou non) est déclenché. Ceci peut se faire selon trois modalités :

- Déclenchement Périodique : le processus décisionnel de basculement est intégré dans le fonctionnement normal de l'entité et est déclenché de manière régulière/périodique.
- Déclenchement sur Évènement : le processus décisionnel de basculement est déclenché lorsqu'un évènement inattendu externe à l'entité survient.

- Déclenchement Mixte : le processus décisionnel de basculement peut être déclenché périodiquement et sur évènement.

Localisation du basculement

Ce second critère concerne la localisation du processus décisionnel de basculement. Ce processus est ici considéré par soucis de simplification comme atomique, non décomposable (cf. **annexe 6**). Il peut être situé sur un des deux niveaux suivants :

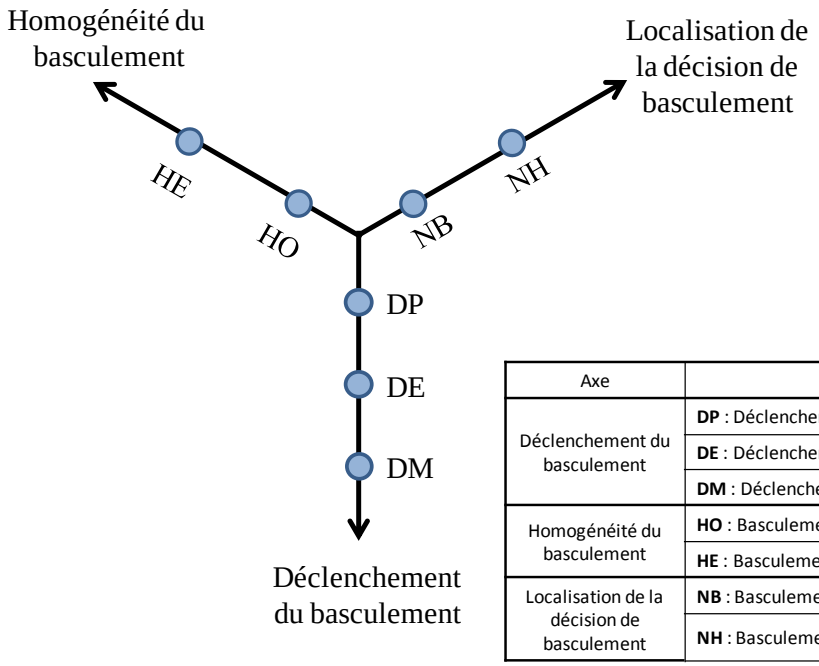
- Niveau Haut : la décision de basculement est prise au niveau supérieur.
- Niveau Bas : la décision de basculement est prise au niveau inférieur.

Homogénéité du basculement

Le dernier critère correspond à l'homogénéité du basculement. Ce critère répond à la question : « une fois la décision de basculement prise, les entités du mode actuel doivent-elles toutes basculer dans le nouveau mode ou certaines peuvent-elles rester dans le mode actuel ? ». Deux valeurs peuvent être prises par ce critère :

- basculement Hétérogène (HE) : seules certaines entités basculent dans le nouveau mode de fonctionnement.
- basculement Homogène (HO) : toutes les entités doivent basculer dans le nouveau mode de fonctionnement.

Les trois critères peuvent être représentés de manière graphique comme proposé figure 2.1.



Axe	Coordonnées
Déclenchement du basculement	**DP** : Déclenchement Périodique
	DE : Déclenchement sur Evènement
	DM : Déclenchement Mixte
Homogénéité du basculement	**HO** : Basculement Homogène
	HE : Basculement Hétérogène
Localisation de la décision de basculement	**NB** : Basculement localisé au Niveau Bas
	NH : Basculement localisé au Niveau Haut

Figure 2.1– Typologie des mécanismes de basculement (*switch*)

La classification des mécanismes de basculement d'une architecture peut aussi être décrite de manière littérale en présentant les critères dans l'ordre suivant : déclenchement, homogénéité et localisation de la décision de basculement. Ainsi un mécanisme de basculement X qui est déclenché sur évènement, qui est homogène et qui est localisé au niveau haut, est représenté

de la façon suivante : Y_X = {DE, HO, NH}. Une approche Z composée de deux mécanismes de basculement de type X est représentée de la façon suivante : Y_Z = { DE, HO, NH | DE, HO, NH }. Dans le cas où les mécanismes de basculement sont strictement identiques au regard des trois critères une étoile peut remplacer les critères afin d'alléger l'écriture ainsi :

Y_Z ={ DE, HO, NH |*}.

3.3. Classification des architectures de la littérature avec la typologie proposée

Dans (Novas et al., 2013), les deux processus de basculement sont déclenchés périodiquement, les décisions sont prises par le DMES qui est l'entité de niveau bas et le basculement est homogène. Ainsi cette approche peut être classée de la façon suivante : Y_{Novas} = { DP, HO, NB | * } comme représenté figure 2.2.

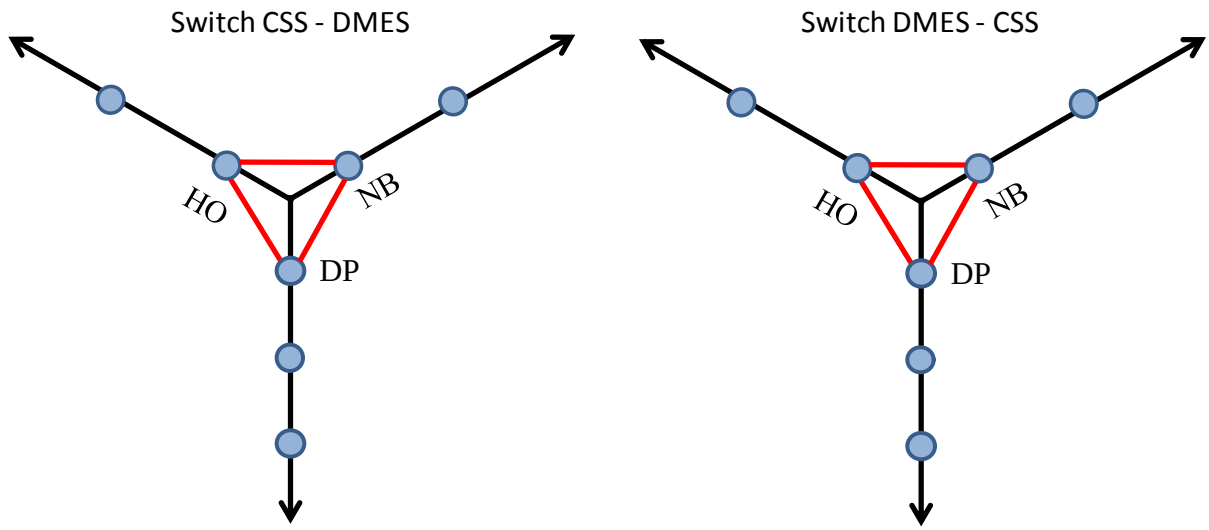


Figure 2.2– Classification de Novas et al., 2013.

Dans (Raileanu et al., 2012) les basculements entre Contrôle Hiérarchique et hétérarchique sont déclenchés périodiquement, sont homogènes et la décision de basculement est prise par une entité de niveau haut. Ce premier niveau de basculement est représenté par :
$Y^1_{Raileanu}$ = { DP, HO, NH | * }.

Le second niveau de basculement entre Hétérarchie Négociée et Hétérarchie Non Négociée est identique au premier : $Y^2_{Raileanu}$ = { DP, HO, NH | * }. Les figures 2.3 et 2.4 représentent respectivement le positionnement d'$Y_{Raileanu}^1$ et d'$Y_{Raileanu}^2$.

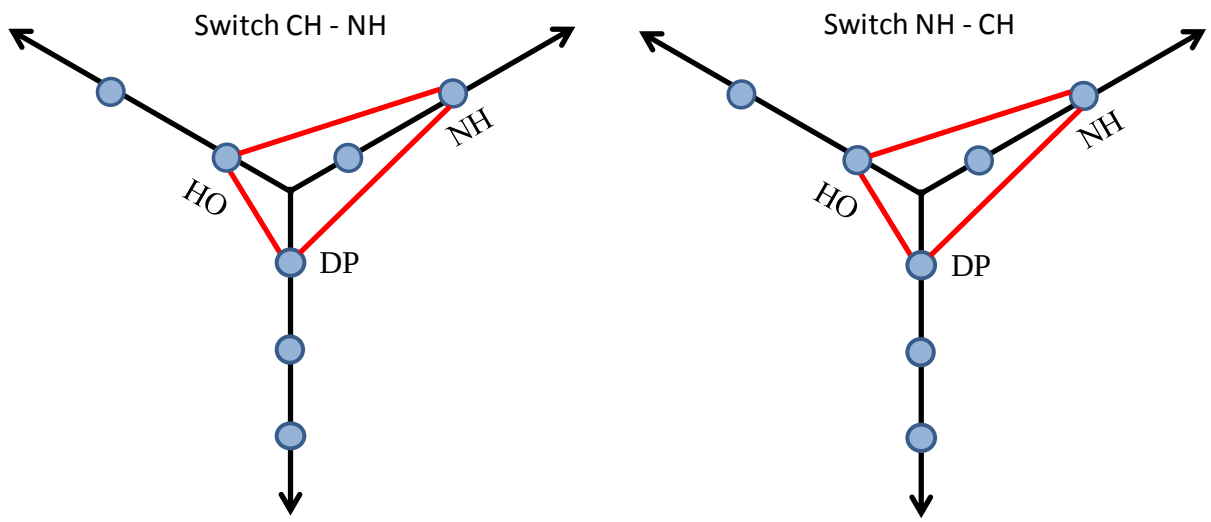


Figure 2.3– Classification du basculement 1 de Raileanu et al. 2012.

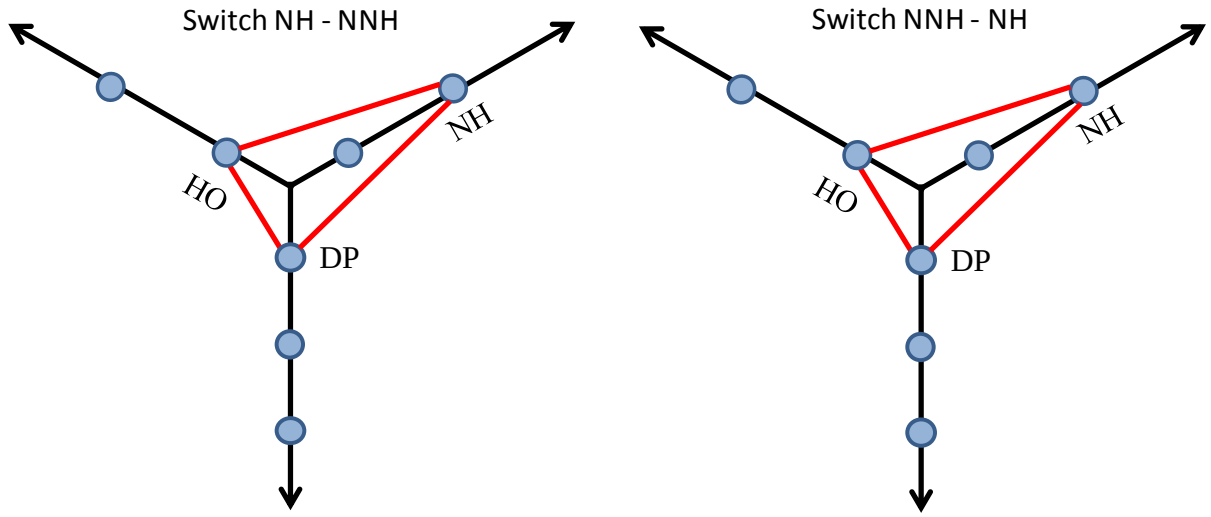


Figure 2.4– Classification du basculement 2 de Raileanu et al. 2012.

Dans (Barbosa et al., 2011), le basculement en mode réactif est déclenché sur évènement (phéromones) par l'entité de niveau haut (holon *supervisor*) et est hétérogène. La question du retour en mode hiérarchique n'est pas détaillée. La classification de cette approche est donc la suivante : $Y_{Barbosa}$ = { DE, HE, NB } comme représenté figure 2.5.

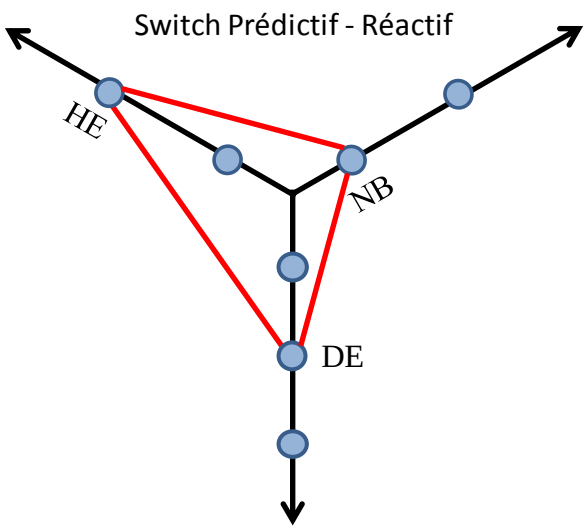


Figure 2.5– Classification de Barbosa et al. 2011

Dans (Zambrano et al., 2011), le basculement du mode « prédictif » au mode « réactif » est déclenché périodiquement, la décision est prise par chaque entité de bas niveau (produit intelligent) pour elle-même (basculement hétérogène). L'architecture ne revient pas dans un état prédictif. Cette architecture est classée de la façon suivante :
$Y_{Zambrano}$ = { DP, HE, NB } ou comme sur la figure 2.6.

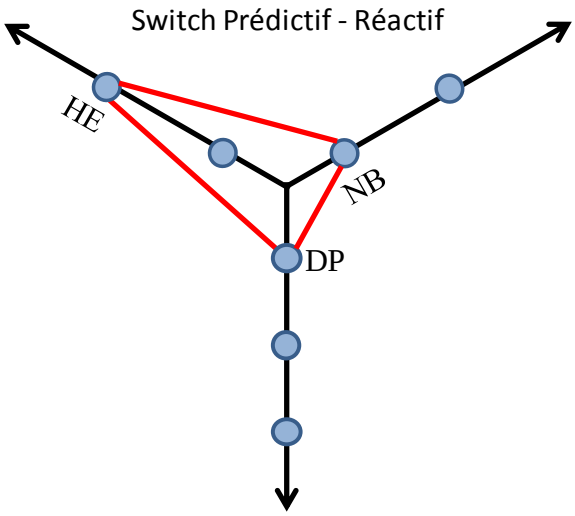


Figure 2.6– Classification de Zambrano et al. 2011

Dans (Valckenaers et al., 2007), le basculement entre ordonnancement de haut niveau et « solutions obtenues par les agents fourmis » est déclenché périodiquement, par les entités de bas niveau de manière hétérogène. Le basculement pour revenir à un ordonnancement de haut niveau n'est pas détaillé dans le papier. Cette architecture peut être classée de la façon suivante :
$Y_{Valckenaers}$ = { DP, HE, NB } ou comme sur la figure 2.7.

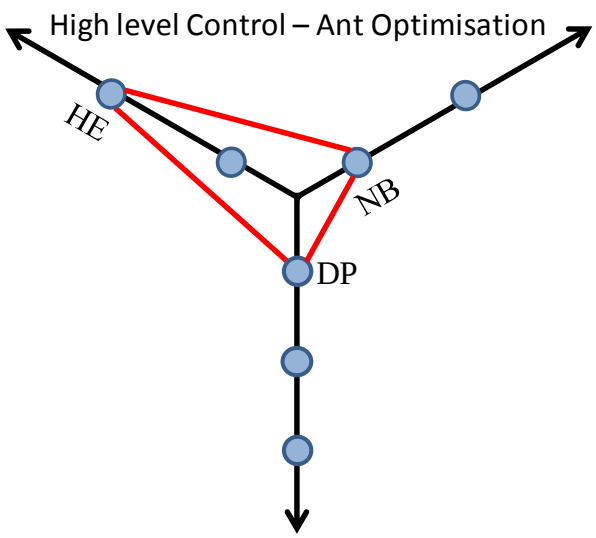


Figure 2.7– Classification de Valckenaers et al. 2007

Tous les mécanismes de basculement présentés précédemment ont été appliqués dans des cas d'études spécifiques. Ainsi une comparaison directe entre performances des mécanismes n'est pas aisée. Cependant, cette typologie permet de les différencier et de se positionner plus facilement au moment du choix du mécanisme lors de la définition de l'architecture permettant le contrôle de la myopie.

4. Synthèse

L'étude menée dans ce chapitre nous a permis d'étudier les architectures hybrides existantes afin d'identifier par la suite un ou plusieurs types d'architectures en adéquation avec le contexte identifié dans le chapitre I.

Pour aider ce choix, deux axes de travail sont possibles comme présentés Figure 2.8. Premièrement, les architectures de classe I peuvent être rendues réactives en conservant leur optimisation globale (flèche en pointillés dans la figure 2.8). C'est le cas dans certaines architectures hybrides statiques qui laissent une marge de manœuvre aux entités de niveau de pilotage bas. Deuxièmement, les architectures de classe III peuvent être rendues performantes en conservant leur réactivité (flèche pleine dans la figure. 2.8). Par exemple, les architectures hybrides dynamiques permettent de contrôler la myopie des entités de niveau de pilotage bas en fonctionnement normal tout en basculant en hétérarchie en cas d'évènement inattendu pour être réactives. Ce travail de thèse s'inscrit dans le deuxième axe et a pour objectif d'améliorer les performances des architectures de classe III en contrôlant leur myopie. Ce choix se base sur deux intuitions. D'une part, en dépit du fait qu'au sein d'un contexte déterministe les architectures de classe I possèdent de meilleures performances que celles de classes III, l'apparition d'incertitude va drastiquement diminuer leurs performances. D'autre part, même si les architectures de classe III souffrent de myopie, elles permettent la modification de leur comportement sans changement majeur de l'architecture de pilotage, ce qui serait nécessaire pour rendre une architecture de classe I réactive. Les architectures de classe III paraissent

donc plus accessibles et il est aisé de les modifier en ajoutant un niveau de pilotage hiérarchiquement supérieur et un mécanisme de basculement pour contrôler leur myopie. La zone « objectif » de ce travail de thèse est représentée par la zone pointillée.

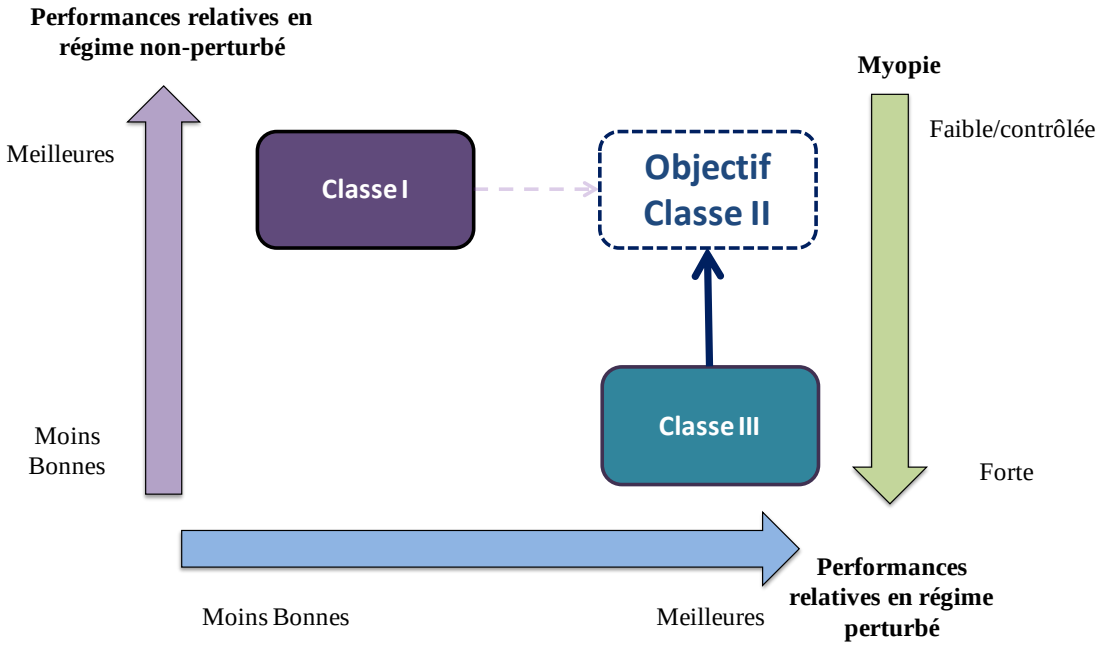


Figure 2.8– Performances relatives des classes I et III

5. Conclusion

Ce chapitre a permis de mettre en valeur les architectures dynamiques qui apparaissent comme un axe de recherche prometteur pour le contrôle de la myopie. Il a ensuite précisé l'étude de ces architectures hybrides dynamiques et notamment de leurs mécanismes de basculement. Ces mécanismes permettent par exemple de basculer d'un mode prédictif à un mode réactif et influence donc directement le contrôle de la myopie et la réactivité des architectures.

Le chapitre III présente l'architecture générique proposée dans cette thèse. Conformément aux constations présentées dans les chapitre I et chapitre II, elle sera hybride, dynamique. Son type de basculement dépendant du cadre applicatif, il sera précisé ultérieurement, dans le chapitre IV, lors de son application dans le domaine manufacturier.

Chapitre III : L'architecture ORCA et le modèle holonique associé

Ce chapitre propose une architecture hybride et dynamique, nommée ORCA : *Architecture for an Optimized and Reactive Control*. Cette explication du terme ORCA résume les préoccupations actuelles relatives au pilotage de systèmes (ex. de production) à savoir réagir/s'adapter aux évènements inattendus tout en contrôlant la myopie des entités du système pour avoir des résultats « optimisés ». L'architecture ORCA est pensée génériquement afin d'être applicable à de nombreux cas d'études.

Ce chapitre débute par la présentation de la démarche de conception d'ORCA et des différents choix effectués lors de cette démarche de conception. L'architecture ORCA est ensuite présentée, puis un modèle permettant de la représenter est recherché. Finalement, le modèle retenu et la représentation d'ORCA via ce modèle est décrite.

1. Choix stratégiques dans la démarche de conception d'ORCA

La démarche suivie pour la conception de l'architecture ORCA comprend les étapes illustrées par la figure 3.1. Ces étapes reprennent et synthétisent les différents choix stratégiques justifiés dans les chapitres précédents et introduisent les choix restant à effectuer.

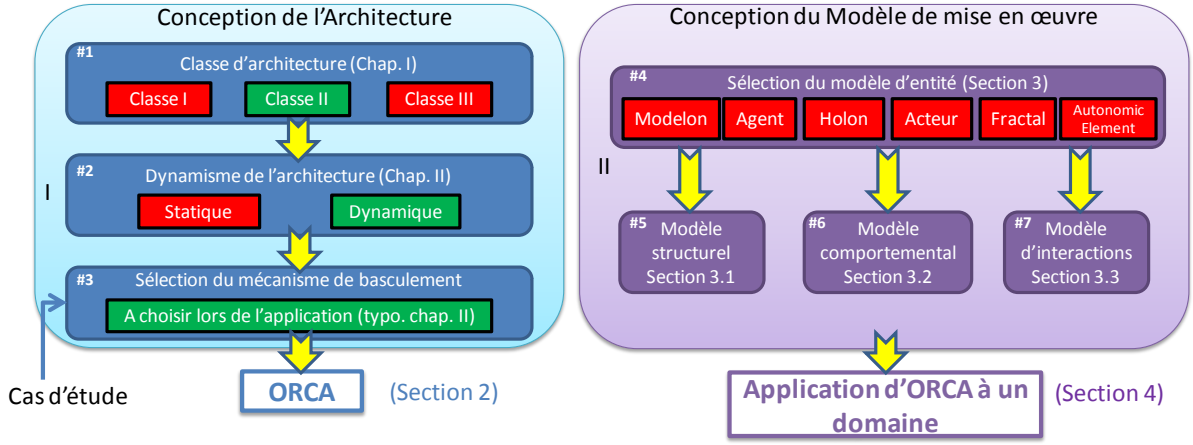


Figure 3.1– Démarche de conception d'ORCA

Tout d'abord la première macro étape de conception (I dans la figure 3.1) a consisté à définir l'architecture en elle-même. Dans cette macro étape, il a fallu tout d'abord choisir entre une architecture de classe I, II ou III (#1 figure 3.1). Conformément aux conclusions du chapitre I, notre choix s'est porté sur les architectures de classe II (hybride). Dans ces architectures deux types d'hybridation existent : une hybridation statique dans laquelle l'architecture n'évolue pas, et une hybridation dynamique qui permet d'évoluer entre plusieurs types de

pilotage. La deuxième étape (#2 figure 3.1) a ainsi consisté à choisir le dynamisme de l'architecture. Conformément aux conclusions du chapitre II notre choix s'est porté sur une architecture dynamique. L'hybridation des architectures dynamiques repose sur un mécanisme de basculement entre plusieurs modes de fonctionnement de l'architecture. La troisième étape de conception (#3 figure 3.1) consiste donc à définir le mécanisme de basculement. Afin d'effectuer ce choix, une typologie des mécanismes de basculement a été proposée dans la section 3 du chapitre II. Comme le choix du mécanisme de basculement dépend du cas applicatif, il sera spécifié dans le chapitre suivant. Sur la base des résultats de ces trois étapes, l'architecture ORCA est présentée dans la section 2 de ce chapitre.

Une fois l'architecture déterminée, une deuxième macro étape est nécessaire. Cette étape est rendue nécessaire par le fait que peu d'architectures de pilotage de classe II et III, directement utilisables par les industriels, existent dans la littérature (Skobelev, 2011). Les architectures conçues en recherche ne sont généralement pas mises en œuvre réellement (Jarvis et al., 2006), ou pas suffisamment éprouvées pour permettre de valider leur comportement dans le milieu industriel (Vrba et al., 2011). Ce manque de travaux significatifs pour les industriels est souvent considéré comme l'un des freins les plus importants à l'utilisation de ces architectures par les industriels (Leitão, 2009b; Marík et McFarlane, 2005; Pechoucek et Marík, 2008). C'est pourquoi la seconde macro étape (II dans la figure 3.1) propose un ensemble de modèles favorisant la mise en œuvre réelle de l'architecture ORCA. Le modèle holonique a été sélectionné pour représenter les entités d'ORCA conformément aux conclusions de la section 4 du chapitre I (#4 figure 3.1). Dans ce chapitre, il faudra donc tout d'abord rechercher un modèle structurel pour chaque entité comme présenté section 3.1. (#5 figure 3.1). Puis, l'étape suivante consistera à rechercher (section 3.2) un modèle comportemental des entités (#6 figure 3.1). Finalement, un modèle définissant les interactions entre ces holons sera recherché section 3.3 (#7 figure 3.1). Le modèle holonique résultant, qui intègre la modélisation de la structure, du comportement des entités et de leurs interactions, sera alors présenté dans la section 4 de ce chapitre.

La section suivante présente l'architecture ORCA.

2. L'architecture hybride ORCA

Cette section présente l'architecture ORCA (*Architecture for an Optimized and Reactive Control)* correspondant aux spécifications des chapitres I et II.

2.1. Présentation générale d'ORCA

ORCA est divisée en trois couches principales : la Couche Système, la couche de Contrôle Local et la couche de Contrôle Global. Cette division des couches de contrôle est justifiée par le souhait d'avoir le moins de couches possibles et de concevoir une architecture à la fois :

- **réactive**, les décisions sont prises localement, là où elles doivent être appliquées comme dans les architectures hétérarchiques,

- **optimisée,** la myopie des entités prenant des décisions locales est contrôlée comme peuvent le faire les holons *staff* de PROSA ou *supervisor* d'ADACOR ce qui permet d'optimiser la performance globale.

La couche système permet de représenter le système réel dans l'architecture. La composition de ces couches est la suivante :

- La Couche Système (CS) est le siège de phénomènes physiques et de processus. Elle est composée d'un ou plusieurs éléments et du contrôle de bas niveau (i.e. régulateurs, asservissements, automatismes…) permettant « d'animer » ces éléments.

- La couche de Contrôle Local (CL) est composée de plusieurs optimiseurs locaux (possédant une ou plusieurs stratégie(s) d'optimisation) ayant une vision partielle du système (ils se basent sur des informations locales). Leur objectif est de réagir/s'adapter aux imprévus qui apparaissent sur la CS en prenant des décisions en temps réel. Ainsi ils fournissent une solution possible à partir des données locales qu'ils ont sur la CS. Chaque optimiseur local de la CL est associé à un ou plusieurs élément(s) de la CS. Un ensemble élément(s) + optimiseur local forme une entité (définie dans le chapitre I).

- La couche de Contrôle Global (CG) possède une vue globale de la CS (i.e. a une connaissance de tout le système) et est composée d'un optimiseur global pouvant utiliser une ou plusieurs stratégie(s) d'optimisation. Son objectif est de garantir de bonnes performances globales grâce à une optimisation basée sur une prédiction de l'état de la CS.

Les informations sur l'état du système peuvent être remontées de la couche système aux couches de contrôle global et local. La couche de contrôle local peut aussi informer la couche de contrôle global de ses décisions. La figure 3.2 illustre les différentes couches d'ORCA et les éléments qui les composent.

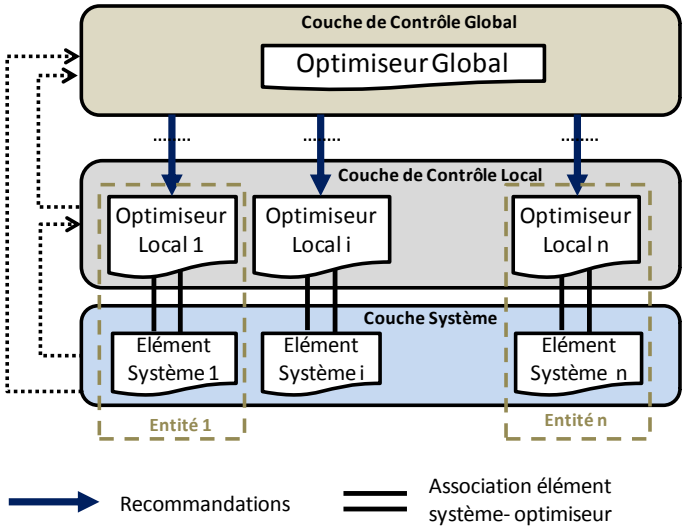


Figure 3.2– Composition de l'architecture ORCA

Les couches de contrôle (global et local) peuvent être récursivement décomposées en d'autres couches de contrôle. La figure 3.3 illustre cette récursivité en présentant une architecture ORCA plus complexe. Les éléments systèmes 1, 2 et 3 sont gérés par les optimiseurs locaux 1.1.1, 1.1.2 et 1.1.3. Ces optimiseurs locaux sont eux même gérés par l'optimiseur global 1.1. Cette structure forme une première architecture ORCA de niveau 1. De la même façon, les éléments systèmes 4, 5 et 6, ainsi que les optimiseurs 1.2.1 et 1.2.2 associés, et l'optimiseur global 1.2, forment une deuxième architecture ORCA de niveau 1. Finalement ces deux architectures ORCA de niveau 1 forment la couche de contrôle local de niveau 2 qui est gérée globalement par une couche de contrôle global de niveau 2 (composée de l'optimiseur global 1). Les deux architectures ORCA étant des optimiseurs locaux dans l'architecture du niveau 2, ils peuvent bien entendu basculer en mode autonome pour permettre la coopération au sein de l'architecture.

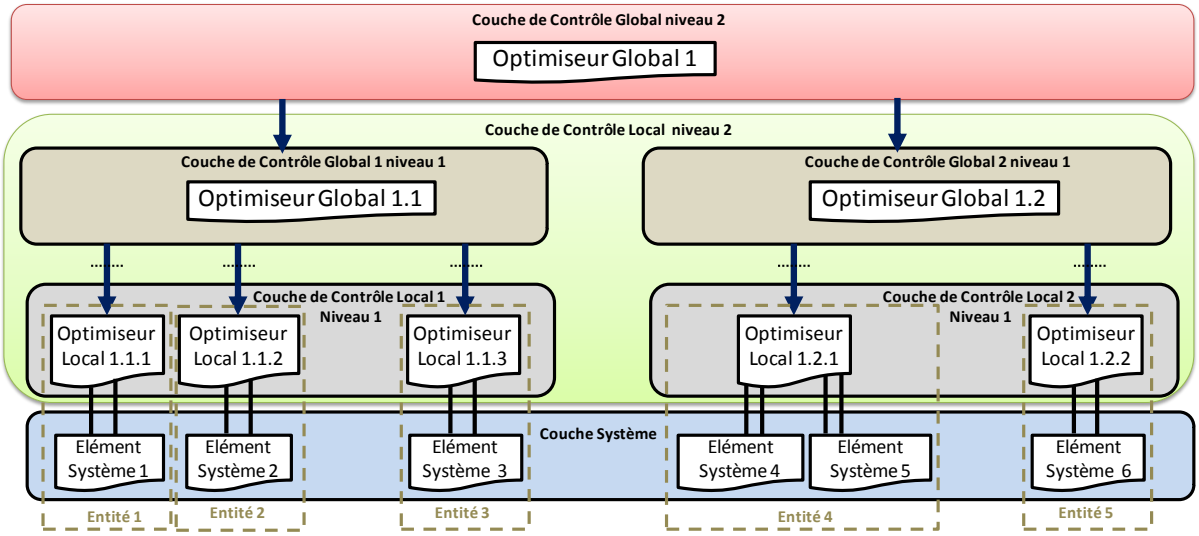


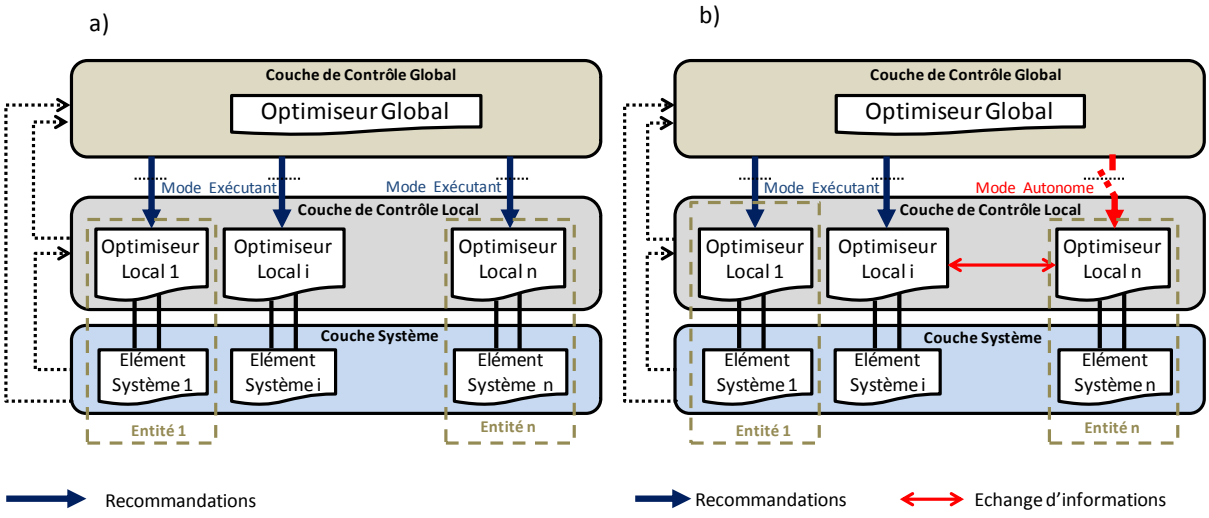
Figure 3.3– Récursivité de l'architecture ORCA

2.2. Modes de fonctionnement des entités d'ORCA

ORCA est une architecture dynamique dont les entités possèdent deux modes de fonctionnement : le mode exécutant (les décisions de pilotage sont prises par l'optimiseur global et appliquées par l'optimiseur local de chaque entité) et le mode autonome (les décisions de pilotage sont prises par l'optimiseur local). Ils sont représentés sur la figure 3.4. Lorsqu'une entité est dans le mode exécutant, elle est contrôlée hiérarchiquement (cf. toutes les entités de la figure 3.4a). La CG optimise de façon prédictive l'utilisation du système et transmet des recommandations (ou ordres) à la CL. Ces recommandations peuvent être par exemple une séquence de machines dans le domaine manufacturier ou une « feuille de route » dans le domaine de la logistique. Chaque optimiseur local de la CL suit les recommandations transmises par la CG le concernant et gère le fonctionnement de sa propre entité. Il vérifie aussi que l'état de la CS correspond à ce qui a été prévu par la CG.

Si un optimiseur local détecte un évènement inattendu, il déclenche le basculement en mode autonome de l'entité à laquelle il appartient (cf. entité n de la figure 3.4b). En mode autonome, la CL contrôle complètement le comportement de l'entité, est responsable de

l'optimisation (locale) de son/ses élément(s) de la CS et essaie d'atteindre les objectifs initiaux de l'entité. La CG ne contrôle plus le comportement de l'entité et l'évènement inattendu est donc géré localement. Ce mode autonome est représenté sur la figure 3.4b par la rupture de la liaison entre entité et CG et la création d'une liaison hétérarchique pour obtenir de l'information de proximité sur l'état actuel du système. Puisque le mode de fonctionnement est défini localement dans la CL, les deux modes peuvent coexister dans le système et le basculement d'une entité en mode autonome ne provoque pas nécessairement le basculement des autres entités.



Figures 3.4a et 3.4b– Les deux modes de fonctionnement de l'architecture ORCA

Afin de caractériser le régime de fonctionnement du système, un indicateur A_s est proposé. Cet indicateur permet également d'obtenir une première quantification de la myopie puisque les entités autonomes ne sont plus contrôlées globalement et donc myopes. Il est calculé de la façon suivante :

$$A_S = \frac{Nombre\ d'entités\ en\ mode\ autonome}{Nombre\ total\ d'entités\ dans\ le\ système} \qquad (1)$$

Trois cas significatifs apparaissent :

- $A_s = 0$ – Toutes les entités du système sont en mode exécutant. Le pilotage du système est effectué avec une vision globale et prédictive. Cette organisation favorise la performance globale avec une myopie totalement contrôlée.
- $A_s = 1$ – Toutes les entités du système sont en mode autonome. Le pilotage du système est effectué avec une vision locale et réactive. Cette organisation favorise la réactivité avec une myopie maximale.
- $0 < A_s < 1$ – Le système est en fonctionnement hybride et contient des entités en mode autonome et d'autres en mode exécutant. Cette organisation permet un compromis entre performance globale et réactivité. Une partie des entités est myope alors que pour le reste la myopie est contrôlée.

La suite de ce chapitre présente le modèle holonique permettant de représenter ORCA et ses entités. Ainsi la section 3 présente les différents modèles structurels, comportementaux et d'interactions existants dans la littérature. La section 4 quant à elle présente le modèle de représentation holonique d'ORCA retenu et le détaille.

3. Vers un modèle holonique pour ORCA

La modélisation choisie pour représenter l'architecture ORCA est holonique. En effet, la modélisation holonique est générique, intègre explicitement la prise en compte du système physique et son intérêt n'est plus à prouver dans la communauté *manufacturing control* depuis la fin des années 90 (notamment depuis l'architecture PROSA (Van Brussel et al., 1998)). La programmation de la partie décisionnelle des holons peut être elle réalisée indépendamment à base d'*autonomic element*, d'agents ou d'acteurs. Cette partie présente les spécifications retenues pour le modèle structurel (i.e. comment est représentée l'architecture à partir du modèle d'entités), comportemental (i.e. comment se comporte l'entité au sein du modèle structurel) et d'interactions (i.e. comment les holons du modèle interagissent entre eux) du modèle holonique représentant ORCA.

3.1. Le modèle structurel

Plusieurs modèles holoniques structurels existent dans la littérature. Les deux modèles principaux sont PROSA (Van Brussel et al., 1998) et ADACOR (Leitão et Restivo, 2006). Dans PROSA les quatre types de holons suivants sont définis : *product*, *resource*, *order* et *staff*. Dans ADACOR les holons sont *product*, *task*, *operational* et *supervisor*. Ainsi la première spécificité de ces approches est que les holons de base sont différenciés. Ainsi un holon *order* (ou *task)* et un holon *resource (operational)* sont totalement différents. Avec une considération « service », un robot par exemple, peut effectuer un assemblage pour un produit et donc « fournir un service ». Pour effectuer cet assemblage il a besoin de matière première disposée dans un stock ou fourni par un opérateur, il est alors « demandeur de service ». Ce qui signifie que ce robot sera représenté de deux manières complètement différentes (*order/task* ou *resource/operational*) dans PROSA ou ADACOR, selon un point de vue ou l'autre, ce qui peut constituer une première limite de ces modélisations. Une autre critique de la littérature (Jarvis et al., 2006) souligne que le concept holonique de récursivité n'est pas bien respecté dans ces modélisations. Ces architectures ont en effet pris le parti de mettre en valeur quatre classes différentes d'entités au lieu de définir une classe générique déclinable en sous-classes. Ainsi les quatre types de holons sont dédiés à des rôles définis a priori et en aucun cas n'intègrent d'autres holons. Une cellule de production (vue comme un tout), ne peut donc pas être représentée de manière récursive (pour des détails sur la récursivité holonique, voir (Suárez et al., 2013)). Finalement, le lien entre ces modélisations et la mise en œuvre effective d'un Système Flexible de Production n'est pas direct. Ainsi la mise en œuvre de ces architectures dans un cadre autre que celui de leur cellule de production d'origine avec des moyens technologiques différents nécessite un travail de re-conception important. De façon générale, la couche « contrôle de haut niveau – pilotage » est définie très clairement. Cependant les couches au dessus (business) et en dessous (opérationnelle) sont moins bien prises en compte et déterminées (Jarvis et al., 2011) ce qui tend à gêner la mise en œuvre. En conclu-

sion, ces deux modélisations ne semblent pas convenir parfaitement pour notre approche en termes de généricité et récursivité. Il est cependant important de souligner que la modélisation PROSA est la base de la majorité des approches holoniques actuelles et ce sont les travaux décisifs effectués sur PROSA par le passé qui nous permettent désormais d'approfondir d'autres aspects.

Une troisième architecture holonique dans la littérature est appelée HCBA (Holonic Component-Based Architecture) et a été proposée par (Chirn et McFarlane, 2000). HCBA est une architecture « bottom-up » et donc construite à partir des éléments physiques de bas niveau. Elle considère deux composants : les produits et les ressources. Les produits entrent dans le système avec un plan détaillé de leur production et essaient d'utiliser les ressources de la meilleure façon possible en négociant avec elles. Il n'y a pas d'optimisation globale du système, cependant l'holon produit intègre une partie « coordination » qui créée un agent (*Work In Process*) pour négocier avec les ressources. Comme pour les deux architectures précédentes, l'entité décisionnelle est déterminée à l'avance dans HCBA. Il s'agit du produit et il n'est donc pas facile de représenter directement un système où les ressources prendraient des décisions de pilotage. De plus, HCBA n'intègre pas de fonctionnalité de haut niveau de pilotage telle qu'un ordonnancement global de la production.

En s'appuyant sur toutes ces remarques, le choix a été fait de concevoir un nouveau modèle structurel (détaillé section 4.1) pour correspondre aux besoins de généricité d'ORCA. Il conserve les caractéristiques de base des holons (autonomie, interaction) tout en intégrant les nouvelles caractéristiques suivantes :

- Élément de base générique (ressource et produit sont des cas particuliers de cet élément),
- Récursivité au sens de la décomposition tout-partie,
- Mise en œuvre facilitée grâce à l'isomorphisme (i.e. il existe un lien direct entre entités du modèle et éléments du système réel).

La section suivante présente le modèle comportemental retenu.

3.2. Le modèle comportemental

ORCA étant volontairement conçue de manière générique dans le sens où elle est conçue afin d'être applicable à différents domaines et cas d'étude, le choix du modèle de comportement ne doit pas imposer de mise en œuvre particulière mais laisser une certaine flexibilité en fonction du cas d'étude. Cette section présente différents modèles permettant de représenter le comportement d'un holon du modèle d'ORCA. Le comportement d'un holon peut être décrit selon deux grandes catégories de modèles.

La première catégorie de modèles contient ceux pouvant être qualifiés de modèle « graphiques ». Dans ces modèles le comportement est la plupart du temps défini par un graphe. Par exemple les réseaux de Petri (Monteiro et Ladet, 2001; Peterson, 1981; Petri, 1980) permettent de modéliser le comportement d'un holon sous forme de places et de transitions. Les

réseaux de Petri permettent généralement de modéliser le comportement dynamique des systèmes à évènements discrets comme les systèmes manufacturiers, les systèmes de télécommunications, les réseaux de transport. De la même façon, les machines à états finis (Gill, 1970) comme les *Statecharts* (Harel, 1987) permettent de représenter graphiquement les différents états d'un système et les transitions possibles entre ces états. Dans ces machines à états finis, les machine de Moore (Moore, 1956) considèrent que les sorties ne dépendent que de l'état courant alors que les machines de Mealy (Mealy, 1955) considèrent que les variables de sorties dépendent de l'état courant et la lettre d'entrée. Le Grafcet (David et Alla, 1992) permet assez similairement de représenter un système sous forme d'étapes et de transitions : un Grafcet est un réseau de Petri particulier avec un seul jeton par place possible et soumis à des règles spécifiques (notamment en cas de conflit).

La deuxième catégorie de modèles contient les « textuels ». C'est le cas des modèles linéaires en recherche opérationnelle comme les (M)ILP ((*Mixed-)Integer Linear Programming*) (Paschos, 2005) ou des modèles de Programmation par Contraintes (Esquirol et al., 1995; Mackworth, 1977) qui permettent de modéliser un système par les contraintes qui le définissent. Dans les systèmes multi-agents, la notion de rôles (Ferber et Gutknecht, 1998) peut être utilisée pour décrire la fonction (rôles organisationnels) qu'occupe un agent dans l'organisation ; ou les comportements (rôles fonctionnels) d'un agent (Bauer et al., 2001).

Une vision générique qui permet différentes mises en œuvre est la vision « rôles ». Cette vision permet d'une part une mise en œuvre graphique notamment par les *JBehaviourTrees* (Bojic et al., 2011) qui proposent des arbres de comportement dédiés au développement sous la plateforme JADE ou *AgentUML* qui étend les modèles de UML, plus particulièrement le diagramme de séquences et le diagramme de classes afin de permettre la représentation des rôles joués par les agents logiciels. D'autre part les rôles permettent aussi une mise en œuvre textuelle à l'aide de plateformes multi-agent comme JADE, NetLogo, RePast ou de langages de programmation usuels comme le C++ et Java. Le modèle holonique d'ORCA utilise donc cette vision « rôles » correspondant à ses besoins de généricité. La partie suivante étudie les modèles d'interactions permettant de représenter les interactions entre les holons du modèle d'ORCA.

3.3. Le modèle d'interactions

L'architecture ORCA a été conçue afin d'être générique. Ainsi le modèle d'ORCA doit permettre l'utilisation de nombreux mécanismes d'interactions entre ses holons. Le mécanisme définitif ainsi que le type de basculement est à choisir en fonction de chaque cas d'étude (cf. chapitre IV).

Cette section énumère une bibliothèque de mécanismes d'interactions de base envisageables pour définir les interactions entre les holons d'ORCA. De façon à « cartographier » ces mécanismes interactions, le concept d' «*Open Control*» développé au sein de l'équipe TEMPO-PSI a été repris. L'*Open Control* (Pach et al., 2012b) définit trois types d'interactions : le contrôle explicite, le contrôle implicite sociétal et contrôle implicite environnemental.

- Le contrôle explicite représente une relation maître–esclave entre entités. Un contrôleur envoie ses ordres à un subordonné qui lui envoie des informations en retour. Ce contrôle peut s'effectuer soit en modifiant les entrées de l'entité soit en ajustant ses paramètres. Cela correspond au contrôle hiérarchique de base avec des ordres du niveau haut qui sont appliqués à un niveau inférieur. Ce contrôle hiérarchique dans le domaine manufacturier est présenté en détails dans (Baker, 1998). Un exemple typique est celui d'un responsable de production qui décide d'arrêter une machine pour effectuer une maintenance sur celle-ci.

- Le contrôle implicite permet d'influencer le comportement d'une entité sans ordre direct ni relation maître–esclave. Deux types de contrôle implicite sont différenciés : le contrôle implicite sociétal et le contrôle implicite environnemental.

    - Dans le contrôle implicite sociétal les échanges entre entités se font directement. Ainsi ce contrôle intègre les mécanismes d'interactions directs comme le protocole Contract-Net (Smith, 1980) utilisé notamment dans le domaine de l'intelligence artificielle distribuée. Dans ce protocole un gestionnaire est responsable de l'exécution d'une tâche et cherche à contacter un ou plusieurs contractant afin d'obtenir la meilleure qualité de service possible. Un autre exemple d'interactions directes est le *Mutual Assistance Protocol* (Polajnar et al., 2012) dans lequel une entité peut aider une autre entité lorsque celle-ci demande de l'aide. De cette façon, les deux entités agissent dans l'intérêt du groupe. Le contrôle implicite regroupe l'ensemble des mécanismes d'interactions basés sur des enchères. Les travaux de (Andersson et Sandholm, 2001) permettent d'avoir une vue générale de plusieurs de ces mécanismes. Finalement dans (Gascueña et al., 2012), deux types d'interactions sont proposées : une interaction hiérarchique où un gestionnaire de véhicules autonomes leur fournit les tâches les plus appropriées en fonction de leurs estimations ; et une interaction de proche en proche entre véhicules autonomes, correspondant à un contrôle implicite sociétal, leurs permettant d'échanger sur leurs estimations à réaliser leurs tâches.

    - Le contrôle implicite environnemental regroupe les interactions pour lesquelles un intermédiaire (souvent l'environnement) est utilisé. C'est le cas pour la plupart des approches bio-inspirées. Ainsi la stigmergie utilisée dans les travaux de (Dorigo et al., 2000) permet à des fourmis d'influencer le comportement des autres en déposant des phéromones dans l'environnement. Les approches basées sur les hormones (Shen et al., 2004) fonctionnent de la même façon. Les interactions à base de *blackboard* (Hayes-Roth, 1985) utilisent un phénomène similaire même si l'information n'est pas déposée directement dans l'environnement mais dans une entité intermédiaire. Finalement, les champs de potentiel (Khatib, 1986) issus de la robotique mobile, permettent d'attirer et/ou de repousser des entités grâce à des champs émis en champ libre et atténués par la distance entre l'émetteur et le récepteur.

Une vision synthétique de tous ces types d'interactions consiste à les considérer comme moyens pour demander ou fournir des services. Cette vision également reprise dans les Architectures Orientées Service (*Service Oriented Architecture* ou *SOA*), est de plus en plus utilisée de nos jours et est basée sur deux types d'entités : le producteur de service (ou fournisseur) d'une part et le consommateur (ou client) d'autre part. Plus d'informations sur les SOA peuvent être trouvées dans (Erl, 2004; Krafzig et al., 2005). Le modèle d'architecture ORCA reprend cette vision « service » par soucis de généricité.

Le modèle holonique d'ORCA retenu est présenté dans la section suivante.

4. Le modèle holonique d'ORCA retenu

Cette section présente le modèle (nommé Holo-Gen (Pach et al., 2012c)) répondant aux spécifications présentées dans la section précédente (e.g. récursivité, notion de rôle, de services). Il est décomposé en trois sous-parties présentant un modèle structurel, un modèle comportemental et un modèle d'interactions entre holons.

4.1. Le modèle structurel

Le modèle structurel est basé sur les principes holoniques de récursivité (holon décomposable en holons) et de généricité (un seul type d'holon générique constituant la « matrice mère » de tous les autres holons). Il se positionne sur la problématique du pilotage/contrôle de systèmes mais il est possible de l'envisager dans d'autres contextes tels que l'*Internet of things* (Atzori et al., 2010) ou la mécatronique. Le holon de base du modèle holonique proposé, permet de représenter tous les éléments d'un système qu'ils soient des entités décisionnelles ou des composants physiques. Le modèle étant holonique, il respecte un des principes de base d'un holon qui est d'intégrer à la fois une partie informationnelle et une partie physique. Le holon proposé s'inspire du modèle de holon de Christensen (Christensen, 1994) et l'étend en proposant les deux couches suivantes : la couche pilotage et la couche pilotée. La figure 3.5 illustre la constitution du holon proposé.

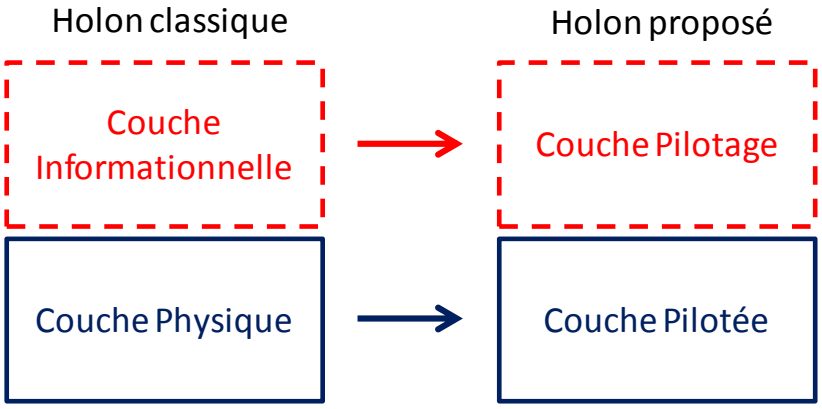


Figure 3.5– Constitution du holon classique vs holon proposé

Ainsi le holon proposé diffère de celui de Christensen car il décompose le holon en couches fonctionnelles (pilote / piloté) au lieu de se baser sur la nature des entités (informationnelle /

physique). Ainsi dans ce holon, la couche pilotage pilote la couche pilotée en lui envoyant des ordres/recommandations de pilotage. La couche pilotée lui remonte son état (figure 3.6). La couche pilotage peut contenir des traitements (décisionnels ou non), du stockage d'information et des protocoles de communication. Cette modélisation permet un continuum de représentation (pilote-piloté dans le domaine informationnel puis lorsque nécessaire physique) jusqu'au holon physique terminal, cas particulier d'une couche pilotée. La récursivité illustrée figure 3.6, permet de représenter l'intégralité des éléments du système, en partant des holons terminaux composés d'éléments physiques jusqu'au holon qui représente le système en entier ou inversement.

Afin de définir une instance de ORCA, on donnera la structure d'un holon, récursivement par une notation de type HP **π** HC (HP **pi**lote HC). Ainsi l'instance d'ORCA correspondant à la figure 3.6 est notée :

$(H_1 \, \boldsymbol{\pi} \, \{H_{1.1}, H_{1.2}\})$;

$(H_{1.1} \, \boldsymbol{\pi} \, \{H_{1.1.1}\})$; $(H_{1.1.1} \, \boldsymbol{\pi} \, \varphi_1)$;

$(H_{1.2} \, \boldsymbol{\pi} \, \{H_{1.2.1}, H_{1.2.2}\})$; $(H_{1.2.1} \, \boldsymbol{\pi} \, \varphi_2)$; $(H_{1.2.2} \, \boldsymbol{\pi} \, \varphi_3)$.

La figure 3.6 reprend cette instance d'ORCA de manière graphique. Les noms inscrits dans la couche pilotage des holons (ex : H_1, $H_{1.1}$) correspondent au holon complet (couches pilotage + pilotée).

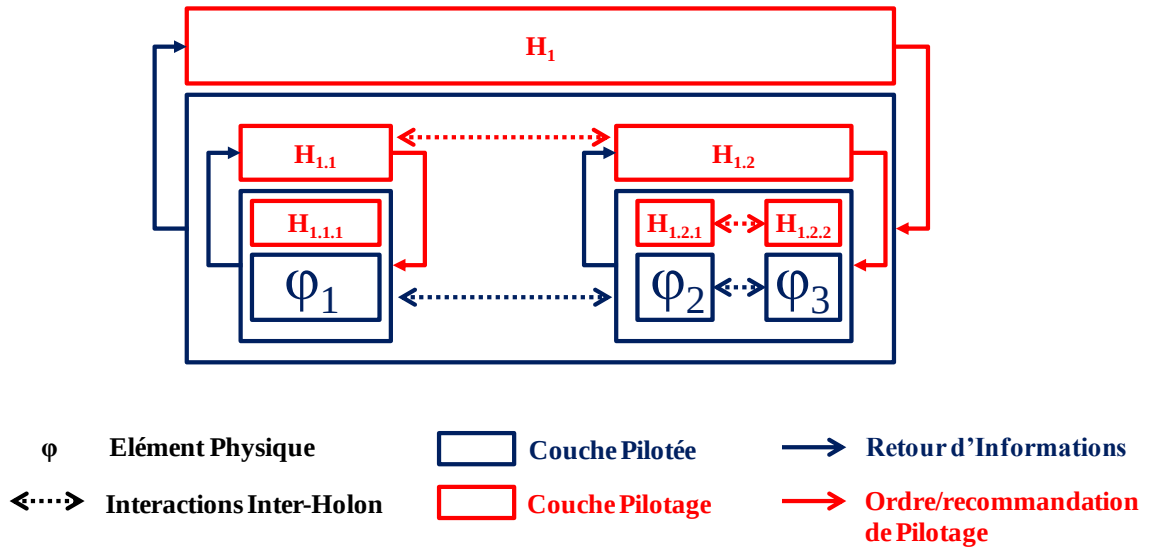


Figure 3.6– Illustration de la récursivité du modèle d'ORCA

Le modèle structurel permet de modéliser l'architecture ORCA via des holons représentant chaque entité du système. Ainsi l'exemple d'architecture ORCA présentée figure 3.3 peut être représentée via des holons comme sur la figure 3.7. Les éléments systèmes sont représentés par des quasi-holons (le terme quasi signifiant que ces holons ne présentent aucune autonomie) comprenant une partie physique associée au contrôle de bas niveau de cette partie physique. On définit l'exemple présenté figure 3.7 par :

(H_1 π {$H_{1.1}$, $H_{1.2}$}) ;

($H_{1.1}$ π {$H_{1.1.1}$, $H_{1.1.2}$, $H_{1.1.3}$}) ; ($H_{1.1.1}$ π $H_{1.1.1.1}$); ($H_{1.1.2}$ π $H_{1.1.2.1}$); ($H_{1.1.3}$ π $H_{1.1.3.1}$) ; ($H_{1.1.1.1}$ π φ_1) ; ($H_{1.1.2.1}$ π φ_2) ; ($H_{1.1.3.1}$ π φ_3) ;

($H_{1.2}$ π {$H_{1.2.1}$, $H_{1.2.2}$}) ; ($H_{1.2.1}$ π $H_{1.2.1.1}$) ; ($H_{1.2.2}$ π $H_{1.2.2.1}$) ; ($H_{1.2.1.1}$ π {φ_4, φ_5}) ; ($H_{1.2.2.1}$ π φ_6).

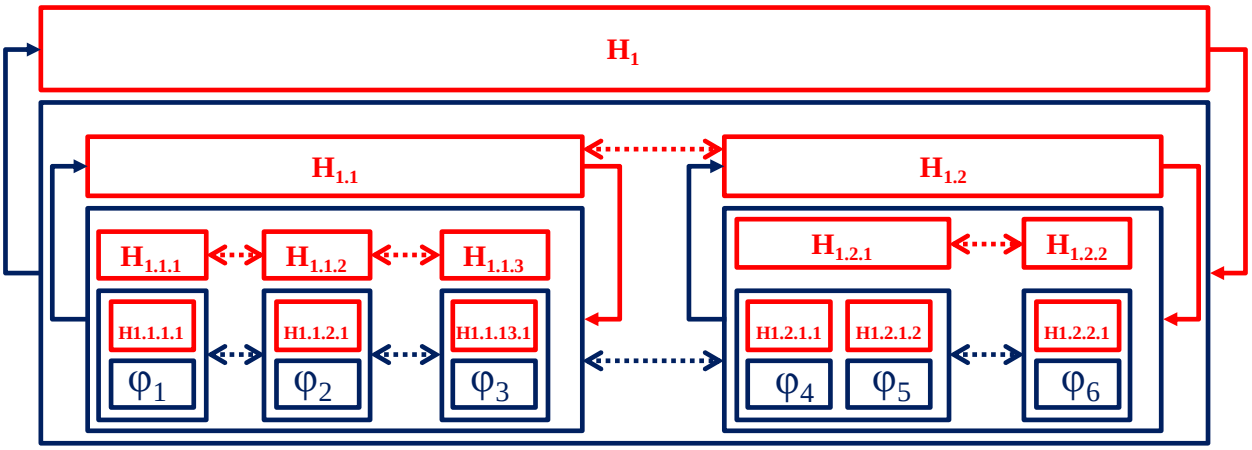


Figure 3.7– Exemple de représentation d'architecture ORCA

La partie suivante présente le modèle comportemental d'un holon HG.

4.2. Le modèle comportemental

Le modèle comportemental utilise la notion de rôle définie de la manière suivante par (Kristensen et Østerbye, 1996) :

> Le rôle d'un objet est un ensemble de propriétés qui sont importantes pour que cet objet soit capable de se comporter de la façon attendue par un ensemble d'autres objets.
>
> Kristensen et Osterbye, 1996

Ainsi le rôle va définir le ou les comportements suivi(s) par un objet (i.e. entité). Comme décrit dans (Kristensen et Østerbye, 1996) un objet peut changer de rôle dynamiquement. Les rôles utilisés ici correspondent aux rôles fonctionnels en opposition aux rôles dits « sociaux ». Un rôle social identifie un objet « par ce qu'il fait et ses relations avec les autres objets » alors qu'un rôle fonctionnel décrit « ce que fait concrètement un objet avec et pour un autre objet » (Darling et al., 2002).

Un rôle dans ce modèle ne comporte pas de notion de mission (i.e. groupe de buts collectifs ou individuels) à l'instar de JaCoMo (Sorici et al., 2012) ou de Janus (Gaud et al., 2008). Les buts sont ici intégrés dans le rôle lui-même. Ainsi par ses objectifs et ses règles, les rôles de ce modèle correspondent ici à ceux de la méthode Gaia est peuvent donc être vus comme une « description abstraite des responsabilités et des fonctions attendues d'un agent (Wooldridge et al., 2000).

Les rôles joués par les holons de ce modèle sont les fonctions « endossées » par le holon dans le système comme présenté dans (Pach et al., 2011a). Un rôle est joué dans un but précis et est défini par un ensemble de règles. Un rôle est défini à partir de descripteurs comme présenté par la formule suivante inspiré de (Adam et Mandiau, 2007).

Le comportement d'un holon est défini par les deux descripteurs suivants :

C_{Holon} = {Rôles, Gestion Rôles}	
Rôles	Ce sont les fonctions du Holon dans le système. Un rôle est joué dans un but précis et est défini par un ensemble de règles.
Gestion Rôles	C'est un rôle permanent permettant de gérer le passage d'un rôle à un autre pour le holon. Dans le cas particulier, où un seul rôle est associé (statiquement) au holon, la « gestion rôles » n'a pas lieu d'être.

Un rôle est défini par les trois descripteurs comme représenté ci-dessous.

Rôle = {Connaissance, Buts, Règles}	
Connaissance	Elle regroupe l'ensemble des données nécessaires pour pouvoir jouer le rôle. Les pré-requis à l'obtention du rôle, et à son activation sont inclus dans la connaissance.
Buts	Ce sont les objectifs à atteindre et les contraintes à satisfaire par le rôle. Ils correspondent aux obligations liées au rôle.
Règles	Elles sont constituées d'ensembles d'inférences élémentaires à suivre pour atteindre les buts du rôle.

Les deux exemples suivant présentent un modèle comportemental associé une ressource de production pouvant passer en veille dans le domaine manufacturier et à un service de soin dans le domaine hospitalier.

Exemple 1 : Ressource de production

Le premier exemple de comportement décrit celui d'une ressource qui peut adopter une stratégie d'économie d'énergie lorsque le système n'est pas très chargé ou une stratégie de performance lorsqu'on doit produire très vite. Les rôles de cette ressource peuvent être décrits de la façon suivante.

$$C_{RessourceProduction} = \{\ \text{Performance} - \text{Economie},\quad \text{Gestion Rôles}\ \}$$

Rôle = Performance	
Connaissance	Vitesse de fonctionnement maximum, Gamme de fabrication, Trajectoires, Ordonnancement Global.
Buts	Fournir des services aux produits à manufacturer le plus rapidement possible.
Règles	Respecter les impératifs de l'ordonnancement Global, Respecter les gammes de fabrications, Utiliser la vitesse maximum.

Rôle = Économie	
Connaissance	Vitesse de fonctionnement économe, Gamme de fabrication, Trajectoires.
Buts	Fournir des services aux produits en consommant le moins possible.
Règles	Respecter les gammes de fabrications, Utiliser la vitesse économe.

Rôle = Gestion Rôles	
Connaissance	Ordonnancement Global, Taux de charge de la ressource.
Buts	Choisir le meilleur rôle possible pour la ressource.
Règles	{**Si** le taux de charge prévu pour la ressource est faible, **Alors** jouer le rôle Économie, **Sinon** jouer le rôle Performance}.

Exemple 2 : Service de soin

Dans les hôpitaux les services de soin fonctionnent différemment en fonction de la situation et doivent par exemple réagir en cas de situation de tension ou de crise (plus de détails disponibles dans (Kadri et al., 2013)). Ainsi un exemple de comportement d'un service peut être représenté de la façon suivante via un holon disposant de deux rôles : fonctionnement normal et fonctionnement en tension et d'une gestion rôles.

$$C_{ServiceSoin} = \{ \text{Fct\_Normal} - \text{Fct\_Tension}, \quad \text{Gestion Rôles} \}$$

Rôle = Fct_Normal	
Connaissance	Planning Global du personnel, Planning Global des ressources, Agenda d'arrivée prévu des Patients, Procédures et protocoles de soin.
Buts	Fournir les services de soin nécessaires aux patients en respectant les plannings.
Règles	Respecter le planning global du personnel, Respecter le planning global des ressources, Respecter les procédures et protocoles de soin.

Rôle = Fct_Tension	
Connaissance	Procédures et protocoles de soin.
Buts	Fournir les services de soin nécessaires aux patients au plus vite.
Règles	Respecter les procédures et protocoles de soin.

Rôle = Gestion Rôles	
Connaissance	Planning Global du personnel, Planning Global des ressources, Agenda d'arrivée prévu des Patients, Situation du service.
Buts	Choisir le meilleur rôle possible pour le service.
Règles	{**Si** la situation du service est compatible avec les plannings, **Alors** jouer le rôle Fct_Normal, **Sinon** jouer le rôle Fct_Tension}.

La partie suivante présente le modèle d'interactions entre les holons du modèle.

4.3. Le modèle d'interactions

La section 3 a illustré l'intérêt de considérer les interactions sous forme de services. Selon ce point de vue, les holons interagissent entre eux en demandant ou fournissant des services. Suivant la théorie du système générale (Le Moigne, 1994) ces services peuvent être divisés selon trois catégories : Services d'Espace (S_E) correspondant à des déplacements, Services de Temps (S_T) correspondant à des immobilisations ou Services de Forme (S_F) correspondant à des transformations. Dans le domaine holonique deux types de services sont à différenciés, informationnels, physiques. Ainsi les trois catégories précédentes (Espace, Temps, Forme) peuvent être divisées en deux pour permettre la différenciation des Services Informationnels

(S_I) et Physiques (S_P). Dans un système les services peuvent donc être représentés de la manière suivante :

- S_{EI} = Service Espace Informationnel.
- S_{EP} = Service Espace Physique.
- S_{TI} = Service Temps Informationnel.
- S_{TP} = Service Temps Physique.
- S_{FI} = Service Forme Informationnel.
- S_{FP} = Service Forme Physique.

Chaque holon peut donc fournir ou demander une partie ou la totalité de ces services (lorsqu'ils existent) (figure 3.8).

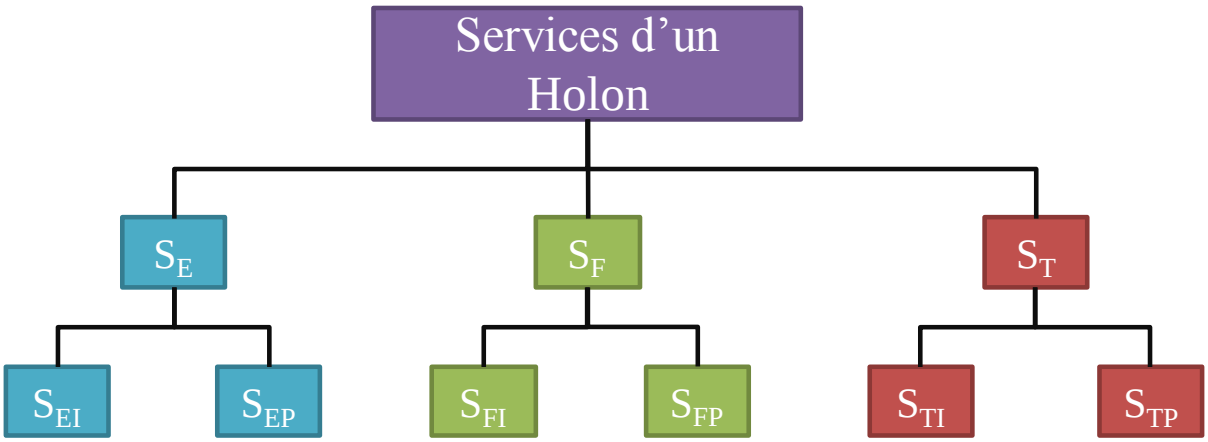


Figure 3.8– Services proposés ou fournis par un holon

5. Conclusion

Ce chapitre a permis de présenter l'architecture générique ORCA répondant aux spécifications définies dans les chapitres I et II. Cette architecture a été conçue en suivant la démarche présentée en début de ce chapitre.

Dans la première partie de ce chapitre, l'architecture ORCA qui est hybride et dynamique a été présentée. ORCA permet à ses entités d'évoluer selon deux modes de fonctionnement : un mode exécutant dans lequel l'entité suit le comportement optimal (pour le modèle considéré) fournit par l'optimiseur global et un mode autonome lui permettant de réagir aux évènements inattendus. Ainsi la myopie peut être contrôlée lorsque le système est dans un fonctionnement normal (sans évènement inattendu) tout en permettant aux entités de réagir en cas d'évènement inattendu.

La seconde partie de ce chapitre a permis de définir un modèle de représentation d'ORCA et de ses entités afin d'accompagner la mise en œuvre. Après l'étude des limites des modèles existants le choix de proposer un nouveau modèle holonique s'est imposé. Ce modèle a été

décrit et les modèles structurel, comportemental et d'interactions le composant ont été détaillés.

Le chapitre IV présente une application de l'architecture ORCA à l'ordonnancement de la production d'un système de production flexible (SFP ou *FMS* en anglais) sous l'appellation ORCA-FMS. Dans ce chapitre IV, seront définies les approches d'optimisation locale et globale, le mécanisme de basculement utilisé, ainsi que les modèles structurel, comportemental et d'interactions retenus pour l'application d'ORCA à l'ordonnancement de la production d'un SFP.

Chapitre IV : Application d'ORCA au pilotage de SFP – ORCA-FMS

1. Introduction

Ce chapitre présente une application de l'architecture de pilotage ORCA au pilotage (ici ordonnancement et conduite cf. chapitre I figure 1.3) d'un système flexible de production (SFP). Cette architecture est nommée ORCA-FMS (pour *Flexible Manufacturing System).* ORCA-FMS a pour objectif d'illustrer l'intérêt des architectures hybrides de type ORCA pour résoudre les problèmes d'ordonnancement et conduite dans les SFP tout en contrôlant la myopie des entités au sein de ces systèmes.

Comme indiqué dans la conclusion du chapitre III, l'application d'ORCA au pilotage d'un SFP, implique de déterminer un ensemble d'éléments. Ces éléments sont illustrés dans la figure 4.1.

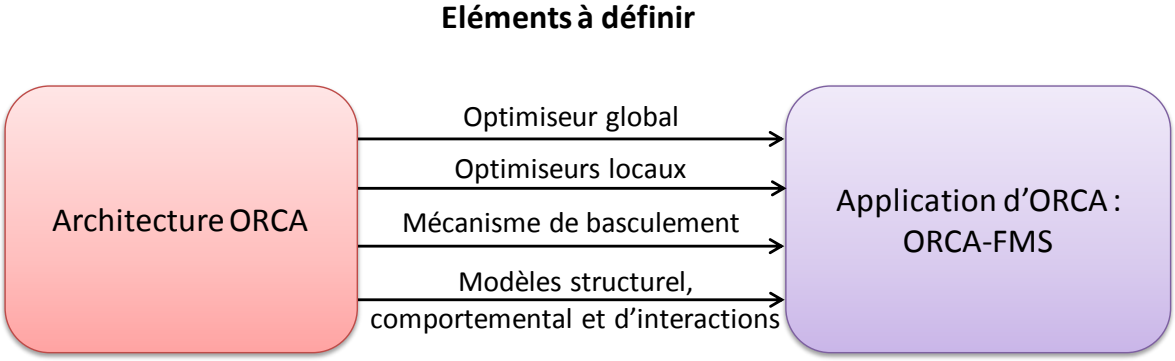


Figure 4.1– Éléments à définir lors de l'application d'ORCA

Ce chapitre présente tout d'abord le problème d'ordonnancement d'un SFP et le fonctionnement global de l'architecture ORCA-FMS (section 2). Puis il détermine les éléments présentés figure 4.1, à savoir la méthode d'optimisation globale (section 3), la méthode d'optimisation locale (section 4) et le mécanisme de basculement (section 5) utilisés. Finalement, ce chapitre présentera la modélisation d'ORCA-FMS via le modèle holonique proposé dans le chapitre III.

2. Présentation d'ORCA-FMS

Cette section présente le fonctionnement global de l'architecture ORCA-FMS qui réalise le pilotage (i.e. ordonnancement et conduite) d'un SFP. Le problème complexe d'ordonnancement d'un SFP est tout d'abord présenté.

2.1. Le problème de flexible Job-Shop

Dans un SFP, chaque opération d'un produit en cours de fabrication peut être réalisée sur plusieurs ressources, et chaque ressource peut proposer plusieurs opérations aux produits (Shivanand et al., 2006). Ce type de problème est considéré par la communauté de Recherche Opérationnelle comme un *Flexible Job-Shop Problem* (FJSP) avec des contraintes supplémentaires associées par exemple aux files d'attente des ressources et aux temps de transport. Il a été démontré que ce problème est NP-difficile (*NP-Hard)* notamment par (Mati et Xiaolan Xie, 2003). Il s'ensuit alors que le temps de résolution du problème est exponentiel à la taille du problème (taille exprimée par exemple par le nombre de ressources, le nombre de produits, etc.) et donc obtenir une solution optimale peut prendre énormément de temps. De plus, comme illustré en **annexe 7**, augmenter la capacité de calcul de l'ordinateur qui résout un problème complexe ne suffit pas pour réduire significativement le temps de résolution (plus de détails sur les problèmes NP-difficile sont données dans (Garey et Johnson, 1979)). C'est pourquoi ce problème est généralement résolu en relaxant des contraintes (comme la capacité des ressources qui est considérée infinie), en négligeant les temps de transport entre ressources, en ne prenant pas en compte les perturbations ou encore, la résolution est envisagée en cherchant une solution approchée plutôt que la solution optimale. Le travail de cette thèse s'inscrit dans un contexte où le modèle à résoudre se doit d'être le plus proche possible du système physique afin d'obtenir des résultats de référence fortement pertinents. C'est pourquoi le choix a été fait de considérer un maximum de contraintes malgré le risque d'accroître la complexité du modèle résultant (nous montrerons cependant que notre approche permet de limiter cet accroissement).

2.2. Fonctionnement général d'ORCA-FMS

Les fonctions supportées par ORCA-FMS correspondent aux fonctions du niveau pilotage 1 (ordonnancement et conduite) présentées figure 1.3. La figure 4.2 rappelle ces fonctions et positionne ORCA-FMS par rapport à un système d'information classique (dans les entreprises) comprenant un ERP et un MES*. MES* correspond à un MES (*Manufacturing Execution System)* pour lequel les fonctions ordonnancement et conduite sont « descendues » dans ORCA-FMS.

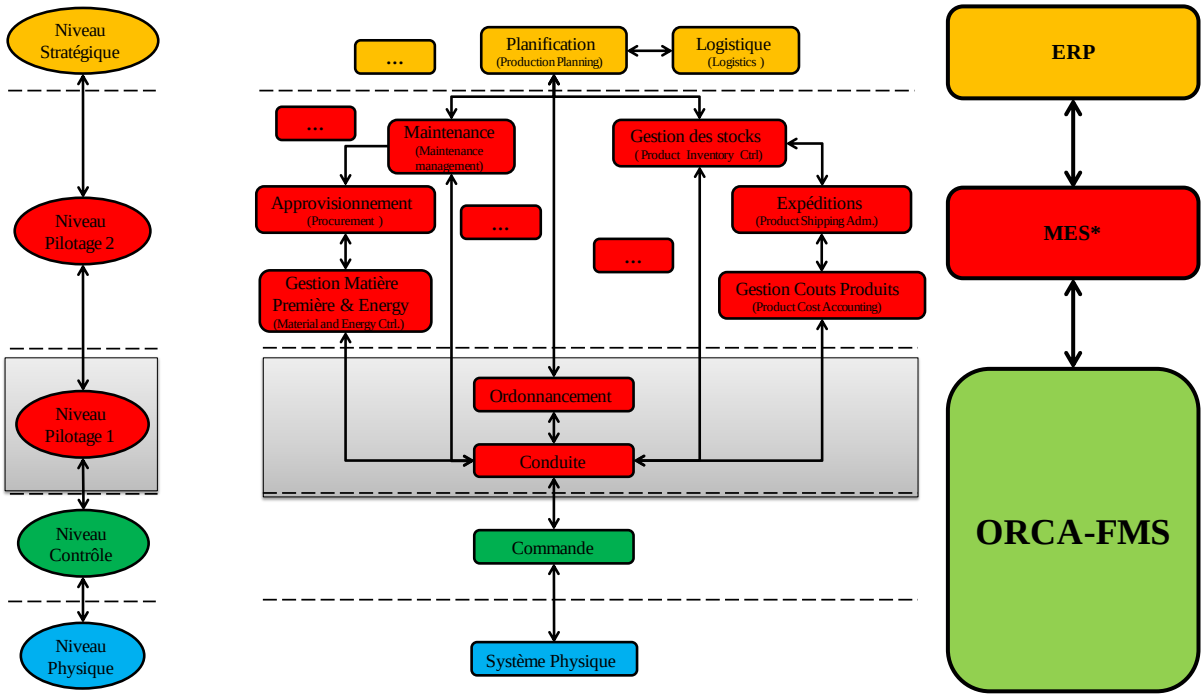


Figure 4.2– Fonctions et positionnement d'ORCA-FMS

Ainsi en mode exécutant, l'optimiseur global (OG) de la couche de contrôle global d'ORCA-FMS réalise l'allocation des produits aux ressources et fournit des données sur cette allocation aux optimiseurs locaux (OL) de la couche de contrôle local. Ces données peuvent être par exemple une séquence de ressources à utiliser par chaque produit à fabriquer, c'est le choix fait dans cette thèse. À chaque début de lot de production, les données sur l'allocation sont fournies aux OL et ne sont recalculées que pour le lot suivant. Dans ORCA-FMS, chaque OL est associé à un produit. Il exécute la séquence fournie par l'OG et gère le routage en déterminant dynamiquement le chemin le plus rapide pour arriver à la prochaine ressource prévue dans la séquence. Si une ressource ou une partie du système de convoyage n'est plus disponible (c'est-à-dire si la séquence n'est plus réalisable compte tenu de la situation actuelle), l'OL associé au produit gêné par l'évènement inattendu déclenche le basculement dans le mode autonome pour ce produit à fabriquer. Dans ce mode, l'OL réalise à la fois le routage, mais aussi l'allocation du produit qui lui est associé aux ressources et ne respecte donc plus la séquence fournie en début de production par l'OG.

Dans cette application d'ORCA, l'OG est mis en œuvre au travers d'un ILP (*Integer Linear Programming*). Il est présenté et justifié section 3. L'OL quant à lui est implémenté en utilisant une approche par champs de Potentiel. Celle-ci est présentée et justifiée section 4. Deux types d'entités existent dans ORCA-FMS. Premièrement, l'ensemble composé d'un OL, d'un produit physique et d'une ressource de transport forme une Entité « Produit ». Puis l'ensemble composé d'un OL, d'un stock de composants, d'une file d'attente et d'un robot forme une Entité « Ressource ». Ces concepts sont détaillés dans la section 6 de ce chapitre. La figure 4.3 illustre (pour le mode exécutant) ORCA et l'architecture ORCA-FMS associée utilisée dans cette thèse.

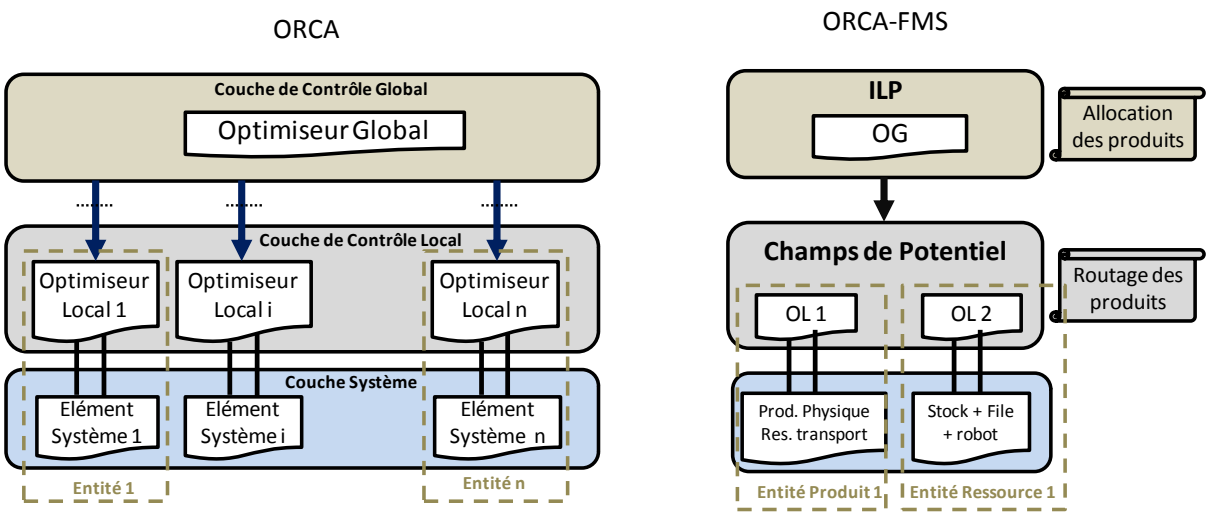


Figure 4.3– Mise en œuvre des couches d'ORCA dans ORCA-FMS

La section 3 présente le modèle linéaire associé à la couche de contrôle global d'ORCA-FMS. Dans un souci de cohérence vis-à-vis des usages en Recherche Opérationnelle, le terme « machine » est utilisé pour décrire une « ressource » et le terme « job » remplace le « produit » utilisé dans le reste de la thèse.

3. Présentation de la couche de contrôle global : l'ILP

Un modèle linéaire a été utilisé car cela permet d'obtenir une solution optimale (compte tenu du modèle considéré) pour le problème à résoudre. Ceci permet d'avoir une référence, et, dans le futur, de pouvoir positionner des méthodes approchées par rapport à cette référence. Le modèle utilisé contient uniquement des variables entières et est donc qualifié *d'Integer Linear Programming* (ILP). La figure 4.4 positionne ce modèle de programmation mathématique, aux côtés des modèles permettant l'utilisation de variables réelles (*Mixed-Integer Programming*), des modèles n'utilisant que des variables binaires (*Binary Integer Programming*) ou des modèles non linéaires (*Integer NonLinear Programming*).

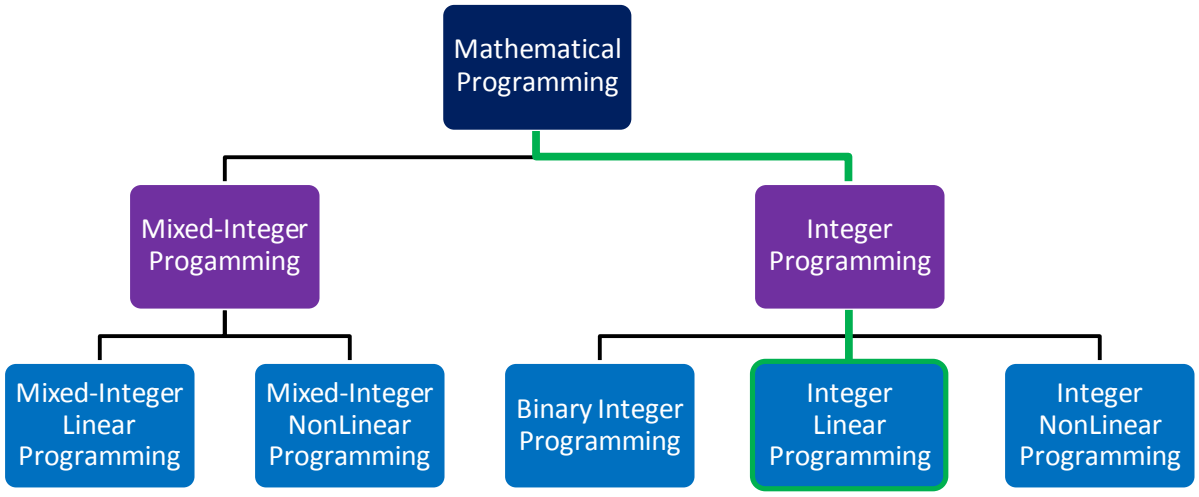


Figure 4.4– Positionnement du modèle utilisé

Comme indiqué auparavant, le modèle linéaire d'ORCA-FMS se doit d'être le plus proche possible de la réalité, grâce notamment à la prise en compte de contraintes relativement originales mais pertinentes dans un contexte de type FMS :

- Des temps de transport non négligés entre les machines,
- Un nombre fini de ressources de transport (ex. limitation à 10 ressources de transport véhiculant les produits dans le système),
- Une capacité finie pour ces ressources de transport (ex. 1 produit par ressource de transport).

Ces contraintes ne sont généralement pas considérées et prises en compte dans la littérature (ou seulement en partie). Par exemple, certains travaux de la littérature limitent le nombre de moyens de transport à 1 ressource (Bilge et Ulusoy, 1995; Liu et MacCarthy, 1997), d'autres ne considèrent pas le temps de transport entre les machines (Caumond et al., 2009; Fattahi et al., 2007), ou encore ne considèrent pas ou négligent la capacité du système de transport (Anwar et Nagi, 1998; Gomes et al., 2005). C'est pourquoi un ILP a été spécifiquement développé pour cette thèse. Le modèle présenté ci-dessous calcule l'ordonnancement optimal (pour le modèle considéré) de la production, permettant d'obtenir le Cmax le plus faible possible. Cependant, une fois cet ordonnancement calculé, l'ILP ne fournit que la séquence des opérations (sans date) aux optimiseurs locaux. Plus de détails sur ce choix sont fournis dans la partie 5 de ce chapitre.

Lors de la réalisation de cet ILP, il a également été supposé que les machines sont à capacité finie. Un certain nombre d'hypothèses ont toutefois été posées pour limiter le contexte de notre étude :

- Les stocks de matières premières ne sont pas étudiés (considérés infinis),
- Le temps opératoire est indépendant de la machine sur laquelle est réalisée l'opération,
- La production est considérée par lot (production non continue),
- Les ressources de transport non utilisées sont stockées dans une zone proche de la machine de chargement, avec une capacité infinie.

Dans cette application d'ORCA, la fonction objectif utilisée est le Cmax principalement par souci de cohérence avec le mécanisme de fonctionnement de la couche locale d'ORCA-FMS. En effet, cette couche locale, gérée par une approche de champs de potentiel, est purement réactive, sans projection vers le futur. Elle a pour objectif de faire le mieux possible à l'instant t. Ainsi mesurer sa performance est plus aisé via le Cmax. De plus, les performances de la couche locale sont amenées à être comparées avec celles de la couche globale (i.e. ILP) et la couche globale doit donner des recommandations à la couche locale. Ainsi pour faciliter les interactions entre couches d'ORCA-FMS (i.e. envoi des recommandations notamment), le même indicateur de performance a été utilisé pour les deux couches d'ORCA-FMS, le Cmax. Ainsi, une production à flux tiré, basée sur des fonctions objectif liées aux retards ou avances (*tardiness*, *earliness*) ne peut en l'état être gérée via cette application d'ORCA et nécessiterait d'autres développements, notamment au niveau de la couche locale et du modèle ILP.

Le modèle linéaire utilisé est disponible en **annexe 8**. Ces paramètres, variables et contraintes sont repris et détaillés ci-dessous.

Notations pour les paramètres

J	ensemble de jobs, J={1,2,...,j,…|J|}
M	ensembles de machines, M={1,2,…,i,...|M|}
O_j	ensemble d'opérations du job j, O_j={1,2,...,h,…$\|O_j\|$}, $j \in J$
$PT(j,h)$	Temps de réalisation de l'opération h du job j
$TT(i,x)$	Valeur du temps de transport entre la machine i et la machine x
$a(i,j,h)$	$\begin{cases} 1 \text{ si l'opération h du job j peut être réalisée sur la machine i} \\ 0 \text{ sinon} \end{cases}$
BN	Un grand nombre
MJ	Nombre de jobs simultanés dans le système

Notations pour les variables

$t(j,h)$	Temps de début de l'opération h du job j
$Tm(i,l)$	Temps de début de l'opération réalisée avec la priorité l sur la machine i
$Tr(j,h,i,x)$	$\begin{cases} 1 \text{ s'il y a un temps de transport entre machines i et x après l'opération h du job j} \\ 0 \text{ sinon} \end{cases}$
$y(i,j,h)$	$\begin{cases} 1 \text{ si l'opération h du job j est réalisée sur la machine i} \\ 0 \text{ sinon} \end{cases}$
$X(i,j,h,l)$	$\begin{cases} 1 \text{ si l'opération h du job j est réalisée sur la machine i à la priorité l} \\ 0 \text{ sinon} \end{cases}$
$b(j,h,k,m)$	$\begin{cases} 1 \text{ si l'opération h du job j est réalisée avant l'opération m du job k} \\ 0 \text{ sinon} \end{cases}$
$z(j,k)$	$\begin{cases} 1 \text{ si le job j et le job k sont en même temps dans le système} \\ 0 \text{ sinon} \end{cases}$

Fonction Objectif

Minimize (Cmax)

Détail des Contraintes

Contrainte de calcul du Cmax : définit le Cmax comme au moins égal au temps après lequel la dernière opération a été réalisée dans la cellule (1).

$$Cmax \geq t(j,h) + PT(j,h) \qquad (1)$$

Contraintes de précédence : La première contrainte ci-dessous (2) assure qu'une opération d'un job ne peut commencer avant que l'opération précédente de celui-ci ne soit terminée (plus éventuellement le temps de transport). La seconde contrainte assure qu'une opération réalisée sur une machine ne peut pas commencer avant que l'opération précédente réalisée sur cette machine ne soit terminée.

$$t(j,h) + PT(j,h) + \sum_{i,x} Tr(j,h,i,x) * TT(i,x) \leq t(j,h_{+1}) \quad (2)$$

$$Tm(i,l_{+1}) \geq Tm(i,l) + \sum_{j,h} PT(j,h) * X(i,j,h,l) \quad (3)$$

Contrainte de transport : Si deux opérations successives d'un même job sont effectuées sur différentes machines, le temps de transport entre ces machines est considéré (4).

$$Tr(j,h,i,x) = \begin{cases} 1 \; if \; y(i,j,h) + y(x,j,h_{+1}) = 2 \quad (\forall x \neq i) \\ 0 \text{ sinon} \end{cases} \quad (4)$$

Contraintes d'allocation : Ces contraintes assurent que les opérations ne sont allouées qu'aux machines qui peuvent les réaliser (5), qu'une seule opération ne peut être réalisée au même moment sur une même machine (6) et que chaque opération n'est allouée qu'à une seule machine (7) et (8).

$$y(i,j,h) \leq a(i,j,h) \quad (5)$$

$$\sum_{j,h} X(i,j,h,l) \leq 1 \quad (6)$$

$$\sum_{i} y(i,j,h) = 1 \quad (7)$$

$$\sum_{i} X(i,j,h,l) = 1 \quad (8)$$

Contraintes de liaisons : Ces contraintes font la liaison entre les variables d'association début de l'opération/job et début de l'opération/machine (9) et (10) ; et le lien entre l'allocation d'opérations aux machines et l'ordre de réalisation des opérations sur les machines (11).

$$Tm(i,l) \leq t(j,h) + \big(1 - X(i,j,h,l)\big) * BN \quad (9)$$

$$Tm(i,l) + \big(1 - X(i,j,h,l)\big) * BN \geq t(j,h) \quad (10)$$

$$\sum_{l} X(i,j,h,l) = y(i,j,h) \quad (11)$$

Contraintes de limitation des jobs : Ces contraintes limitent le nombre de jobs simultanés pour tenir compte du nombre de ressources de transport disponibles MJ. O_{0j} définit la première opération du job j et O_{uj} définit la dernière. La première contrainte (12) définit si une opération est faite avant une autre. Les autres contraintes ((13), (14), (15), (16)) permettent de calculer le nombre de jobs simultanés.

$$t(j,h) + BN * b(j,h,k,m) \leq t(k,m) + BN \quad (12)$$

$$\sum_j z(j,k) \leq MJ - 1 \quad (\forall j \neq k) \quad (13)$$

$$z(j,k) \geq b(k,0,j,u) + b(j,0,k,0) - 1 \quad (14)$$

$$z(j,k) \leq 1 - b(k,0,j,0) + b(j,u,k,u) \quad (15)$$

$$z(j,k) \geq b(k,0,j,0) + b(j,u,k,u) - 1 \quad (16)$$

La partie suivante présente les champs de potentiel utilisé dans la couche de contrôle local d'ORCA-FMS

4. Présentation de la couche de contrôle local : les Champs de Potentiel

Cette section présente l'approche par champs de potentiel correspond à la couche de contrôle local d'ORCA-FMS.

4.1. Introduction aux Champs de Potentiel

Le concept de Champs de Potentiel (CP) est traditionnellement utilisé pour influencer le comportement réactif d'entités mobiles dans un environnement incertain/changeant. Ce concept correspond à un contrôle implicite environnemental au regard de *l'Open Control (*cf. section 3.3 chapitre III). Les premières applications des champs de potentiel sont liées à la navigation de robots mobiles (Khatib 1986 ; Samsudin, Ahmad, et Mashohor 2011) où des champs attractifs attirent des robots vers leur destination alors que des champs répulsifs permettent aux robots d'éviter des obstacles. Les champs de potentiel ont déjà été appliqués aux SFP au sein de l'équipe PSI du laboratoire TEMPO avec des entités mobiles dotées de capacité décisionnelle comme des « produits actifs » (Pach et al., 2012a). Dans cette première application, des champs attractifs permettent de réaliser simultanément et dynamiquement l'allocation des produits aux ressources et le routage des produits vers ces ressources. Cette étude a permis de mettre en valeur la simplicité et l'efficacité de l'approche par champs de potentiel. Les entités dans une approche par champs de potentiel n'ont pas besoin d'une connaissance complète du système. L'allocation et le routage peuvent être gérés uniquement avec les champs sans besoin d'un ordonnancement calculé a priori. Cette approche est donc adaptée au contrôle réactif (sans prévision à court terme ou moyen terme) dans un contexte incertain et convient donc pour la couche de contrôle local d'ORCA-FMS. Bien entendu d'autres approches comme la stigmergie (Sallez, Berger, et Trentesaux 2009) ou encore les *Delegate MAS* (Rutten et Valckenaers, 2013) peuvent être utilisées pour cette couche locale.

La figure 4.5 illustre le concept de CP dans l'ordonnancement de la production. Trois ressources (R_1, R_2 et R_3) sont disponibles et émettent des CPs pour les services (correspondant aux « opérations » dans le modèle linéaire) qu'elles peuvent fournir. Le CP émis est propagé le long du système de convoyage (en sens inverse des flux de produits physiques en cours de

fabrication). Le produit P1 entre dans le système avec sa liste de services à obtenir (S_1, S_2 et S_3). Il commence donc par choisir la ressource émettant le champ le plus fort pour le premier service S_1 qu'il doit obtenir. Dans le cas présent c'est donc R_1 qui est choisie. Dans cet exemple le produit n'émet pas de champs de Potentiel. Cet exemple illustre l'orientation « services » des champs de potentiel.

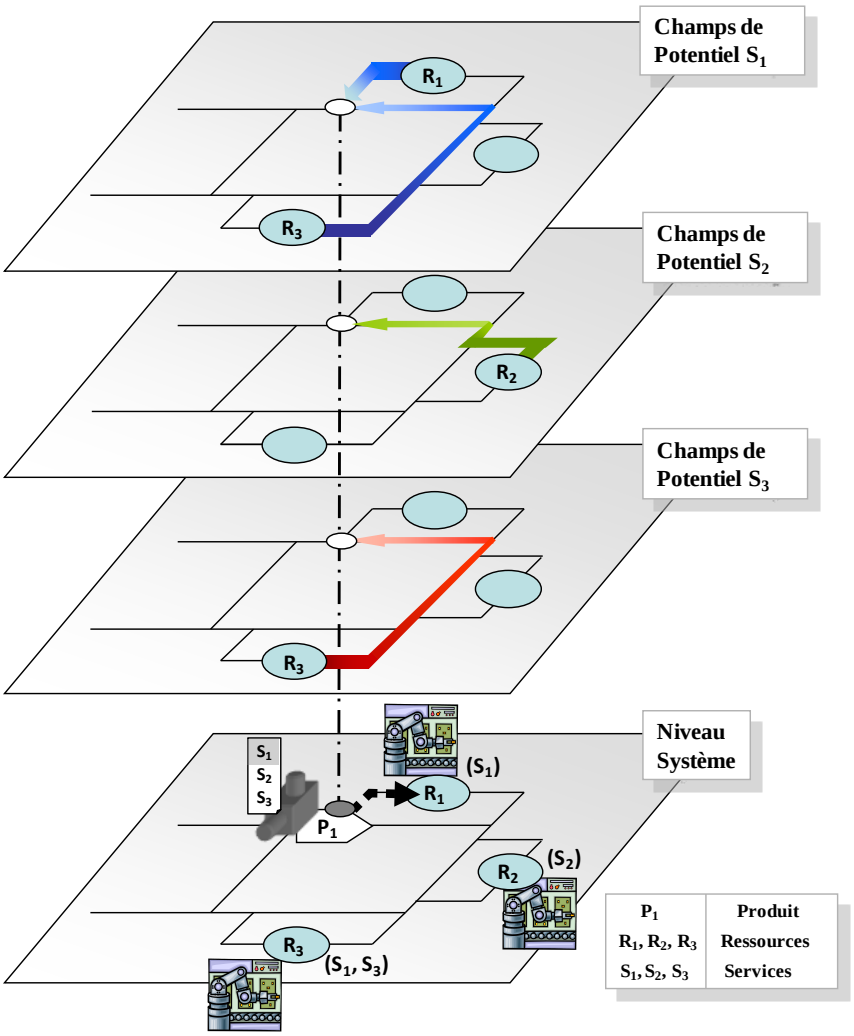


Figure 4.5– Illustration des champs de potentiel en production

4.2. Modèle de champs de potentiel

Le modèle de champs de potentiel (CP) précédent a été étendu aux ressources afin qu'elles soient également capables de prendre des décisions et ainsi être « actives ». Dans ce modèle étendu notamment présenté par (Pach et al., 2013b), les ressources intègrent un mécanisme leur permettant de basculer entre différents états pour, par exemple, réduire leur usure, limiter la consommation énergétique du système ou avoir un système efficient lorsque le système est peu chargé. L'idée est que lorsqu'aucun produit n'est sur le point d'utiliser une ressource, celle-ci bascule dans un mode de veille. Si un produit s'approche de cette ressource pour l'utiliser, elle peut se réveiller et se tenir ainsi prête à l'accueillir pour réaliser une opération nécessaire. Cette approche nécessite que les ressources aient de l'information sur la proximité

des produits qui les entourent (notamment la ressource qui a été choisie par un produit). Ces informations de proximité sont perçues grâce à l'émission de CP par les produits. Comme pour les champs de potentiel émis par les ressources, ces champs sont affaiblis par la distance séparant le produit de la ressource (produit proche : « CP produit » fort ; produit loin : « CP produit » faible). Le fonctionnement global du modèle de champs de potentiel est illustré par la figure 4.6.

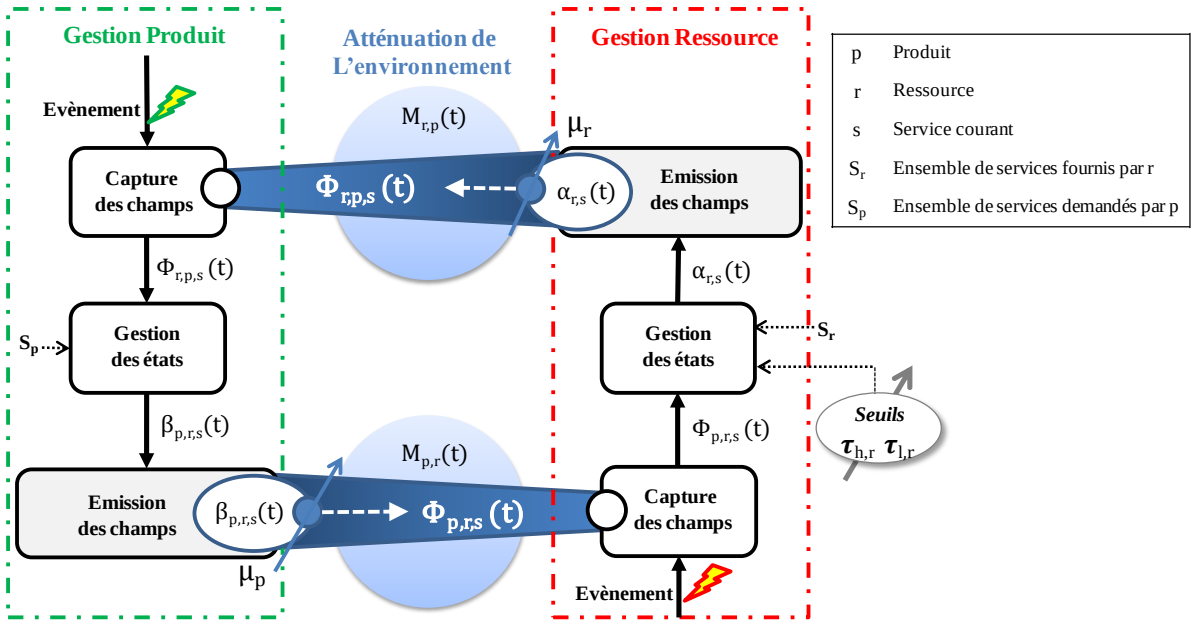


Figure 4.6– Fonctionnement du modèle de champs de potentiel

Dans ce modèle, deux cycles de gestion apparaissent. Le cycle de gestion du produit (présenté section 4.3) permet au produit d'émettre des champs de potentiel en fonction des champs ressources captés. Le cycle de gestion de la ressource (présenté section 4.4) permet aux ressources d'émettre des champs de potentiel attractifs en fonction des champs produits captés.

4.3. Cycle de gestion du Produit

Le cycle de gestion du produit est divisé en trois phases : la capture des champs, la gestion des états et l'émission des champs. Ce cycle est déclenché sur évènement. Cet évènement dépend du cas d'application et le cycle de gestion du produit doit être déclenché à chaque fois qu'une décision doit être prise (par exemple quand un chemin alternatif est possible, toutes les 5 secondes ou encore quand les données captées changent). La première phase consiste à capturer les champs de potentiel $\Phi_{r,p,s}(t)$ émis par chaque ressource r pour le service s et capté par le produit p. Ces champs sont filtrés pour conserver ceux qui concernent le service courant (à obtenir) du produit. Une fois que les données locales du produit ont été mises à jour avec les nouvelles valeurs des champs, il entre dans la phase de gestion des états qui suit le graphe d'état de la figure 4.7.

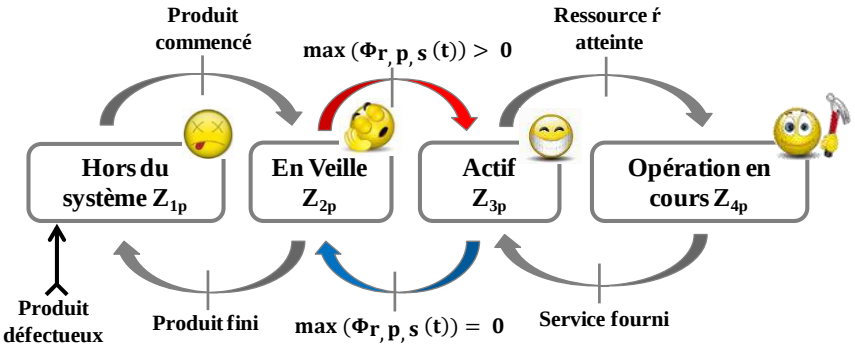


Figure 4.7– Graphe de gestion des états du produit

Quatre états sont distingués dans ce graphe :

- Hors du système : Cet état représente un produit dont le produit physique n'est pas encore commencé (il n'a pas obtenu sa liste de services) ou déjà terminé (fini ou défectueux).
- En veille : Cet état est atteint si un produit ne peut pas trouver de champs de potentiel provenant de ressources pour le service qu'il cherche à obtenir du fait d'une panne ressource par exemple. Il reste dans ce mode jusqu'à ce qu'un champ soit détecté.
- Actif : Cet état est atteint si au moins un champ de potentiel est perçu pour le service à obtenir. Dans cet état, le produit gère son allocation et son routage en temps réel et se déplace vers la ressource choisie.
- Opération en cours : Le produit bascule dans cet état une fois qu'il atteint la ressource choisie. Le produit demande la réalisation de son service courant et attend qu'il lui soit fourni. Ensuite, le produit retourne dans l'état actif et continue jusqu'à ce que sa liste de service à obtenir soit vide auquel cas il sort du système.

Si plusieurs ressources émettent des champs de potentiel $\Phi_{r,p,s}(t)$ pour le service courant à obtenir, le produit en état actif capte tous ces champs et choisit la ressource émettant le plus fort (c'est-à-dire le plus attractif). À un instant t, le produit p choisit la ressource $\acute{r}_{p,s}(t)$ selon la formule suivante :

$$\acute{r}_{p,s}(t) = arg\,max_r \left(\Phi_{r,p,s}(t)\right) \qquad (17)$$

Une fois la ressource choisie, le produit construit une intention $\beta_{p,r,s}(t)$, en direction des ressources, selon la formule suivante :

$$\beta_{p,r,s}(t) = Z_{i,p}(t) * C_{p,r,s}(t) \qquad (18)$$

Où $Z_{i,p}(t)$ est un coefficient qui dépend de l'état courant du produit (Figure 4.7) et $C_{p,r,s}(t)$ vaut 1 si la ressource r est la ressource choisie par le produit p pour le service s à l'instant t ; 0 sinon. L'ensemble des champs de potentiel $\Phi_{p,r,s}(t)$ construit plus tard sur la base de cette intention ne sera « ressenti » que par la ressource choisie $\acute{r}_{p,s}(t)$. Une fois cette intention calculée, le produit entre dans la phase d'émission de champs. Dans cette phase, les intentions

précédemment construites sont amplifiées, propagées et atténuées par l'environnement. Les CPs résultants $\Phi_{p,r,s}(t)$ émis par le produit *p* vers la ressource *r* pour le service *s* à l'instant *t* sont calculés par la formule suivante :

$$\Phi_{p,r,s}(t) = max\ \left(\beta_{p,r,s}(t) * \mu_p - M_{p,r}(t),\ 0\right) \qquad (19)$$

Où μ_p représente l'amplitude qui détermine la portée d'émission des CPs, $M_{p,r}(t)$ représente l'atténuation des CPs par l'environnement, et la fonction max () assure que les CPs ne sont jamais négatifs puisque dans ce modèle la fonction $M_{p,r}(t)$ est choisie comme linéaire. La figure 4.8 illustre l'émission des $\Phi_{p,r,s}(t)$. Dans cette figure, deux produits requièrent le service S_1 qui est fourni par R_1. Ils émettent donc des champs en direction de R_1. Dans cet exemple, seul le champ de P_2 atteint R_1.

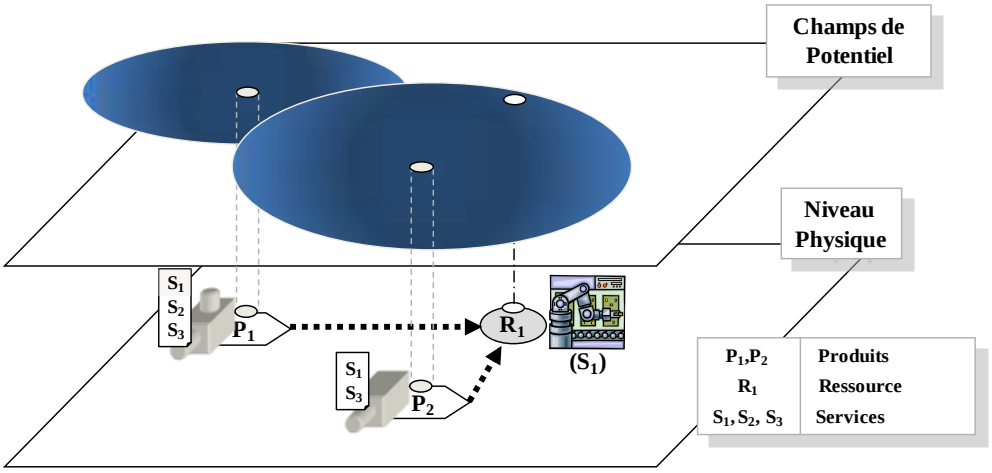


Figure 4.8– Émission de champs par les produits

4.4. Cycle de Gestion de la Ressource

Le cycle de gestion de la ressource est similaire à celui du produit. Il est aussi divisé en trois phases : la capture des champs, la gestion des états et l'émission des champs. Il est aussi déclenché sur évènement. La ressource capte tout d'abord les CPs $\Phi_{p,r,s}(t)$ émis par les produits qui se dirigent, en vue de l'obtention d'un service, vers elle. En fonction des valeurs de ces CPs, les états de la ressource sont gérés comme décrit dans la figure 4.9.

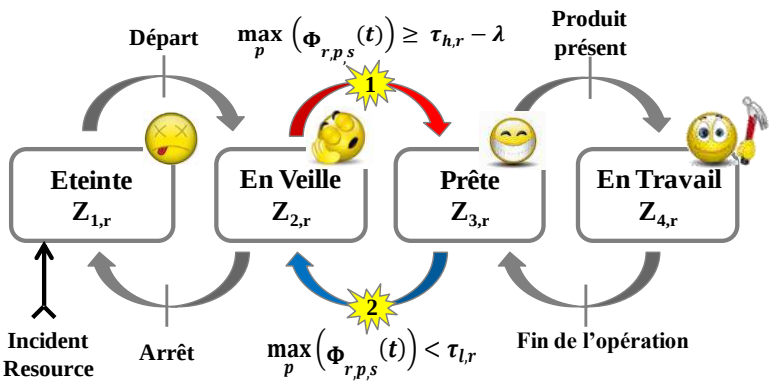


Figure 4.9– Graphe de gestion des états de la ressource

Quatre états sont distingués dans ce graphe :

- Éteinte : la ressource n'est pas disponible (en maintenance, en panne, pas démarrée…)
- En Veille : la ressource est disponible mais en veille.
- Prête : la ressource est disponible et potentiellement opérationnelle.
- En travail : la ressource est en train de réaliser une opération sur un produit.

Les ressources peuvent passer d'un état vers un autre selon des transitions présentées figure 4.9. Le modèle a été conçu pour favoriser l'allocation des produits aux ressources déjà disponibles ou prêtes et ainsi conserver les autres ressources dans l'état de veille (pour conserver un système efficient autant que possible). Une ressource va passer dans l'état « prête » uniquement si elle capte des CPs provenant de produits qui arrivent vers elle. Le basculement entre les états « prête » et « en veille » s'effectue quand les CPs captés dépassent un seuil prédéterminé $\boldsymbol{\tau}_{h,r}$. Le seuil permet de contrôler la sensibilité du basculement. Un temps de setup de la ressource peut être considéré en plus de ce seuil grâce à l'offset λ, dans le cas où un temps de préparation de la ressource r pour fournir le service *s* demandé par le produit *p* est nécessaire. La condition de basculement d'état est donnée par la formule suivante :

$$\max_{p} \left(\Phi_{p,r,s}(t)\right) \geq \tau_{h,r} - \lambda \qquad (20)$$

À l'inverse une ressource bascule « en veille » si elle ne capte aucun CP $\Phi_{p,r,s}$, ne dépassant le seuil bas $\boldsymbol{\tau}_{l,r}$ comme donné dans la formule suivante :

$$\max_{p} \left(\Phi_{r,p,s}(t)\right) < \tau_{l,r} \qquad (21)$$

En fonction de l'état dans laquelle la ressource se trouve, elle construit son attractivité $\alpha_{r,s}(t)$ qui définit la part du champ de potentiel qui ne dépend que de la ressource, suivant la formule suivante :

$$\alpha_{r,s}(t) = Z_{i,r}\,(t) * A_{r,s}\,(t) \qquad (22)$$

Où $Z_{i,r}$ (t) est un coefficient qui module l'attractivité en fonction de l'état courant de la ressource (figure 4.9) et $A_{r,s}\,(t)$ qui vaut 1 si la ressource *r* peut fournir le service *s* à l'instant *t* ; 0 sinon. Avec cette formule, la ressource construit une attractivité initiale, qui est ensuite amplifiée, propagée et affaiblie selon l'environnement. L'ensemble des CPs $\Phi_{r,p,s}(t)$ émis par la ressource *r* pour un service *s*, et ressenti par un produit *p* à l'instant *t*, est construit selon la formule suivante :

$$\Phi_{r,p,s}(t) = max\,\left(\alpha_{r,s}(t) * \mu_r - M_{r,p}(t), 0\right) \qquad (23)$$

De la même façon que pour les champs émis par les produits, μ_r représente l'amplitude qui détermine la portée d'émission des CPs, $M_{r,p}(t)$ représente l'atténuation des CPs par l'environnement, et la fonction max () assure que les CPs ne soient jamais négatifs puisque dans ce modèle la fonction $M_{r,p}(t)$ est choisie comme linéaire.

Les deux parties précédentes (3 et 4) ont permis de présenter les deux approches retenues pour les couches de contrôle global et local d'ORCA-FMS. Le pilotage des entités va être réalisé par l'une ou l'autre de ces couches en fonction du mode de fonctionnement (exécutant ou autonome) de chaque entité d'ORCA-FMS. Ceci va permettre soit de contrôler la myopie (en mode exécutant) et d'assurer de bonnes performances, soit de réagir aux évènements inattendus (mode autonome) en acceptant des décisions locales. La partie suivante présente en détails le mécanisme de basculement entre chacun des modes de fonctionnement.

5. Le Mécanisme de basculement d'ORCA-FMS

L'architecture ORCA est basée sur un basculement entre les modes exécutant et autonome pour réagir aux imprévus. Le chapitre II a permis de présenter et classer différents mécanismes de basculement. Le chapitre II a également mis en évidence l'intérêt des architectures de sous-classe II-DHe. Dans cette application d'ORCA, nous avons choisi un mécanisme de type $Y_{ORCA\text{-}FMS}$ = { DP, HE, NB | *} comme présenté figure 4.10.

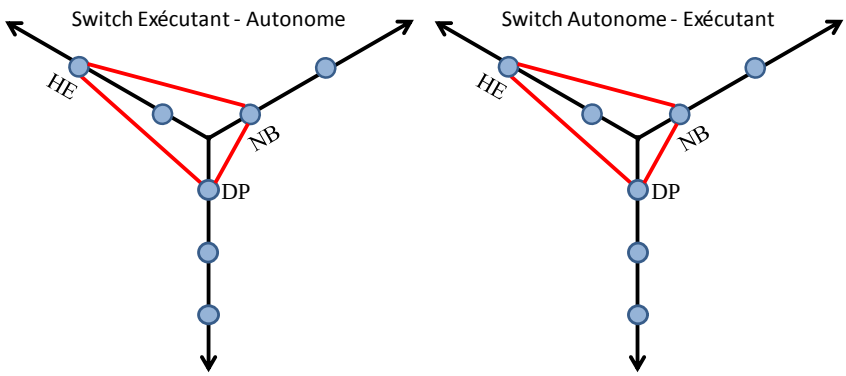


Figure 4.10– Classification du mécanisme de basculement d'ORCA-FMS

5.1. Détails sur le mécanisme de basculement

Comme indiqué précédemment, le mécanisme de basculement permet aux entités « produit » d'ORCA-FMS de basculer du mode de fonctionnement exécutant au mode autonome. Lorsqu'un produit physique est lancé dans le système, il est supposé lié à une entité « produit » jusqu'à ce qu'il soit terminé. L'entité gère l'exécution de la séquence des opérations définies par l'ILP ainsi que le routage dynamique du produit. Lorsque le produit physique est terminé, il est déchargé (i.e., sort du système) et l'entité produit associée est libérée afin d'être utilisé pour prendre en charge la réalisation d'un nouveau produit physique.

Si un évènement inattendu survient pendant la réalisation du produit physique, il y a basculement dans l'état autonome. Ce mécanisme de basculement est détaillé dans les figures 4.11a et 4.11b. La figure 4.11a présente le mécanisme de basculement intégré dans l'entité produit et la figure 4.11b positionne les étapes du basculement dans un réseau de Petri représentant la réalisation d'un produit.

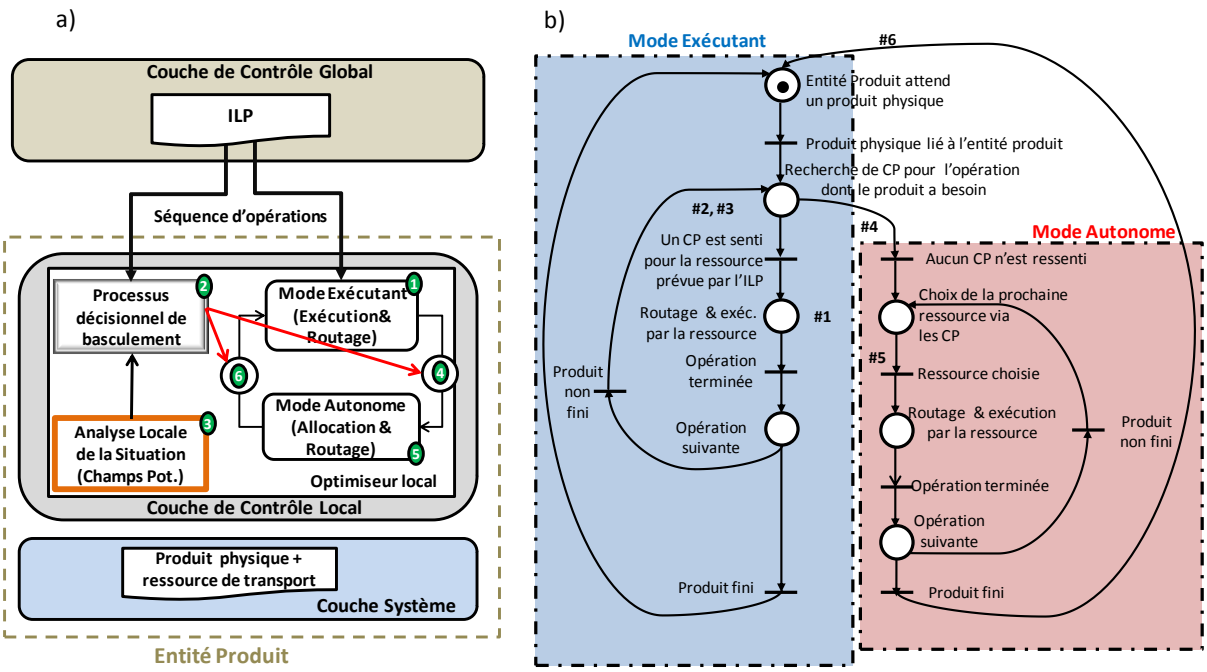


Figure 4.11a et 4.11b– Mécanisme de basculement au sein d'une entité produit

Dans le mode exécutant, comme indiqué précédemment, les entités « produit » utilisent les CPs pour atteindre les ressources qui leur ont été assignées dans la séquence optimale fournie par l'ILP (#1 des figures 4.11a et 4.11b). Dans ce mode, les CP ne sont pas utilisés pour l'allocation des produits aux ressources. Cependant, ils sont utilisés pour déclencher le processus décisionnel de basculement (#2 des figures 4.11a et 4.11b). En effet, à chaque fois qu'une entité « produit » avance vers une ressource suivant la séquence fournie par l'ILP, elle vérifie que cette ressource émet un CP et effectue donc une analyse locale de la situation (#3 des figures 4.11a et 4.11b). Si cette ressource n'émet aucun champ (i.e. la ressource ou un tronçon de convoyage permettant de s'y rendre est indisponible), l'optimiseur local intégré dans l'entité « produit » déclenche le basculement en mode autonome (#4 des figures 4.11a et 4.11b). Dans ce mode l'entité utilise les CP pour réaliser à la fois son routage mais aussi son allocation dynamiquement (#5 des figures 4.11a et 4.11b). Ainsi la séquence optimale fournie par l'ILP n'est plus respectée. Les entités non concernées par l'évènement inattendu restent en mode exécutant. En d'autres termes, la fonction ordonnancement est descendue de la couche de contrôle global d'ORCA-FMS vers la couche de contrôle local uniquement pour les entités concernées. Les entités ayant basculé restent dans le mode autonome jusqu'à la fin de leur produit physique actuel. Lorsqu'un produit physique est terminé et qu'un nouveau produit physique est lié à l'entité, celle-ci rebascule dans le mode exécutant pour commencer ce nouveau produit physique (#6 des figures 4.11a et 4.11b).

5.2. Justification du choix du mécanisme de basculement retenu

Le premier choix effectué en lien avec le rôle du mécanisme de basculement d'ORCA-FMS est de fournir uniquement la séquence des opérations (sans datation) à la couche de contrôle local au lieu d'un ordonnancement complet (avec datation). L'objectif derrière ce choix est de rendre le système plus robuste. En effet, fournir une séquence (non datée) aux optimiseurs

locaux permet d'éviter les décalages éventuels, dus aux différences entre le modèle résolu par l'ILP et la situation réelle sur laquelle se base les optimiseurs locaux. Ainsi, les entités ne basculent pas en mode autonome à cause d'écarts minimes entre le modèle et la réalité, ce qui limite la nervosité du système. Les lecteurs intéressés peuvent consulter (Barbosa et al., 2012a) au sujet de la nervosité qui peut être représentée dans ORCA-FMS en termes de nombres de basculements. Grâce à ce choix (séquence au lieu d'ordonnancement), le basculement en mode autonome n'apparaît que lors de grandes différences entre situation réelle et situation prévue comme lors de la panne d'une ressource qui aboutit à une disparition des champs de potentiel émis par celle-ci.

Le second choix se situe au niveau du basculement (retour) entre mode autonome et mode exécutant. Ce basculement n'est pas traité dans la littérature (ou de façon très floue) à cause de sa complexité. En effet, pour revenir à un état prévu, il faut que l'optimiseur global recalcule un ordonnancement intégrant l'évènement inattendu survenu. Le premier problème ici est le temps nécessaire pour que l'optimiseur global propose une nouvelle solution. Pour des raisons exposées précédemment, un ILP a été préféré à une heuristique dans cette application mais, dans une optique de re-calcul dynamique, l'ILP n'est bien évidemment pas adapté (temps de calcul non maîtrisé, pas de garantie sur l'obtention d'une première solution dans des temps fixés…). Cependant, même avec une (méta)heuristique à la place de l'ILP comme optimiseur global, les deuxième et troisième problèmes suivant perdurent. Le deuxième problème provient du fait que le système n'est plus dans son état initial lors du re-calcul de la couche globale. Ainsi le problème résolu doit être paramétrable afin de calculer un ordonnancement à partir d'un état différent et observé du système. Le troisième problème est que le système aurait tout intérêt à continuer d'évoluer pendant ce re-calcul, surtout s'il prend du temps. Ainsi l'ordonnancement ne doit pas être recalculé sur l'état actuel du système mais sur l'état dans lequel sera le système quand l'ordonnancement sera prêt. Ceci implique d'avoir 1) une estimation du temps de résolution du problème, qui évolue fortement avec le nombre de produits, ressources à traiter et 2) une estimation de l'état futur (cible) du système qui nécessite une simulation, par exemple, précise du fonctionnement du niveau réactif. À partir de ces deux éléments un point de reprise doit être défini pour que toutes les entités rebasculent en mode exécutant et suivent l'ordonnancement nouvellement calculé par le niveau de contrôle global.

Finalement, la compatibilité entre les entités « produits » en mode exécutant et celles en mode autonome n'est pas étudiée dans cette application d'ORCA. Ainsi il se peut qu'une entité « produit » en mode autonome vienne « voler » la ressource destinée à une autre en mode exécutant. Cette dernière devrait alors aussi basculer en mode autonome car sa séquence de machine optimale n'est plus réalisable. Ainsi si l'évènement ayant fait basculer certaines entités tarde à être corrigé/réparé, il se peut que l'ensemble des entités finisse en mode autonome. La gestion de cette « cohabitation » est un problème à part entière qui n'est pas traité dans cette thèse. L'instauration de priorités dynamiques associées aux entités est une première piste qui pourrait être étudiée pour résoudre ce problème.

ORCA-FMS n'a pas été conçue dans l'objectif de résoudre tous ces problèmes mais de fournir une première application d'ORCA afin de montrer l'intérêt de cette approche et la façon selon laquelle elle peut être mise en œuvre et nous avons conscience que plusieurs choix qui sont présentés dans la suite de ce paragraphe mériteraient une étude plus approfondie. En l'occurrence, il a été choisi de ne pas réaliser de re-calcul au niveau de l'optimiseur global pour le moment : l'ILP n'est lancé qu'une fois, avant que la production débute, pour tous les produits. Le passage du mode autonome au mode exécutant est donc simplifié et réalisé à chaque fois qu'à une entité produit est associée un nouveau produit physique à réaliser (après déchargement du produit physique précédent). En effet, lorsqu'un nouveau produit physique débute sa réalisation, l'entité « produit » associée suit la séquence proposée par l'ILP (en mode exécutant donc), même si le produit physique précédent s'était terminé en mode autonome. Ainsi si une perturbation est locale ou ne dure pas dans le temps, toutes les entités « produit » vont revenir dans le mode exécutant au bout d'un certain temps et suivre la séquence prévue par l'ILP pour leurs futurs produits physiques. Plusieurs points de vue peuvent par conséquent être distingués. Du point de vue du produit physique, il n'y a pas de retour en mode exécutant. Cependant d'un point de vue système, il y a un retour progressif de toutes les entités « produit » en mode exécutant. Cette situation est illustrée figure 4.11. Dans cet exemple utilisant 10 entités « produit », deux entités basculent en mode autonome suite à un évènement inattendu. Après avoir fini leur produit physique en mode autonome, ils commencent un nouveau produit physique en mode exécutant. Le graphique avec le point de vue du système montre le pourcentage d'entités « produit » en mode exécutant, alors que les graphiques avec les points de vue des entités « produit » montrent le mode courant de celles-ci. Ainsi cette figure illustre comment, avec ORCA-FMS, le système peut revenir à une situation plus optimisée globalement après un évènement inattendu.

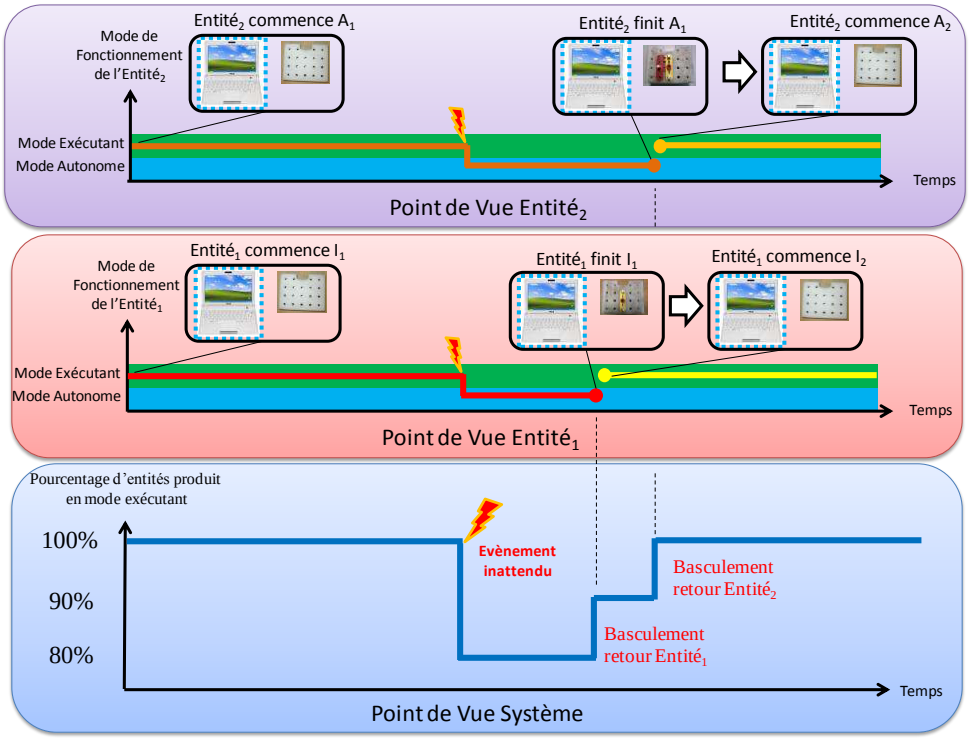


Figure 4.12– Les différents points de vue sur le mécanisme de basculement

5.3. Différences notables entre ORCA-FMS et la littérature

Comme précisé dans le chapitre II ORCA, et donc ORCA-FMS, relève de la classe II-DHe. En effet, l'architecture peut dynamiquement évoluer et le contrôle peut basculer d'un mode hiérarchique (i.e. mode exécutant) à un mode hétérarchique (i.e. mode autonome). Le pilotage est aussi hétérogène, le système peut donc simultanément contenir des entités en mode autonome, qui réagissent à un évènement inattendu via des champs de potentiel, et des entités en mode exécutant qui suivent la séquence fournie par l'ILP. Ceci constitue la différence majeure avec les architectures dynamiques de la sous-classe II-DHo présentées dans la typologie du chapitre II. En effet, ces dernières réagissent globalement à un évènement inattendu en faisant basculer la totalité des entités dans un mode de fonctionnement différent alors qu'ORCA-FMS permet aux entités non touchées par l'évènement inattendu de continuer à suivre la séquence optimale. Ainsi ORCA-FMS contribue à maintenir au mieux la performance globale.

Les travaux présentés dans (Zambrano et al., 2011), (Leitao et al., 2005) et (Valckenaers et al., 2007), ont mis en valeur le potentiel des architectures de cette sous-classe et ont servi de base permettant de tester de nouveaux mécanismes d'optimisation et d'interaction. Toutefois, des différences existent aussi entre ces trois architectures référencées dans la sous-classe II-DHe et ORCA-FMS.

(Zambrano et al., 2011) présente un des premiers travaux de l'équipe sur la myopie. L'architecture ORCA-FMS peut être vue comme une extension de l'approche proposée dans ce papier mais avec plusieurs différences majeures. Tout d'abord, un ILP a remplacé le mécanisme d'exploration d'arbre. Le mécanisme utilisé dans (Zambrano et al., 2011) était expérimental et le choix d'utiliser un ILP dans ORCA-FMS permet de se baser sur une approche plus formelle avec une optimalité (au sens du modèle considéré) des résultats prouvée. Ceci est d'autant plus vrai que le mécanisme à base d'exploration d'arbre ne prenait pas en comptes les autres entités dans le système et n'était donc pas globalement optimal mais juste « meilleur » que les champs de potentiel seuls. Deuxièmement, le mécanisme de basculement pour revenir à un mode exécutant n'avait pas été étudié dans (Zambrano et al., 2011). Troisièmement, ORCA-FMS est une application d'une approche plus générique (ORCA) alors que l'approche de (Zambrano et al., 2011) n'avait pour objectif que d'illustrer la myopie et était très spécifique au cas d'étude.

Les différences entre ORCA-FMS et (Leitao et al., 2005) se situent tout d'abord au niveau de la décision de basculement. En effet, dans ADACOR la décision de basculement est prise par les ressources (i.e. *operational holons*) alors que dans ORCA-FMS il s'agit des produits. Ensuite le basculement d'une ressource dans ADACOR conduit au dépôt de phéromones qui sont propagés aux autres ressources. Ainsi il y a une interaction indirecte entre les ressources selon un critère spatial pour changer leur « degré d'autonomie ». Dans ORCA-FMS il n'y aucune interaction entre les produits après le basculement de l'un d'entre eux. Troisièmement, ADACOR utilise une heuristique pour l'optimisation globale alors qu'ORCA-FMS utilise une approche exacte (ILP). Ensuite le mécanisme de basculement permettant le retour en mode exécutant n'est pas présenté explicitement dans ADACOR. Finalement, comme

pour (Zambrano et al., 2011), ADACOR a été conçue pour un cas d'étude spécifique avec des holons prédéterminés.

L'approche proposée dans (Valckenaers et al., 2007) est assez proche d'ORCA-FMS. Cependant, les approches utilisées en mode « autonome » sont très différentes. (Valckenaers et al., 2007) utilise une approche basée sur une exécution virtuelle des tâches (i.e. simulation) et une pré-réservation partielle (intention) des machines, permettant une prévision à court terme, alors qu'ORCA-FMS utilise les champs de Potentiel qui sont purement réactifs (pas de prévision). Le problème est attaqué d'une façon différente avec d'un côté une approche prédictive avec un horizon de simulation court afin de la rendre plus réactive, et de l'autre une approche purement réactive qu'on peut améliorer pour la rendre plus performante globalement. Le basculement du mode « exécutant » au mode « autonome » est assez similaire dans les deux architectures. Par contre comme pour les deux architectures précédentes le problème du retour en mode exécutant n'est pas explicitement traité dans (Valckenaers et al., 2007).

Ainsi ORCA-FMS va plus loin dans la réflexion en profitant des résultats obtenus via ces trois architectures pour proposer un premier exemple concret de basculement retour pour une architecture de classe II-DHe. Ce basculement retour est à considérer comme un premier pas et non un aboutissement. Il permet d'ouvrir de nouvelles pistes scientifiques (notamment celles présentées section 5.2 sur la définition d'un point de reprise).

L'architecture ORCA-FMS étant désormais définie, la partie suivante présente sa représentation via le modèle holonique proposé dans le chapitre précédent en vue d'une mise en œuvre.

6. Modélisation holonique de l'architecture ORCA-FMS

Le modèle holonique proposé chapitre III permet de modéliser les entités d'ORCA-FMS (i.e. entités « produit » et entités « ressource »), l'optimiseur global et au final l'architecture ORCA-FMS elle-même et les services produits ou fournis par les holons d'ORCA-FMS.

6.1. Modélisation de l'entité « produit »

L'entité « produit » prenant en charge un produit physique peut être facilement représentée au moyen d'un holon. Nous définissons par le terme « quasi-holon » un holon privé de capacité d'autonomie. La couche pilotée est constituée des quasi-holons produit physique et ressource de transport. Le produit physique correspond en effet uniquement à une partie physique ; la ressource de transport est quant à elle décomposée en deux couches : la partie purement physique en tant que couche pilotée et le contrôle de bas niveau associé à cette partie physique. Ces deux quasi-holons forment la couche pilotée du holon « Entité Produit ». L'optimiseur local associé est quant à lui représenté par la couche pilotage de cet holon. Cette représentation est illustrée figure 4.13. En reprenant la notation vue au chapitre III, l'entité « produit » présentée en figure 4.13 se note :

EntitéProduit= ($H_{GestionProduit}$ $\boldsymbol{\pi}$ {$H_{ContrôleResTransport}$, $\varphi_{produit}$}) ; ($H_{ContrôleResTransport}$ $\boldsymbol{\pi}$ $\varphi_{ResTransport}$) ;

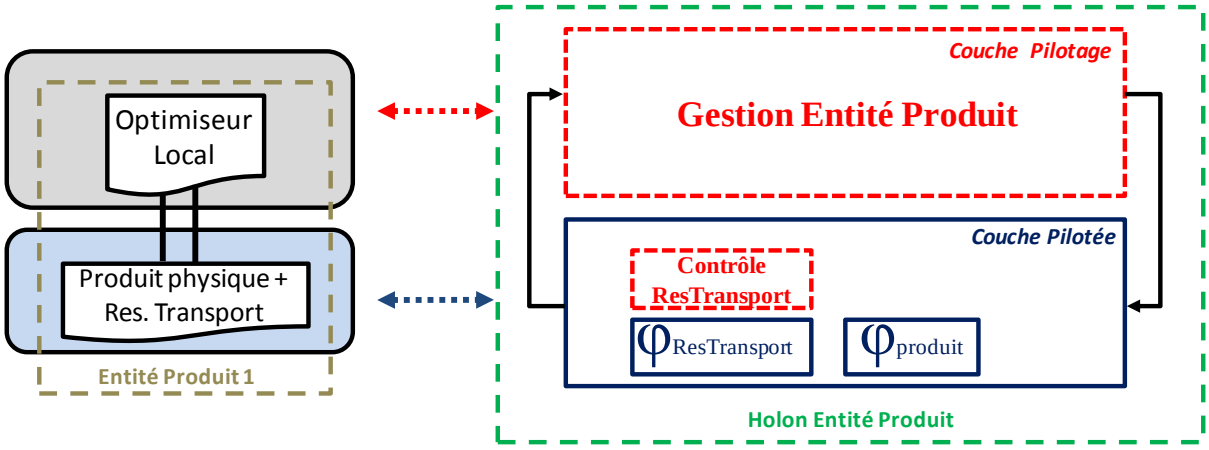


Figure 4.13– Représentation d'une entité « produit » d'ORCA-FMS

Le comportement de l'entité « produit » peut être défini selon deux rôles alternatifs correspondant à ses deux modes de fonctionnement dans ORCA-FMS et un rôle permanent gérant le passage d'un rôle alternatif à l'autre.

$$C_{EntitéProduit} = \{Executant - Autonome,\quad Gestion\ Rôles\}$$

Les deux rôles sont définis par les descripteurs suivants.

Rôle = Exécutant	
Connaissance	Séquence de l'optimiseur global, état de l'entité « produit ».
Buts	Terminer le produit physique associé le plus rapidement possible en suivant la séquence.
Règles	Suivre la séquence de ressources de l'optimiseur global, effectuer le routage vers ces ressources grâce aux champs de potentiel.

Rôle = Autonome	
Connaissance	État actuel du système, état de l'entité « produit ».
Buts	Terminer le produit physique associé le plus rapidement possible.
Règles	Effectuer l'allocation du produit aux ressources et le routage vers ces ressources grâce aux champs de potentiel.

La gestion des rôles est décrite par les descripteurs suivants.

Rôle = Gestion Rôles	
Connaissance	Séquence fournie par l'optimiseur global, état actuel du système, état de l'entité « produit ».
Buts	Choisir le meilleur rôle possible pour la fabrication du produit physique.
Règles	**{Si** l'état réel du système permet de suivre la séquence, **Alors** jouer le rôle Exécutant, **Sinon** jouer le rôle Autonome}.

La « Gestion Rôles » peut être aussi décrite par le graphe présenté figure 4.14.

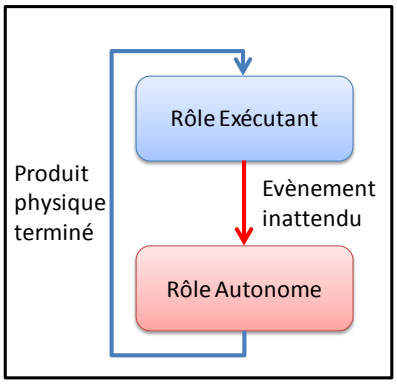


Figure 4.14– Gestion Rôle de l'entité « produit »

L'entité « produit » débute dans le rôle Exécutant. Si un évènement inattendu apparaît, elle prend le rôle autonome afin de pouvoir le gérer localement. Elle revient dans le rôle exécutant lorsqu'elle finit le produit physique qui lui est associé.

6.2. Modélisation de l'entité « ressource »

L'entité « ressource » peut être aussi représentée au moyen d'un holon. La file d'attente de la ressource, son stock et le robot associé sont représentés par des quasi-holons (sans autonomie d'un point de vue pilotage). Ceux-ci forment la couche pilotée du holon « Entité Ressource ». L'optimiseur local associé est quant à lui représenté par la couche pilotage de cet holon. Cette représentation est illustrée figure 4.15. En reprenant la notation vue au chapitre III, l'entité « ressource » présentée en figure 4.15 se note :

EntitéRessource= $(H_{GestionRessource} \pi \{H_{ContrôleFile}, H_{ContrôleStock}, H_{ContrôleRobot}\})$;
$(H_{ContrôleFile} \pi \varphi_{file})$; $(H_{ContrôleStock} \pi \varphi_{stock})$; $(H_{ContrôleRobot} \pi \varphi_{robot})$

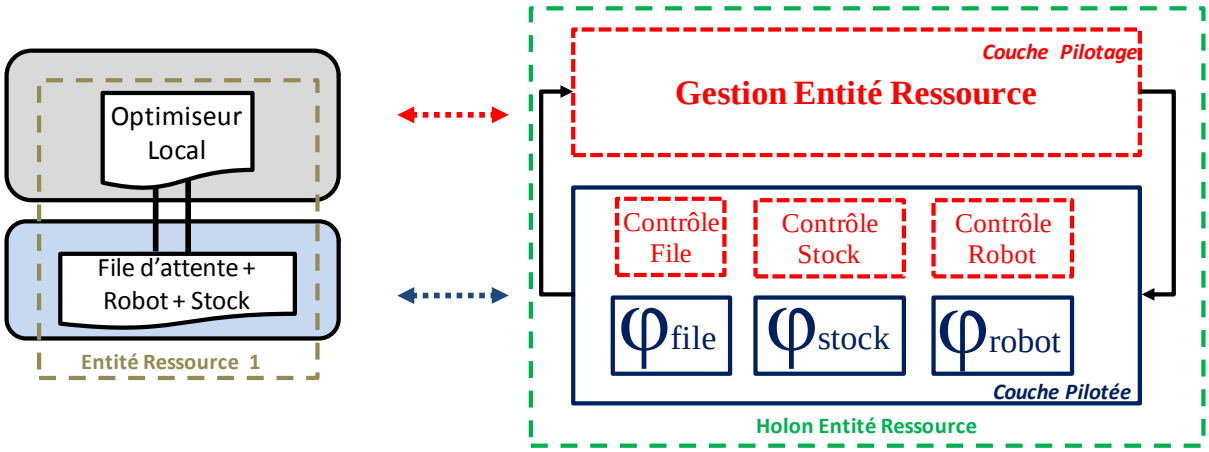


Figure 4.15– Représentation d'une entité ressource d'ORCA-FMS

Le comportement de l'entité « ressource » consiste dans cette application à gérer les champs de potentiel qu'elle émet. Ce comportement est représenté par :

$C_{EntitéRessource} = \{GestionnaireCP, \emptyset\}$

Le seul rôle (d'un point de vue pilotage) est « gestionnaire CP » (Champs de Potentiel) et donc la « Gestion Rôles » est vide pour l'entité « ressource ». Le rôle « gestionnaire CP » est représenté par les descripteurs suivants :

Rôle = Gestionnaire CP	
Connaissance	État actuel de la ressource (stock, file d'attente…), état du système.
Buts	Émettre des CPs correspondant à la situation actuelle de l'entité « ressource », Propager les CPs reçus.
Règles	Atténuer les CP émis en fonction de l'état de l'entité « ressource », atténuer les CPs en fonction des caractéristiques de l'environnement dans lequel ils se propagent.

6.3. Modélisation de l'optimiseur global

L'optimiseur global peut aussi être représenté par un holon. Il s'agit du holon de plus haut niveau avec une couche pilotage correspond à la Gestion de l'Optimisation globale du système et une couche pilotée correspondant à la totalité de la couche de contrôle local. Cet holon est représenté figure 4.16. En reprenant la notation vue au chapitre III, l'optimiseur global représenté figure 4.16 se note :
Optimiseur Global = ($H_{GestionOptimisationGlobale}$ π CoucheContrôleLocal)

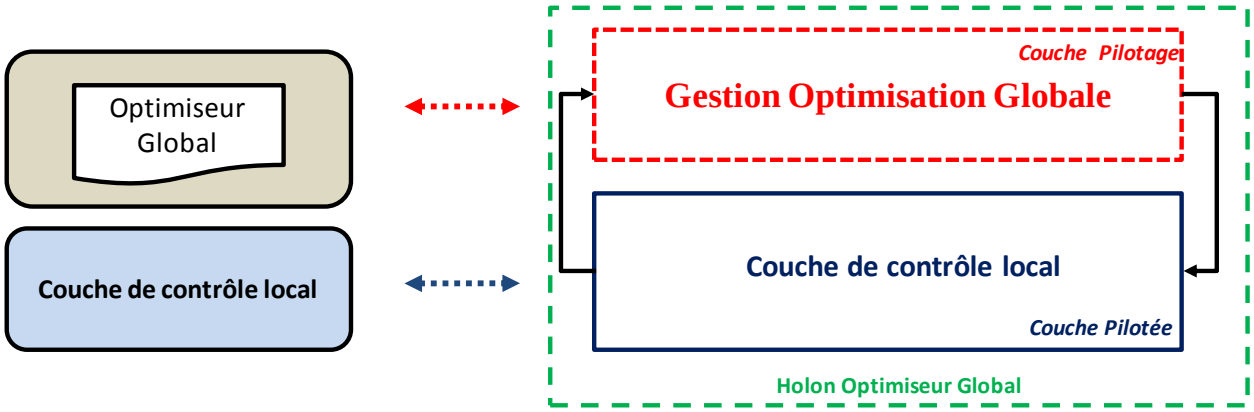


Figure 4.16– Représentation de l'optimiseur global

Le comportement du holon correspondant est représenté par :
$C_{OptimiseurGlobal} = \{OptimisationGlobale, \quad \emptyset\}$

Dans ORCA-FMS, l'optimiseur global n'a qu'un seul rôle : optimiser globalement la production en suivant une unique stratégie d'optimisation. Ainsi dans cette application, le rôle « Gestion rôles » est vide. Le rôle unique « optimisation globale » est décrit de la façon suivante.

Rôle = Optimisation Globale	
Connaissance	État initial du système, données sur le système, production demandée.
Buts	Obtenir la meilleure performance globale pour le système.
Règles	Algorithme permettant de calculer le meilleur ordonnancement pour la production demandée dans le système.

6.4. Modélisation de l'architecture globale

Cette section présente la modélisation de l'architecture globale avec le modèle holonique associé. L'exemple figure 4.17, présente une architecture ORCA-FMS complètement modélisée par des holons. Elle est composée de 16 entités (10 Entités Produit (EP) et 6 Entités Ressource (ER)) et 1 optimiseur global. Seules les EP_1 et ER_1 sont détaillées mais l'ensemble des EP_X et ER_Y correspondent à des holons de même structure. En reprenant la notation vue au chapitre III, l'architecture présentée en figure 4.17 se note :
Architecture ORCA-FMS = ($H_{OptimiseurGlobal}$ π {H_{EP1}, …, H_{EP10}, H_{ER1}, …, H_{ER6}}),

L'écriture des entités « ressource » H_{ERy} et entités « produit » H_{EPx} reprenant les écritures vues précédemment.

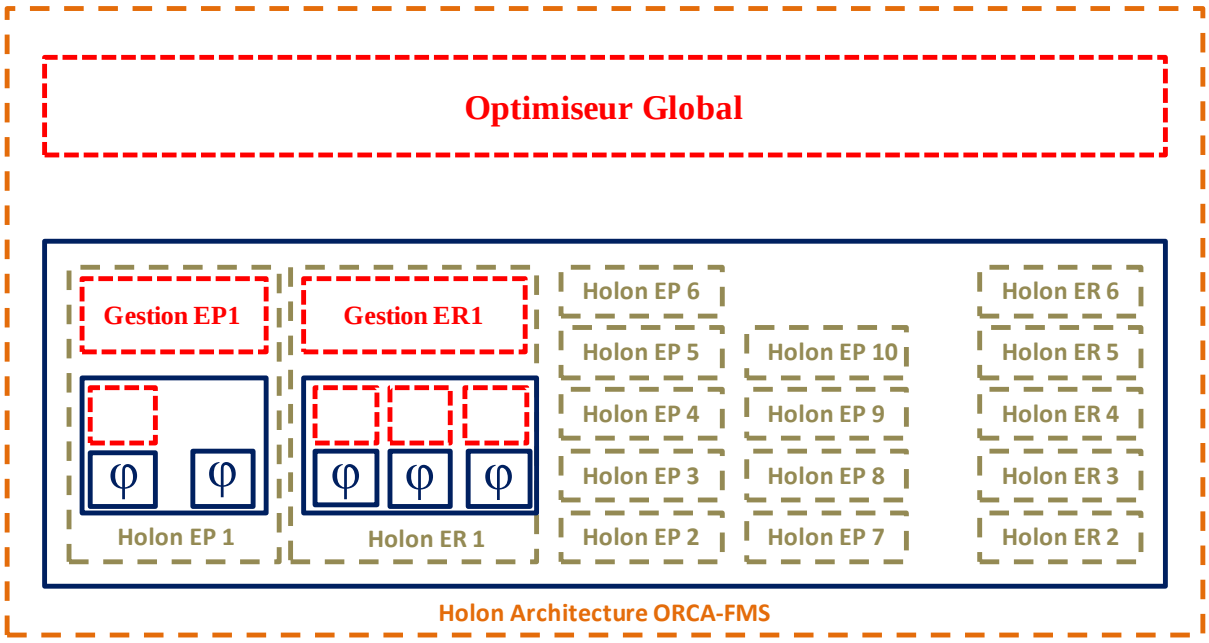


Figure 4.17– Exemple d'architecture ORCA-FMS modélisée

Les interactions entre entités « produit » et « ressource » peuvent être représentés par le modèle d'interactions basé sur les services présentés dans le chapitre III (figure 3.8). Ainsi les services échangés entre les holons d'un SFP sont les suivants (figure 4.18) :

- S_{EI} = Service Espace Informationnel : Communication.
- $S_{EP=}$ Service Espace Physique : Transitique.
- S_{TI} = Service Temps Informationnel : Mémorisation.
- S_{TP} = Service Temps Physique : Stockage
- S_{FI} = Service Forme Informationnel : Traitement.
- S_{FP} = Service Forme Physique : Usinage.

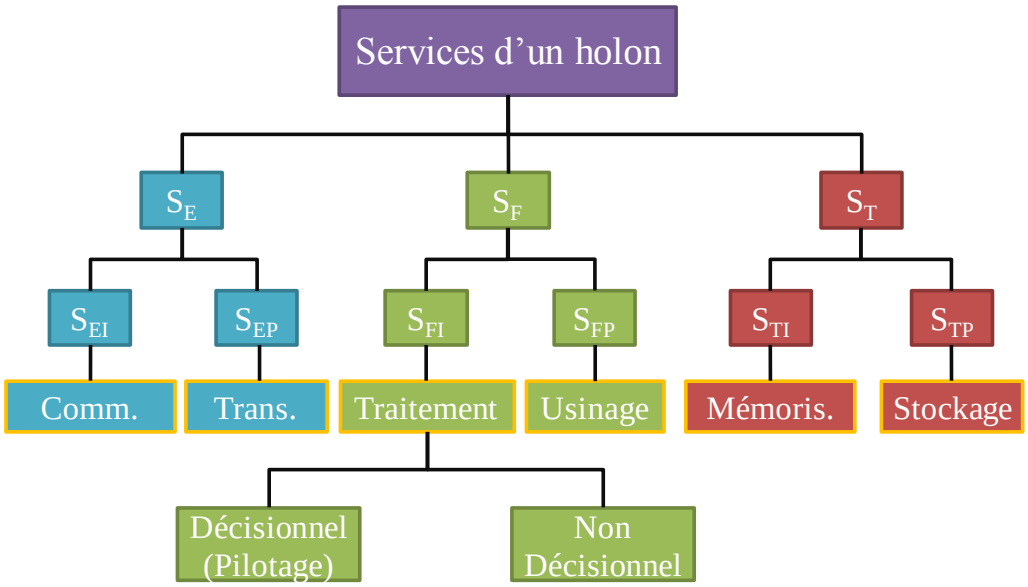


Figure 4.18– Services fournis ou demandés par un holon dans un SFP

Tous ces services peuvent a priori exister dans un SFP. Les plus importants dans ce cas d'utilisation d'ORCA pour un SFP sont :

- Les services d'usinage/d'assemblage qui composent la gamme de fabrication des produits physiques et qui vont donc être fournis par les entités « ressource » aux entités « produit ». La fourniture de ces services correspond à la problématique d'allocation des produits aux ressources dans les SFP.
- Les services de transitique permettant aux entités « produit » d'atteindre les entités « ressource » nécessaires. Ceci est à l'origine de la problématique de routage dans le SFP.

7. Conclusion

Ce chapitre a permis de présenter une application possible d'ORCA à une classe de problèmes concrets : le pilotage des SFP. Cette application, nommée ORCA-FMS, utilise un ILP pour l'optimisation globale et une approche par champs de potentiel pour l'optimisation locale. Après présentation de ces deux approches, le mécanisme de basculement d'ORCA-FMS a été détaillé et positionné par rapport aux mécanismes existants.

Les entités composant ORCA-FMS ont ensuite été modélisées via le modèle holonique proposé dans le chapitre III. Ainsi des modèles structurel, comportemental et d'interactions des entités d'ORCA-FMS ont été présentés.

Le chapitre suivant présente l'étude de la validité d'ORCA-FMS sur le cas d'étude de la cellule flexible AIP PRIMECA de Valenciennes en simulation, dans un premier temps.

Chapitre V : Étude de la validité d'ORCA-FMS en simulation

1. Introduction

Ce chapitre étudie la validité de l'architecture ORCA-FMS en simulation. Dans un premier temps le cas d'étude (basé sur une cellule existante) est détaillé puis la plateforme, les contraintes et le modèle de simulation considérés sont définis. Finalement, des scénarios de simulation sont présentés pour étudier le comportement de chacune des couches de l'architecture puis de l'architecture complète en termes de performances locales et globale.

2. Le cas d'étude

Le cas d'étude utilisé est celui de la cellule de production de l'AIP-PRIMECA de Valenciennes visible sur la figure 5.1. Cette cellule a été utilisée avec deux objectifs :

- Permettre l'application d'ORCA-FMS à un problème concret en simulation et ensuite (chapitre VI) en expérimentation,
- Proposer un cas d'étude de référence afin de pouvoir comparer différentes architectures concurrentes.

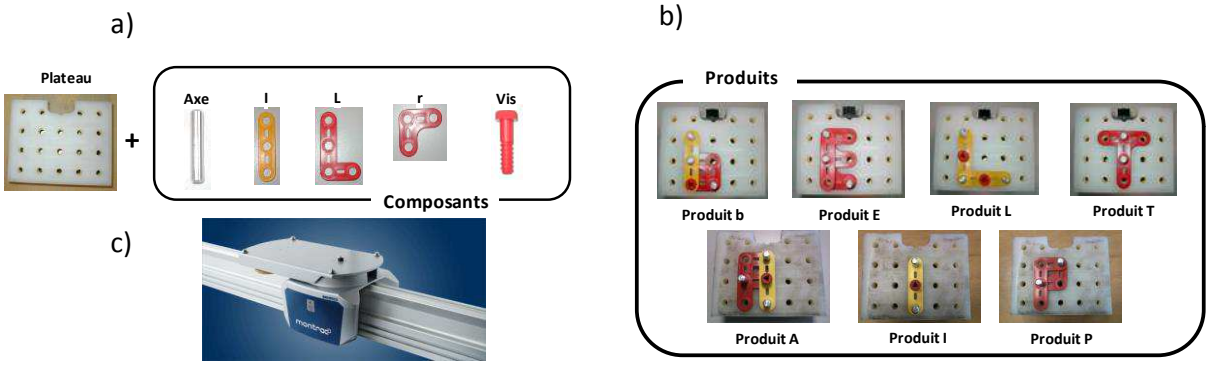
La cellule de l'AIP PRIMECA a été présentée et modélisée de manière formelle dans le benchmark proposé dans (Trentesaux et al., 2013) et un site internet[3] permet d'accéder à toutes les données présentées dans cette section 2. Ce benchmark propose aussi différents scénarios dynamiques permettant de tester la robustesse des approches de pilotage (ex. apparition d'une nouvelle ressource, problème de qualité). Les données présentées dans cette section reprennent les éléments clés de ce benchmark et sont présentées dans trois sous-sections : les données relatives aux produits, celles relatives aux ressources et celles relatives au système de transport.

[3] http://www.univ-valenciennes.fr/bench4star/

Figure 5.1– Cas d'étude

2.1. Données sur les produits à fabriquer

Les cinq composants de base disponibles dans la cellule (« Axe », « I », « L », « r » et « Vis ») ainsi que le plateau accueillant l'assemblage sont représentés sur la figure 5.2a. Les composants sont assemblés sur un plateau vierge déposé sur chaque navette (i.e. ressource de transport présenté figure 5.2c) lors de l'opération de chargement. Ils sont répartis entre les différentes ressources réalisant des opérations d'assemblage (présentées dans la partie données sur les ressources) et leur stock est supposé suffisant (sauf en cas de test de robustesse mentionnant le contraire).



Figures 5.2a, 5.2b et 5.2c– Composants, produits et navette

Avec ces cinq composants, sept produits différents peuvent être assemblés sur la cellule (« B », « E », « L », « T », « A », « I » et « P ») comme illustré figure 5.2b. Les produits assemblés sont déchargés avec le plateau à la fin de la séquence de production.

La séquence de production de ces produits est composée d'actions élémentaires, appelées opérations, réalisées sur les ressources de la cellule. Ces opérations sont au nombre de huit (« Chargement_Plateau », « Montage_Axe », « Montage_r », « Montage_I », « Montage_L »,

« Montage_Vis », « Inspection automatisée » et « Déchargement_Plateau »). Par exemple « Montage_Axe » signifie qu'un Axe est pris du stock de la ressource concernée et monté sur le plateau correspondant au produit en cours d'assemblage (l'emplacement du montage est déterminé par la ressource en fonction du produit). L'inspection automatisée est effectuée par un système de vision présenté par la suite.

Une séquence de production est associée à chaque type de produits. Elle est toujours structurée de la même façon : on charge le plateau vierge sur une navette, une série d'opérations d'assemblage est effectuée, puis le produit est inspecté et enfin le plateau contenant le produit est déchargé. Si deux opérations consécutives d'une séquence de production ne sont pas réalisées sur la même ressource, un transport est alors nécessaire. La table 5.I présente la séquence de production associée à chaque produit.

Table 5.I– Séquence de production des produits

B	E	L	T	A	I	P
Chargement	Chargement	Chargement	Chargement	Chargement	Chargement	Chargement
Montage_Axe	Montage_Axe	Montage_Axe	Montage_Axe	Montage_Axe	Montage_Axe	Montage_Axe
Montage_Axe	Montage_Axe	Montage_Axe	Montage_Axe	Montage_Axe	Montage_Axe	Montage_Axe
Montage_Axe	Montage_Axe	Montage_Axe	Montage_Axe	Montage_Axe	Montage_Axe	Montage_Axe
Montage_r	Montage_r	Montage_I	Montage_L	Montage_r	Montage_Vis	Montage_L
Montage_r	Montage_r	Montage_I	Inspection	Montage_L	Inspection	Inspection
Montage_I	Montage_L	Montage_Vis	Déchargement	Montage_I	Déchargement	Déchargement
Montage_Vis	Inspection	Montage_Vis		Montage_Vis		
Inspection	Déchargement	Inspection		Inspection		
Déchargement		Déchargement		Déchargement		

Pour corriger d'éventuels problèmes de qualité sur certains produits, une opération supplémentaire appelée « réparation » peut être, le cas échéant, insérée juste avant le déchargement. Cette opération est manuelle et corrige les problèmes de qualité avec un taux de réussite de 100%.

Dans les scénarios en fonctionnement normal, les ordres de fabrication provenant des clients peuvent être de trois formes : « BELT », « AIP » et « LATE ». Ces ordres de fabrication contiennent donc trois ou quatre produits qui peuvent être fabriqués dans n'importe quel ordre. L'ordre de fabrication est considéré comme fini lorsque l'ensemble de ses produits est réalisé. La figure 5.3 suivante illustre les 3 ordres de fabrication.

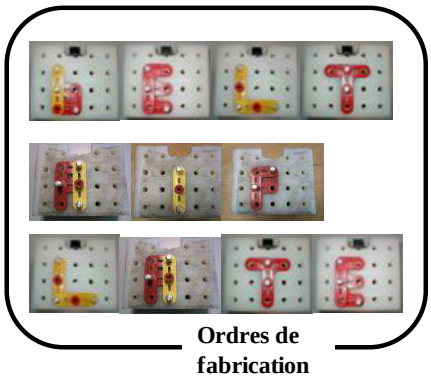


Figure 5.3– Ordres de fabrication

2.2. Données sur les ressources

La cellule est composée de sept ressources (voir figure 5.4) :

- Un poste de chargement / déchargement (R_1)
- Trois postes d'assemblage (R_2, R_3 et R_4)
- Une unité d'inspection automatisée (R_5)
- Une unité de réparation (R_6), seul poste manuel
- Une station optionnelle (R_7), utilisée dans certains scénarios avec perturbation.

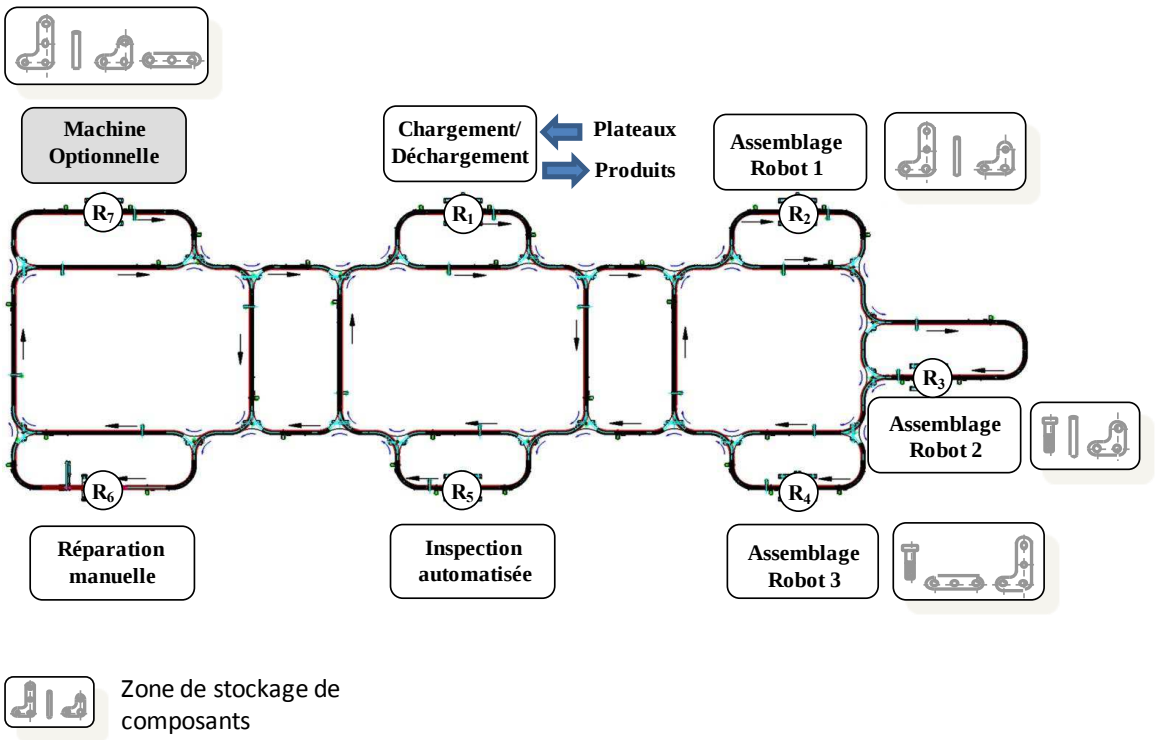


Figure 5.4– Localisation des ressources de la cellule

Chaque ressource est associée à sa propre zone de stockage de composants. Lors de scénarios du benchmark dits « dynamiques » il est possible de causer une perturbation sur une zone de stockage spécifique tout en laissant les autres zones fonctionnelles.

La table 5.II récapitule l'affectation des opérations de production aux ressources. La valeur présentée représente le temps de production en secondes sur cette ressource pour l'opération considérée. Une absence de chiffre signifie que la ressource ne peut pas effectuer l'opération. La ressource R_7 n'est effective que dans certains scénarios dynamiques, lors de fonctionnements normaux elle n'effectue aucune opération. Chaque ressource ne peut traiter qu'un seul produit à la fois.

Table 5.II– Temps de production pour chaque ressource

Opération	R_1	R_2	R_3	R_4	R_5	R_6	R_7
Chargement	10	-	-	-	-	-	-
Déchargement	10	-	-	-	-	-	-
Montage_Axe	-	20	20	-	-	-	(20)
Montage_r	-	20	20	-	-	-	(20)
Montage_l	-	-	-	20	-	-	(20)
Montage_L	-	20	-	20	-	-	(20)
Montage_Vis	-	-	20	20	-	-	-
Inspection	-	-	-	-	5	-	-
Réparation	-	-	-	-	-	60	-

2.3. Données sur le système de transport

D'un point de vue transitique, les ressources sont connectées grâce à un système de transport. Ce système de transport est un monorail unidirectionnel avec des aiguillages rotatifs avant chaque divergence/convergence (Montratec AG, 2013). Ainsi ce système de transport peut être considéré comme un graphe fortement connexe, composé des nœuds suivants :

- R_1, R_2, R_3, R_4, R_5, R_6, R_7 (en blanc dans la figure 5.5) sont les nœuds ressources,
- n_1, n_2, n_3, n_4, n_5, n_6, n_7, n_8, n_9, n_{10}, n_{11}, (en gris dans la figure 5.5) sont les nœuds décisionnels où des décisions de routage doivent être prises. Par exemple en n_1, il faut décider d'aller vers n_2 ou n_{10}.

L'ensemble de ces nœuds est présenté sur la figure 5.5.

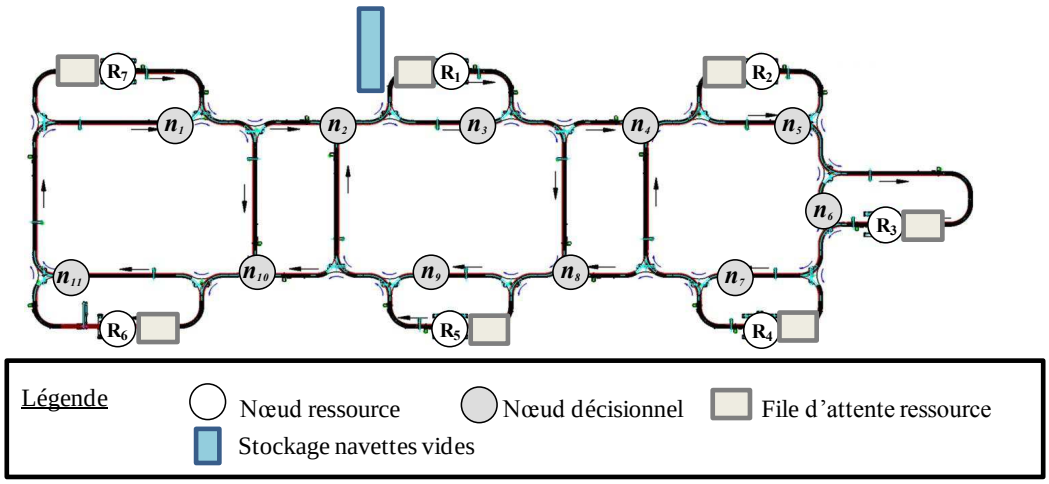


Figure 5.5– Système de transport

Les navettes (présentées en détails dans le chapitre suivant) parcourent le système de convoyage, sont autopropulsées et peuvent chacune transporter un produit de nœud en nœud. Le nombre de navettes est limité et donc, à chaque instant, un nombre (à fixer) maximum de produits peut être en cours d'assemblage sur la cellule. Chaque navette intègre un mécanisme de base permettant de ne pas entrer en collision avec les autres navettes, détecter les aiguillages et s'arrêter en face des ressources. Le convoyeur étant unidirectionnel la première navette qui entre sur un tronçon sera toujours la première à en sortir (règle *First In First Out* ou FIFO).

Pour des raisons de simplification, les navettes vides sont considérées comme stockées dans une zone dédiée, à côté de la ressource R1, et dont la capacité de stockage de navettes est considérée comme infinie (il est supposé qu'un mécanisme extérieur charge et décharge les navettes à cet endroit). Les navettes entrent et quittent ainsi le système sur cette zone. Elles commencent donc par accueillir un plateau (opération chargement sur R_1) avant d'entrer dans le système. Elles transportent ensuite le produit à travers la cellule pour lui permettre d'être assemblé avant de revenir sur R_1 afin qu'il soit déchargé. Elles sont ensuite stockées dans cette zone jusqu'à ce qu'elles soient réutilisées. Ainsi, il est impossible qu'une navette vide (sans plateau) circule dans la cellule.

Avant chaque ressource se trouve une file d'attente avec une capacité finie.

Les temps de transport théoriques entre chaque nœud de la cellule sont donnés dans la table 5.III. Ces temps sont minima et sont fixés par la vitesse réelle des navettes. En réalité, le transport peut prendre plus de temps à cause de ralentissements ou embouteillages sur la cellule.

Table 5.III– Temps de transport entre chaque nœud.

	n_1	n_2	n_3	n_4	n_5	n_6	n_7	n_8	n_9	n_{10}	n_{11}	R_1	R_2	R_3	R_4	R_5	R_6	R_7
n1		4	-	-	-	-	-	-	-	5	-	-	-	-	-	-	-	-
n2	-		4	-	-	-	-	-	-	-	-	5	-	-	-	-	-	-
n3	-	-		4	-	-	-	5	-	-	-	-	-	-	-	-	-	-
n4	-	-	-		4	-	-	-	-	-	-	-	5	-	-	-	-	-
n5	-	-	-	-		3	-	-	-	-	-	-	-	11	-	-	-	-
n6	-	-	-	-	-		4	-	-	-	-	-	-	-	5	-	-	-
n7	-	-	-	5	-	-		4	-	-	-	-	-	-	-	-	-	-
n8	-	-	-	-	-	-	-		4	-	-	-	-	-	-	5	-	-
n9	-	5	-	-	-	-	-	-		4	-	-	-	-	-	-	-	-
n10	-	-	-	-	-	-	-	-	-		4	-	-	-	-	-	7	-
n11	9	-	-	-	-	-	-	-	-	-		-	-	-	-	-	-	10
R_1	-	-	-	6	-	-	-	7	-	-	-		-	-	-	-	-	-
R_2	-	-	-	-	-	5	-	-	-	-	-	-		13	-	-	-	-
R_3	-	-	-	-	-	-	6	-	-	-	-	-	-		7	-	-	-
R_4	-	-	-	7	-	-	-	6	-	-	-	-	-	-		-	-	-
R_5	-	7	-	-	-	-	-	-	-	6	-	-	-	-	-		-	-
R_6	12	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-		13
R_7	-	6	-	-	-	-	-	-	-	7	-	-	-	-	-	-	-	

Un exemple détaillé d'utilisation de ce cas d'étude a été présenté dans (Trentesaux et al., 2013). Des scénarios de fonctionnement normal et perturbé sont proposés et illustrés par

l'utilisation de trois méthodes : un programme linéaire, un simulateur et une mise en œuvre réelle.

La partie suivante présente le simulateur utilisé pour appréhender la validité du comportement d'ORCA-FMS en simulation.

3. Présentation de la simulation

Cette section présente le simulateur et le modèle de simulation utilisé pour étudier le comportement d'ORCA-FMS en simulation.

3.1. Présentation du simulateur

Le simulateur utilisé a été développé à l'aide de NetLogo (Wilensky, 1999). NetLogo est un environnement de développement multi-agent adapté au pilotage distribué. Il fournit tous les éléments appropriés pour modéliser des entités, leur(s) comportement(s) et les interactions entre celles-ci. L'environnement NetLogo est facile à maîtriser, possède un langage de programmation intégré et permet de réaliser des interfaces graphiques personnalisées très facilement. Ces caractéristiques font de NetLogo un outil adapté au prototypage rapide de système multi-agent réactifs. Il permet de valider rapidement des concepts et d'obtenir des résultats pour des scénarios longs ou difficiles à mettre en place en expérimentation réelle.

L'environnement de développement NetLogo se base sur 3 entités principales :

- Les tortues qui représentent des entités décisionnelles mobiles. Le comportement de ces entités peut être défini individuellement. Les tortues conviennent parfaitement dans le cadre de travaux sur des produits dotés d'une capacité décisionnelle.

- Les patchs qui permettent de stocker localement de l'information. Celle-ci n'est accessible qu'aux entités proches des patchs. Ils permettent ainsi de gérer facilement des systèmes de contrôle basés sur la stigmergie (cf. section 3.2.2 du chapitre I).

- Les liens qui permettent de lier les tortues entre elles. Les liens possèdent les mêmes capacités de traitement que les tortues et permettent de construire la typologie du système.

3.2. Modèle de simulation utilisé

La cellule flexible du cas d'étude a été modélisée sous NetLogo en collaboration avec Thérèse Bonte du laboratoire TEMPO, équipe PSI. Cette cellule est illustrée sur la figure 5.6. Dans ce modèle, chaque entité (produit, ressource et nœud) est représentée par une tortue. Un comportement peut être associé à chaque tortue et des informations peuvent être déposées au niveau des nœuds. Chaque nœud décisionnel du cas d'étude contrôle deux aiguillages : un premier où deux chemins convergent puis un second où le chemin diverge vers deux autres. Les décisions sont prises avant le premier aiguillage. Toutes les décisions sont prises par les tortues sur ces nœuds décisionnels selon le mode de décision choisi dans les scénarios.

La topologie du réseau est représentée par des tortues « nœuds » reliées par des liens. Le réseau est ainsi modélisé par un graphe dans lequel il est facile de calculer le plus court chemin (algorithme de Dijsktra).

Deux races (i.e. *breed* en Netlogo) de tortues sont utilisées dans cette modélisation :

- des tortues dont la position est statique permettant de représenter les nœuds du réseau. Ces nœuds peuvent être de différents types : les nœuds « intersection », les nœuds « ressource », les nœuds « de décision », les nœuds « chargement », les nœuds « déchargement ».
- des tortues qui évoluent sur le réseau représentant les navettes et les produits.

Deux races de liens sont utilisées :

- Les liens orientés qui lient les nœuds du réseau et représentent le système de convoyage
- Les liens bi-directionnels qui unissent une navette et un produit. Lorsque un produit est lancé dans le système il est lié à une navette par ce type de lien jusqu'à la fin de sa fabrication.

Les produits terminés ou en attente sont représentés à côté du poste de chargement-déchargement. Les produits en cours de fabrication sont associés à la navette les transportant pour plus de clarté. Lorsqu'un produit ne dispose d'aucune ressource pour effectuer sa prochaine opération, il attend (en circulant) sur la boucle principale du convoyeur afin de ne pas bloquer les autres produits. Finalement la ressource 1 effectue à la fois l'entrée et la sortie des produits dans le système.

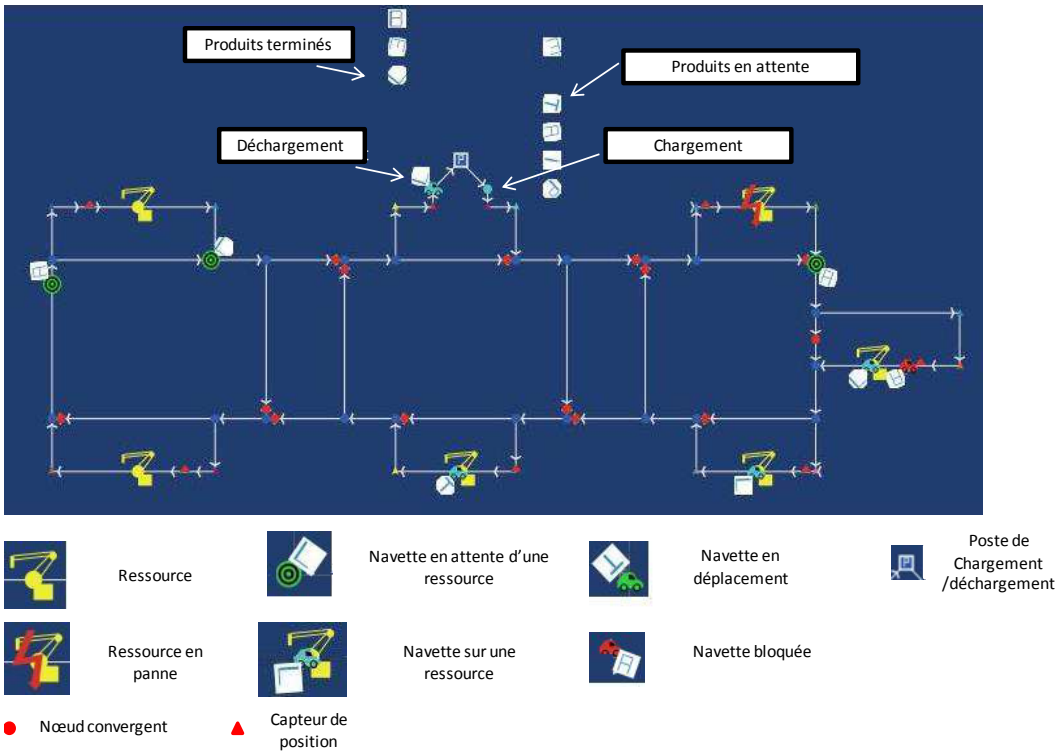


Figure 5.6– Représentation Netlogo du cas d'étude

Fonctionnement général :

La simulation est à temps discrétisé. À chaque incrément de l'horloge, les tortues « navettes » qui sont susceptibles d'avancer et pour qui c'est physiquement possible (ie. l'espace inter-navette et les priorités aux intersections sont respectés) avancent. La navette réagit ensuite en fonction de sa nouvelle position. Lorsqu' une navette a besoin des champs de potentiel sur un nœud, celui-ci met à jour ses champs en fonction de ce qu'il reçoit des ressources. Ainsi les champs ne sont calculés que si nécessaire.

Les algorithmes suivants présentent le fonctionnement général de différentes fonctions du simulateur.

Création du réseau ()
Lecture de la topologie du réseau
Lecture de la description des navettes (dimensions, vitesse)
Lecture des produits, de leur date de lancement, de leur priorité

Go ()
Avancer_les_navettes
Avancer heure (*tick* en netlogo)

Avancer_les_navettes ()
Pour chaque navette (*ask* navette en netlogo)
calculer_prochaine_position_théorique (calcule la prochaine position théorique en fonction de sa vitesse indépendamment des autres navettes)

Pour chaque navette
regarder s'il y aura des collisions avec d'autres navettes
Si oui :
Appliquer les règles de priorité et/ou règles de précédence et
avancer_la_navette (navette prioritaire)
Si non
Avancer_la_navette (navette courante)

Avancer_la_navette (n)
Avancer s
Si navette sur nœud de chgt : gere_chgt
Si navette sur nœud de dechgt : gere_dechgt
Si navette sur nœud ressource : gere_ressource
Si navette sur nœud de décision : gere_decision

Gere_chgt ()
Si t=heure de fin de chargement
La navette s'affecte au prochain produit (Création d'un lien navette/produit)
Calcul de la ressource de destination à l'aide des champs de potentiel
Calcul du chemin le plus court vers la prochaine destination
Gere_dechgt ()

```
    Si t=heure de fin de déchargement
        Mettre le  produit sur la navette en stock
        Calculer la prochaine destination (nœud de parking ou nœud de chargement)
```

```
Gere_ressource ()
    Si t=heure de fin de l'opération courante
        Si pas de prochaine opération
            Prochaine destination = nœud de déchargement
        Si prochaine opération
            Calcul de la ressource destination à l'aide des champs de potentiel
            Si prochaine ressource=ressource actuelle
                Rester au même endroit
            Sinon
                Calcul du chemin le plus court vers la prochaine destination
```

```
Gere_decision ()
    Si destination != nœud de déchargement
        Calcul de la ressource destination à l'aide des champs de potentiel
```

```
Calcul de la ressource destination à l'aide des champs de potentiel ()
    Calculer les champs émis par les différentes ressources (selon le mode de calcul actif)
    Choisir la ressource qui émet le plus grand champ
```

Les données d'entrées (état des ressources, ordres de fabrication, etc.) et de sorties (état des indicateurs, logs, etc.) sont lues/écrites dans des fichiers textes pour faciliter les interactions (imports/exports de données) avec d'autres composants logiciels (optimiseur, traceur de GANTT, etc.).

Les simulations présentées dans ce chapitre ont été lancées en vitesse accélérée dans NetLogo et les résultats ont été obtenus en moins de 5 secondes.

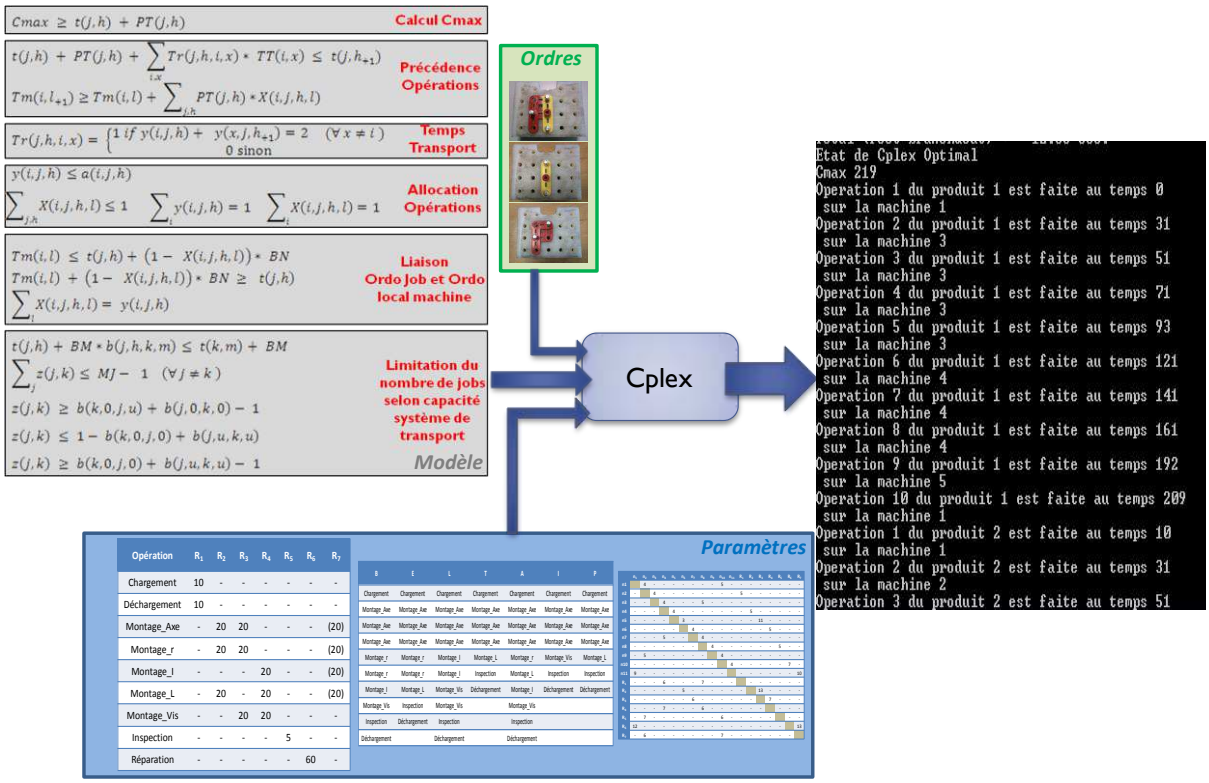
3.3. Modèle linéaire (ILP)

Le modèle linéaire a été mis en œuvre via le solveur Cplex ("IBM ILOG CPLEX Optimization Studio," 2013). Cplex est basé sur un algorithme de *branch-and-cut* couplé à des méthodes de Recherche Opérationnelle, le tout mis en œuvre par IBM ILOG. Ce solveur de programmation mathématique est le plus connu et utilisé de la communauté « Recherche Opérationnelle ».

La construction du modèle a été réalisée grâce à « *ILOG concert technology* ». Il s'agit d'une bibliothèque fournissant de nombreux objets compatibles avec les langages orientés objets tel que C#, Java ou .NET. Cela permet de créer des modèles sans connaître les méthodes et algorithmes intégrées dans Cplex et de s'interfacer facilement avec d'autres programmes orientés objets. Une fois le modèle créé, une instance de Cplex est ouverte pour résoudre le problème et les données peuvent ensuite être récupérées par le programme orienté objets. Il s'agit donc

d'un bon choix pour la communauté « informatique industrielle » habituée à des langages objets « classiques » ou pour toutes personnes souhaitant s'interfacer avec d'autres programmes orientés objets. L'autre avantage de ce mode de programmation est de pouvoir construire un modèle dynamiquement en intégrant ou non certaines contraintes et en modifiant le modèle selon les circonstances. La construction du modèle a été réalisée en C#.

La figure 5.7 présente d'une part les entrées de Cplex (i.e. les contraintes permettant de construire le modèle linéaire, les valeurs des paramètres et les ordres de fabrication à ordonnancer) et les sorties obtenues après résolution du modèle par Cplex notamment l'état de la solution (*Optimal*, *feasible* ou *no solution*) et un affichage personnalisé de l'ordonnancement des opérations.



Opération	R_1	R_2	R_3	R_4	R_5	R_6	R_7
Chargement	10	-	-	-	-	-	-
Déchargement	10	-	-	-	-	-	-
Montage_Axe	-	20	20	-	-	-	(20)
Montage_r	-	20	20	-	-	-	(20)
Montage_l	-	-	-	20	-	-	(20)
Montage_L	-	20	-	20	-	-	(20)
Montage_Vis	-	-	20	20	-	-	-
Inspection	-	-	-	-	5	-	-
Réparation	-	-	-	-	-	60	-

Figure 5.7– Entrées et Sorties de Cplex

La résolution du modèle linéaire par Cplex est réalisée hors ligne. Ainsi les résultats obtenus par Cplex sont stockés dans un fichier texte lu par Netlogo au démarrage de la simulation. Ce fichier en plus de contenir les ordres de fabrication, contient les ressources sur lesquels chaque produit doit être envoyé. La figure 5.8 présente un exemple de ce fichier texte pour les ordres A-I-P-A-I-P.

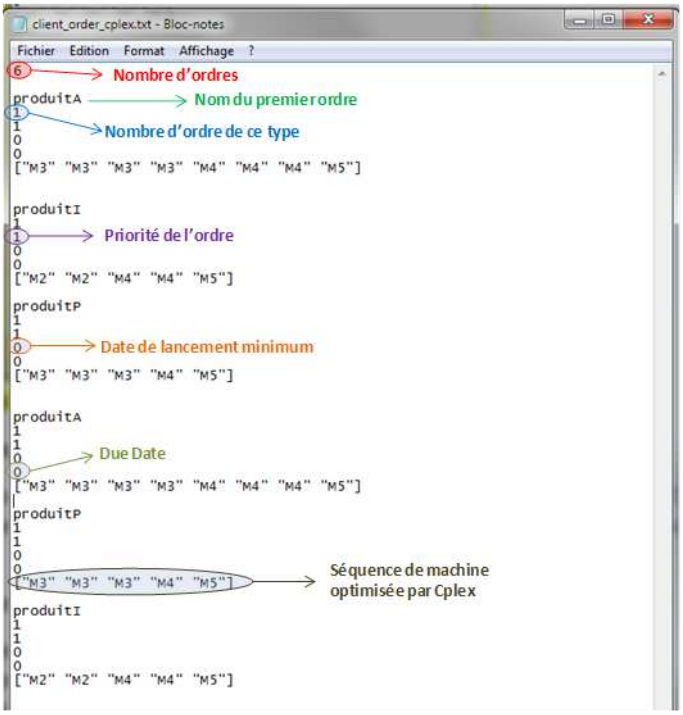


Figure 5.8– Fichier de synchronisation entre Cplex et NetLogo

3.4. Protocole expérimental de simulation

Le modèle NetLogo considère toutes les contraintes du cas d'étude présentées dans la section 2 précédente : les séquences de production des produits, les temps de production et la répartition des composants.

Les valeurs des paramètres sont les suivantes :

- Le nombre maximum de navettes simultanément dans la cellule est fixé à 10.
- La capacité de traitement des ressources est de 1 navette.
- La capacité de la file d'attente devant chaque ressource est de 1 navette.
- Les temps de transport initiaux sont ceux théoriques (table 4.III) mais peuvent être augmentés en fonction de l'état de la cellule.
- Les champs de potentiel initiaux émis par les ressources sont fixés à 200.

Les simulations présentées dans ce chapitre ont pour objectif d'étudier les performances à la fois globales (i.e. efficacité / contrôle de la myopie) et locales (i.e. réactivité) d'ORCA-FMS. Le protocole expérimental suivi est illustré figure 5.9.

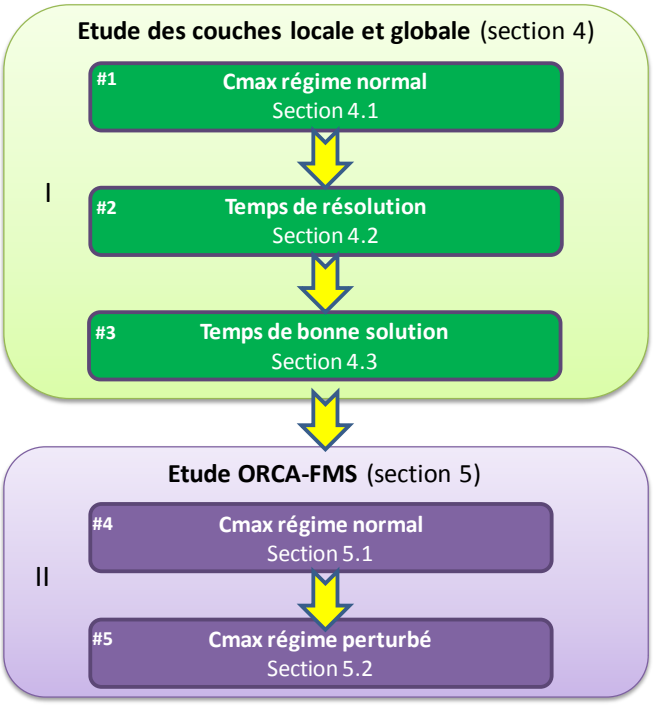


Figure 5.9– Protocole expérimental de simulation

Cette sous-section présente globalement le protocole expérimental dont les résultats seront présentés section 4 et 5. Avant de valider le comportement de l'architecture complète il convient de tester les couches individuellement (macro étape I de la figure 5.9). Les simulations proposées dans la section 4 vont :

- S1 : Étudier le comportement des champs de potentiel en fonctionnement normal et quantifier l'écart entre la solution « champs de potentiel » et celle « optimale » (donnée par l'ILP).
- S2 : Étudier le temps de calcul de l'ILP afin de trouver la solution « optimale » pour une campagne de production donnée.
- S3 : Étudier le temps de calcul de l'ILP pour trouver une solution « juste meilleure » que celle proposée par les champs de potentiel pour une campagne de production donnée.

Ces trois groupes de simulations (S1, S2 et S3) vont être effectués sur des ordres de fabrication de types A-I-P (incluant notamment le *data-set #B0* du benchmark (Trentesaux et al., 2013)). Avant d'utiliser les ordres de fabrication complets, chaque produit les composant (A, I et P) est testé individuellement. La difficulté des tests est croissante en débutant par 1 seul produit A, puis 2 produits A, et ainsi de suite jusqu'à 10 produits A (de même ensuite pour I puis P). L'opération est ensuite réalisée avec l'ordre de fabrication A-I-P, puis 2x A-I-P et 3x A-I-P (qui engage 9 navettes sur les 10 possibles). L'indicateur de performance utilisé pour la simulation S1 est le *Makespan* (aussi appelé Cmax) qui représente le temps total de fabrication de l'ordre de fabrication. Pour les simulations S2 et S3, l'indicateur étudié sera le temps de calcul.

Une fois les différentes couches validées, des simulations sont effectuées dans la section 5 pour valider le fonctionnement de l'architecture ORCA-FMS complète (macro étape II de la figure 5.9). Deux variantes de l'architecture seront testées : une première qui considère en entrée uniquement l'ordre de lancement des produits déterminé par la couche haute (ORCA-FMS 1) et une seconde qui correspond à une prise en compte de la séquence complète des ressources à visiter calculée par la couche haute (ORCA-FMS 2). Deux groupes de simulation vont être lancés pour :

- S4 : Comparer le comportement d'ORCA-FMS 1 & d'ORCA-FMS 2 aux résultats fournis par l'ILP et les champs de potentiel seuls en fonctionnement normal.

- S5: Comparer le comportement d'ORCA-FMS 1 & d'ORCA-FMS 2 aux résultats fournis par l'ILP et les champs de potentiel seuls en fonctionnement perturbé.

Ces deux groupes de simulation seront lancés avec des ordres de fabrication de type A-I-P. L'indicateur de performance utilisé pour ces simulations est le Makespan. La table 5.IV dresse un récapitulatif de tous ces scénarios.

Table 5.IV– Scénarios de simulation

Simulation	Mode de fonctionnement	Approche(s) étudiée(s)	Indicateur étudié	Campagnes lancées
S1	Normal	Champs de potentiel / ILP	Makespan	Produits individuels (A, I & P) Ordres de fabrication complets (A-I-P)
S2	Normal	ILP	Temps de résolution	Produits individuels (A, I & P) Ordres de fabrication complets (A-I-P)
S3	Normal	ILP	Temps « bonne solution »	Produits individuels (A, I & P) Ordres de fabrication complets (A-I-P)
S4	Normal	ORCA-FMS 1 ORCA-FMS 2	Makespan	Ordres de fabrication complets (A-I-P)
S5	Perturbé	ORCA-FMS 1 ORCA-FMS 2	Makespan	Ordres de fabrication complets (A-I-P)

4. Étude des couches locales et globales d'ORCA-FMS

Cette section présente les résultats des simulations ayant permis d'étudier le comportement des deux couches d'ORCA-FMS (ILP et champs de potentiels) et reprend donc les étapes de la macro-étape I de la figure 5.9.

4.1. Simulations S1

Produits A

Cette première série de simulations a pour objectif de comparer les résultats donnés par la couche de contrôle local qui utilisent le concept de champs de potentiel et la solution optimale donnée par un ILP. La résolution du modèle et la simulation sont tout d'abord lancées pour des campagnes allant de 1 produit A jusqu'à 10 produits A.

Pour rappel, dans notre cas d'étude, un produit 'A' nécessite un chargement (sur R1), la pose de 3 axes (sur R2 ou R3), ainsi que la pose d'une forme 'L' (sur R2 ou R4), d'une forme 'r'

(sur R2 ou R3), d'une forme 'I' (sur R4) et d'une vis (sur R3 ou R4), finalement il nécessite également une inspection (sur R5).

La figure 5.10a présente l'écart entre les deux approches au niveau du Makespan et la figure 5.10b présente cet écart en pourcentage.

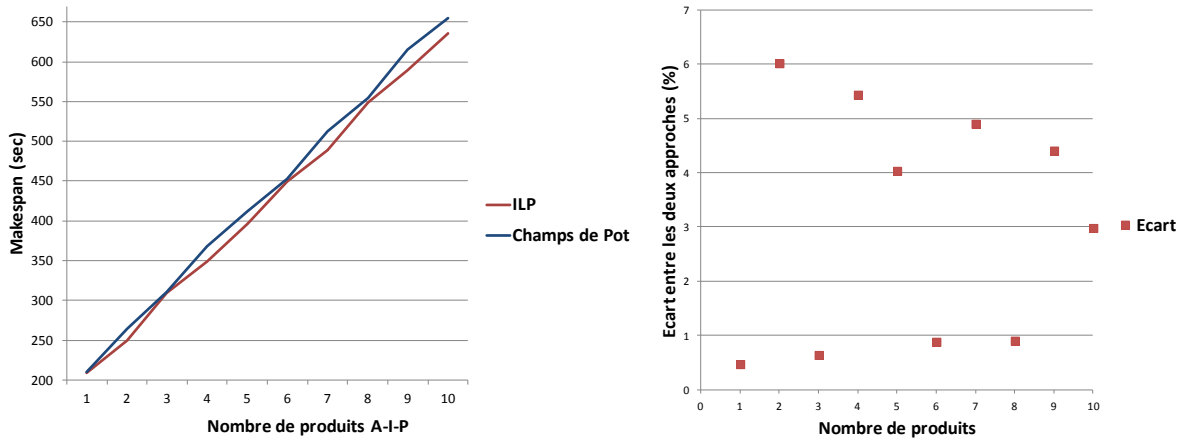


Figure 5.10a et 5.10b– Comparaison ILP-Champs de Potentiel Simulation (S1, A)

Les résultats ici montrent deux tendances. Pour une partie des simulations (pour 1, 3, 6 et 8 produits A), les résultats issus de l'approche par champs de potentiel sont très proches de l'optimal (ILP). Le comportement des champs de potentiel est très correct, l'écart étant inférieur à 1%. Par contre, pour le reste des simulations des problèmes de myopie apparaissent et l'écart se situe entre 3 et 6% pour les raisons données ci-dessous.

Pour la campagne avec 2 produits A, le phénomène de myopie apparaît dès le départ. La figure 5.11 présente le diagramme de GANTT obtenu par la méthode champs de potentiel alors que la figure 5.12 présente celui obtenu en suivant l'ILP.

Les opérations « Montage_Axe » et « Montage_ r» peuvent être réalisées sur deux ressources R_2 et R_3. En utilisant le principe des champs de potentiels présenté précédemment, le premier produit va donc sur R_2 (car plus proche) et le second sur R3. Les 3 « axes » et le « r » sont assemblés pour chacun des produits sur ces 2 ressources. Puis un « L » est nécessaire ; la machine R_2 peut effectuer cette opération mais pas R_3 ; le deuxième produit se dirige donc vers R_4 pour l'assemblage du « L » donc puis du « I ». Cette opération « Montage_I » n'étant disponible que sur R_4, le premier produit à s'y dirige également. Ainsi à cause de son opération supplémentaire sur R_2 le premier produit lancé se retrouve bloqué derrière le second au niveau de R_4 pendant que celui-ci effectue ses trois opérations. Au niveau de la solution optimale, l'ILP envoie le premier produit directement sur R_3 et le suivant sur R_2. Ainsi le premier produit arrive sur R_4 plus tôt, alors que le second est toujours en cours d'opération sur R_2. Lorsque celui-ci arrive sur R4 il lui reste une opération à attendre contre deux dans le cas des champs de potentiel. Au final, la campagne de production se terminera 10 secondes plus tôt en suivant l'ordre de l'ILP qu'avec les champs de potentiel. Le problème est similaire pour la simulation avec 4 produits.

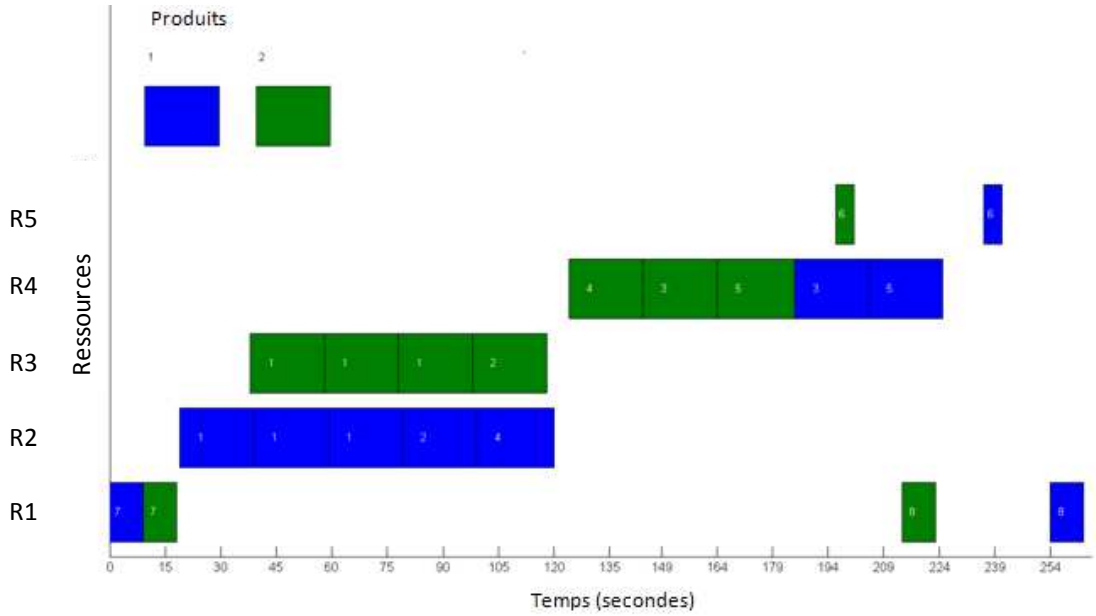


Figure 5.11– Diagramme de GANTT avec les Champs de Potentiel pour 2 produits A

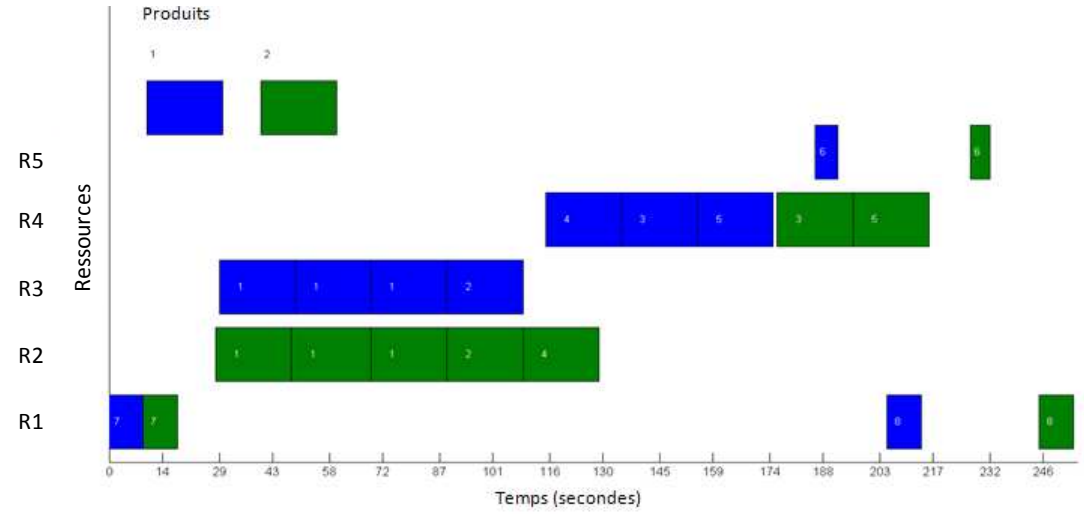


Figure 5.12– Diagramme de GANTT en suivant l'ILP pour 2 produits A

Une myopie assez semblable est responsable des différences pour les simulations à 5, 7, 9 et 10 produits A. En effet, les champs de potentiel répartissent les produits équitablement sur les ressources R_2 et R_3 selon une règle FIFO. Hors comme R_2 effectue une opération de plus, les derniers produits se retrouvent bloqués dans la file d'attente de R_2 alors que R_3 s'est déjà libérée. Au final cela cause des retards de 3 à 6 % en termes de Makespan par rapport à la référence calculée par l'ILP.

Pour ce type de produit les simulations ont montré deux choses. Tout d'abord, les champs de potentiel seuls fournissent des résultats avec retard par rapport à l'optimal ne dépassant pas 6 %. Il existe donc un écart entre cette solution et l'optimale mais cette approche peut être utilisée pour la couche d'optimisation locale avec des performances acceptables dans notre cas d'étude pour les produits A. Cependant, l'utilisation d'un optimiseur global pourrait permettre tout de même d'améliorer la performance globale en termes de Makespan (ici gain maximum espéré : 6%).

Produits I

Cette seconde série de simulations a pour objectif de comparer les résultats des deux approches pour des produits de type I en suivant le même protocole que précédemment. La figure 5.13a présente l'écart entre les deux approches au niveau du *Makespan* et la figure 5.13b présente cet écart en pourcentage.

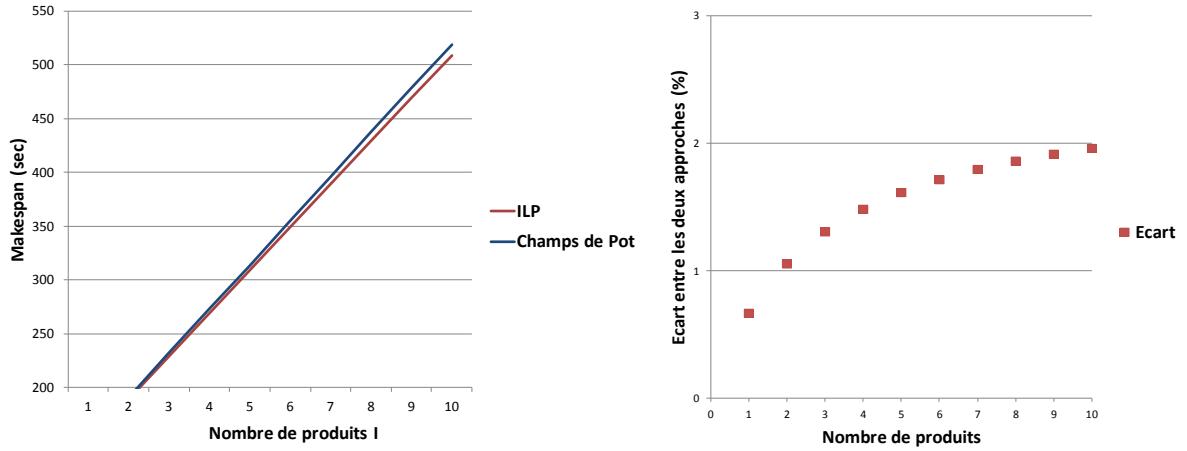


Figure 5.13a et 5.13b– Comparaison ILP-Champs de Potentiel Simulation (S1, I)

Il est constaté grâce à ces deux figures que l'écart entre la solution optimale fournie par l'ILP et l'approche par champs de potentiel est relativement faible (< 2%). Le comportement de l'approche par champs de potentiel pour le produit I dans notre cas d'étude est donc très correct. De plus, les deux modèles sont implémentés sur des outils différents (Cplex / NetLogo). L'écart qui apparaît ici peut donc être ramené à une très légère différence de modèle et non pas à un réel défaut lié exclusivement à l'une ou l'autre approche.

Produits P

La campagne de simulation suivante présente les résultats pour les produits de type P. La figure 5.14a présente donc l'écart entre les deux approches au niveau du *Makespan* et la figure 5.14b, celui en pourcentage.

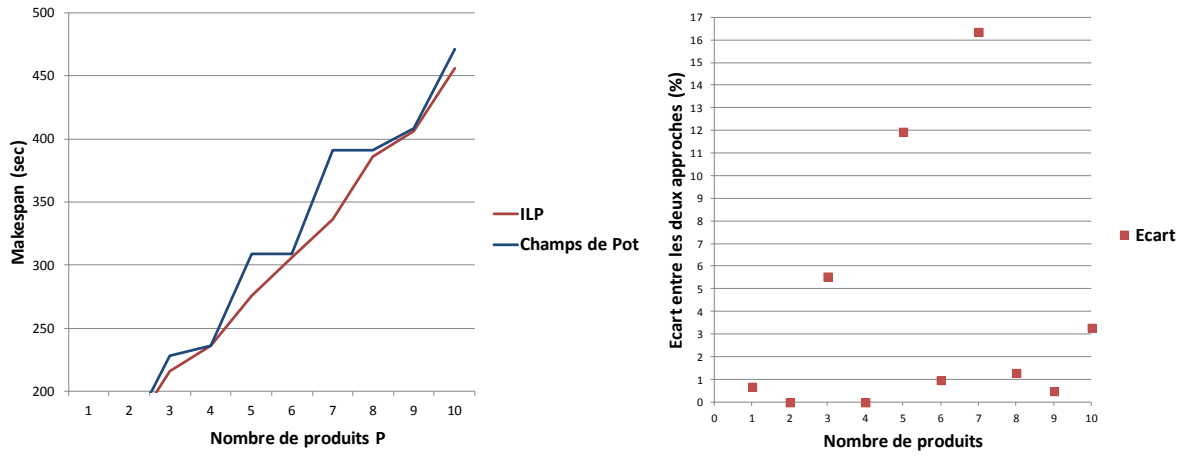


Figure 5.14a et 5.14b– Comparaison ILP-Champs de Potentiel Simulation (S1, P)

Les résultats pour le produit P sont assez semblables à ceux du produit A. Cependant des écarts plus importants sont visibles. Les causes sont les mêmes que pour le produit A avec un problème d'équilibrage entre la ressource R_2 qui effectue les 4 premières opérations et la ressource R_3 qui n'en effectue que 2. Cependant, le produit est plus simple (3 opérations en moins) donc un écart de 10 secondes a un impact plus important sur le temps de fabrication total. De plus, si un produit commence par R_2, il peut y réaliser l'ensemble des opérations d'assemblage alors que pour ceux qui commencent par R_3 doivent passer par R_4 ou retourner sur R_2. Ainsi le choix de la première ressource est primordial et l'approche FIFO des champs de potentiel amène des problèmes d'équilibrage encore plus importants.

Au final comme pour le produit A certaines campagnes donnent des résultats très proches de l'optimal alors que d'autres ont besoin d'être optimisées. Ces dernières simulations nous montrent encore un peu plus l'intérêt d'avoir un optimiseur global pour « aider » l'approche par champs de potentiel.

Produits A-I-P

Concernant les ordres de fabrications AIP, les simulations ont été limitées à 3 ordres pour plusieurs raisons. En effet, le cas d'étude sur lequel sont basées les simulations prévoit des scénarios jusqu'à 10 navettes maximum et passer à 4 ordres (A-I-P) nécessiterait 12 navettes. De plus, la simulation S2 présentée dans la partie suivante montre que le temps de résolution de l'ILP augmente de façon exponentielle avec le nombre de produits. Ainsi avec 4 A-I-P, il faut plusieurs heures pour obtenir ne serait-ce qu'une bonne solution. Finalement, l'arbre des solutions augmente lui aussi de façon exponentielle et à partir de 10 produits il arrive que le solveur du modèle doive s'arrêter à cause de limitations liées à l'ordinateur.

Ces simulations concernent donc les campagnes composées d'ordres A-I-P. Les résultats en termes de *Makespan* sont disponibles sur la figure 5.15a et les écarts en pourcentage sur la figure 5.15b.

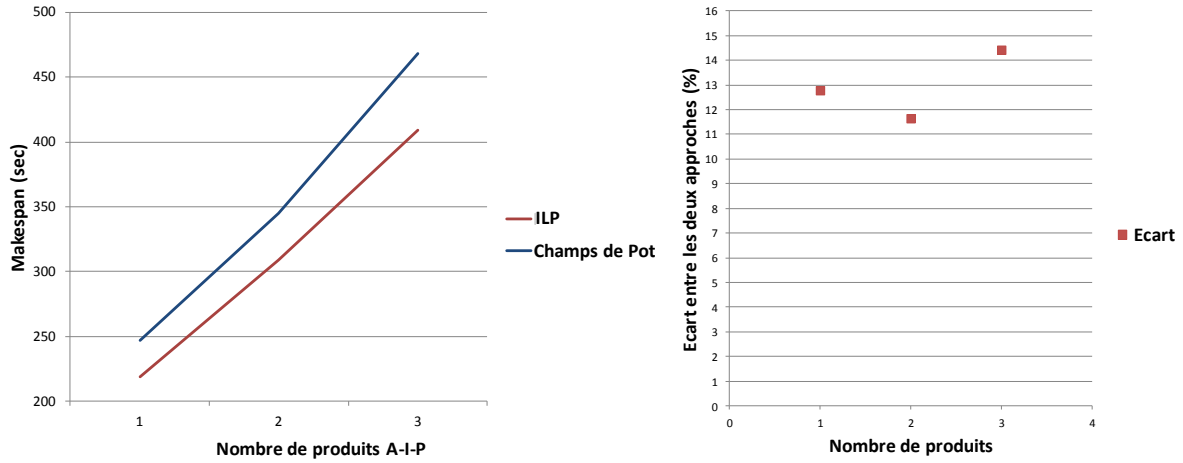


Figure 5.15a et 5.15b– Comparaison ILP-Champs de Potentiel Simulation (S1, A-I-P)

Pour chacune des trois simulations, les écarts entre les Champs de potentiel et l'optimal sont importants (entre 11 % et 15 %). Pourtant les causes ne sont pas les mêmes que pour les produits séparés. En effet, sur les diagrammes de GANTT de la solution pour 3 ordres A-I-P,

l'ordre d'introduction des produits (issu de l'ILP) est différent de l'ordre résultant des champs de potentiel. Les figures 5.16 et 5.17 présentent les résultats pour deux simulations pour le cas 3 A-I-P avec les champs de Potentiel. La première est une simulation avec comme séquence d'introduction des produits A-I-P-A-I-P-A-I-P, la seconde reprend l'ordre issu de l'ILP, I-A-I-A-A-P-P-I-P.

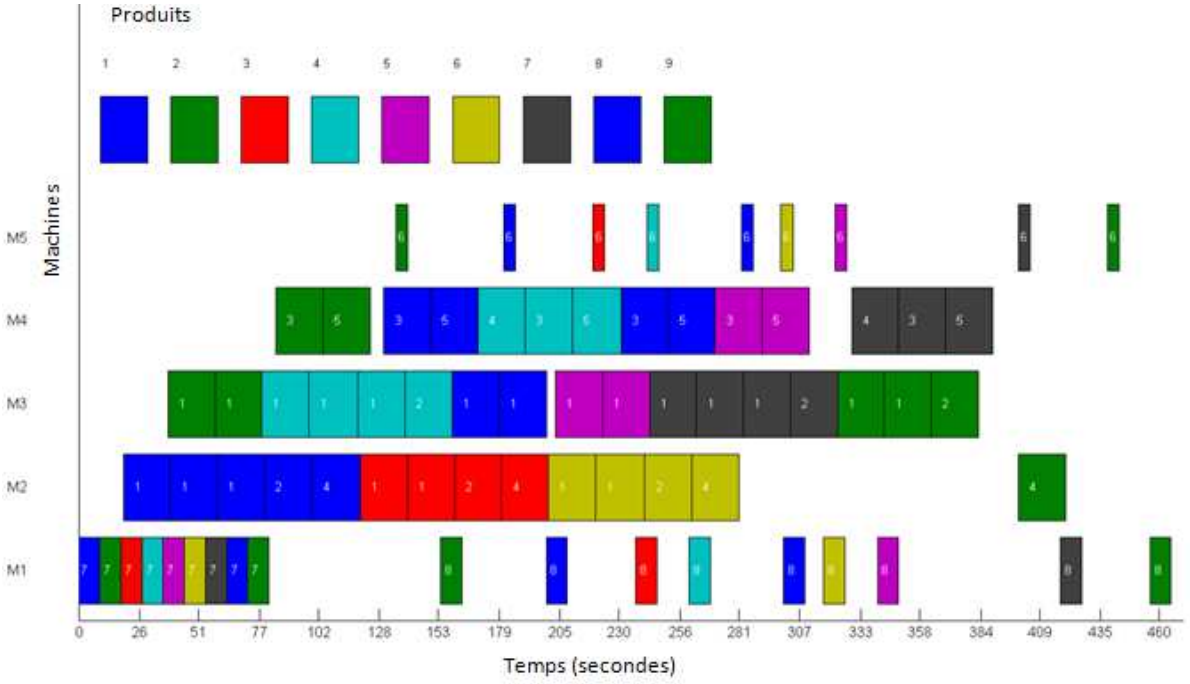


Figure 5.16– Diagramme de GANTT avec les champs de potentiel pour A-I-P-A-I-P-A-I-P

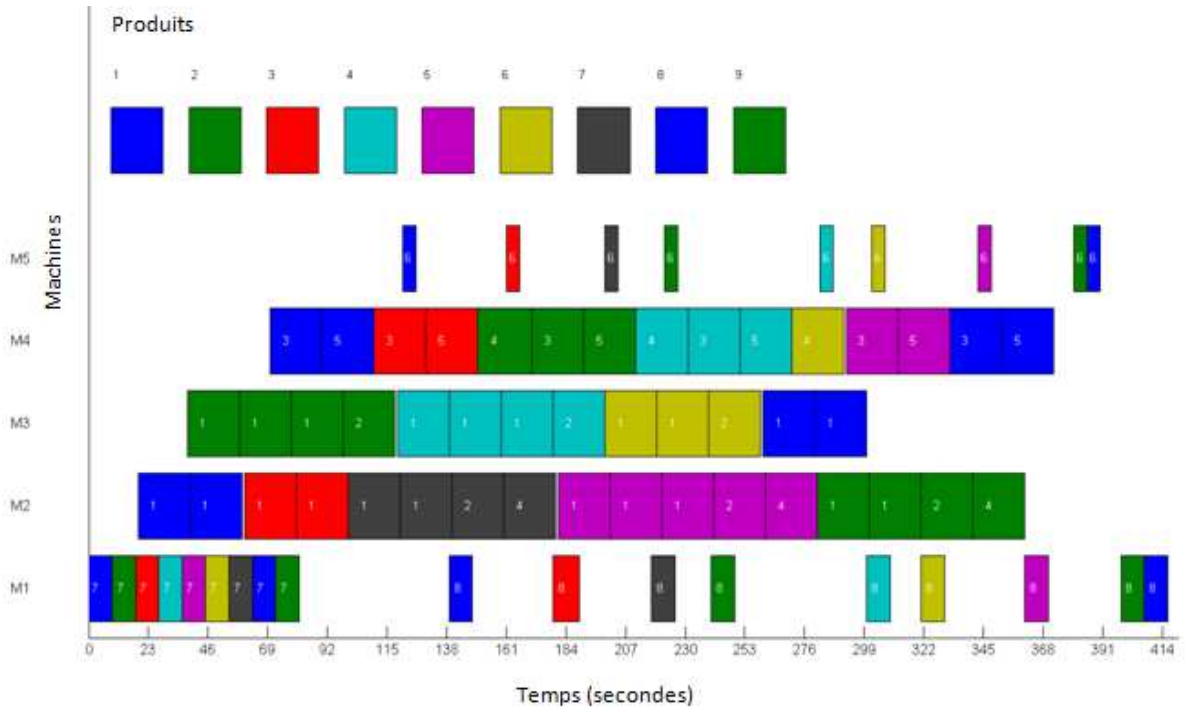


Figure 5.17– Diagramme de GANTT avec les champs de potentiel pour I-A-I-A-A-P-P-I-P

Ces figures montrent que les résultats obtenus pour les champs de potentiel peuvent différencier de 40 secondes en fonction de l'ordre d'introduction. Avec la seconde solution on se rapproche à 4% de l'optimal. L'écart restant provient comme lors des simulations précédentes d'un problème de myopie sur l'équilibrage des ressources et d'une différence de modèle. Ce dernier groupe de simulations a donc permis de mettre en évidence l'importance de l'introduction des produits dans le système. Ainsi dans les simulations S4 et S5 (présentées dans la suite de ce chapitre), un niveau intermédiaire d'hybridation (appelé ORCA-FMS 1) a été retenu. Dans ORCA-FMS 1, l'optimiseur global n'impose que l'ordre de lancement des produits à la couche de contrôle local alors que dans ORCA-FMS 2 la séquence complète des ressources est imposée. Ces simulations permettront de vérifier l'importance de l'ordre d'introduction.

4.2. Simulations S2

Ces simulations permettent d'étudier le temps de résolution de l'ILP. Le problème est considéré comme résolu lorsque l'intégralité de l'arbre des solutions a été parcourue et que la solution optimale (pour le modèle considéré) a été trouvée. Pour ces simulations le temps de résolution maximal a été fixé à 1 heure. En effet, le système doit rester réactif en se basant uniquement sur les résultats de l'ILP. Pour les scénarios étudiés, attendre plus d'une heure (qui est déjà une durée très excessive) pour une solution est inconcevable étant donné que le Makespan ne dépasse pas 10 minutes. Les simulations ont été lancées pour les mêmes campagnes d'essais que la Simulation S1 (1 à 10 produits A, 1 à 10 produits I, 1 à 10 produits P et finalement 1 à 3 produits A-I-P). L'ordinateur sur lequel l'ILP est lancé possède un processeur dual-Core i5 à 3.30 GHz et 6 Go de RAM. La version de Cplex utilisé est la 12.4. Les résultats sont visibles sur la figure 5.18.

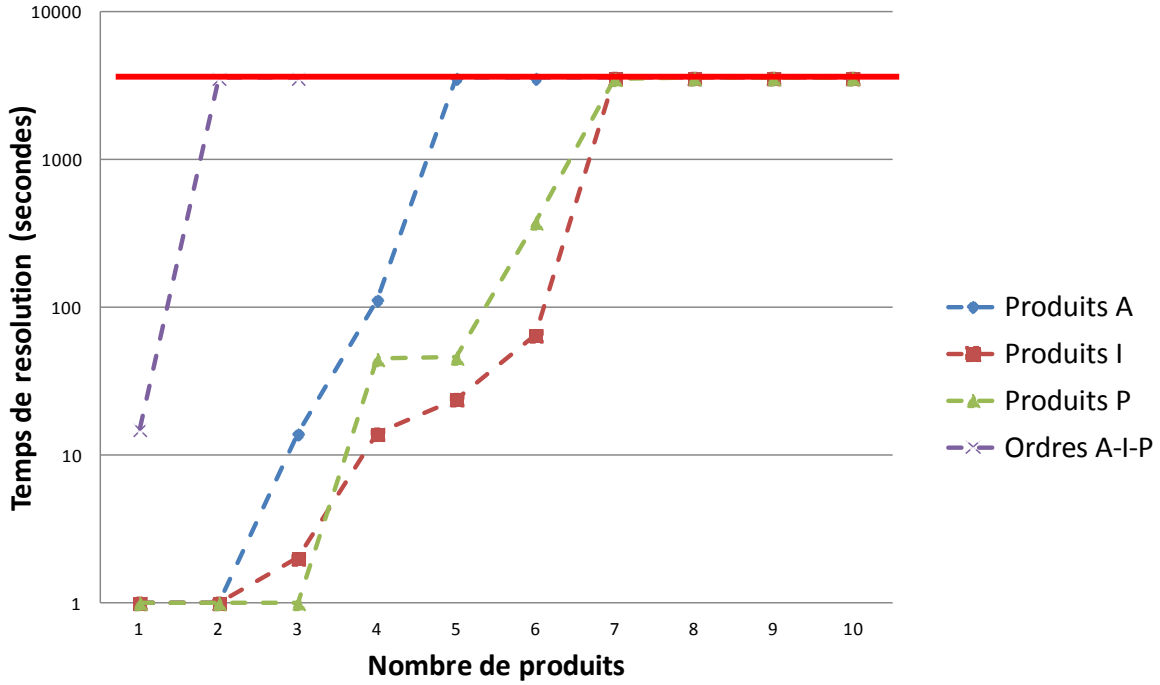


Figure 5.18– Temps de résolution de l'ILP (S2)

Pour les produits uniques (A, I et P) la limite de 1 heure est atteinte à partir de 5 éléments pour le produit A (produit le plus complexe) et à 7 éléments pour les deux autres qui sont plus simples. Si on regarde l'ordre A-I-P, une heure de résolution est nécessaire dès 2 ordres. Sur la base d'un temps de réparation d'une panne ressource de 15 minutes, il est donc inconcevable d'attendre que l'ILP recalcule l'optimal pour réagir à une perturbation qui sera surement réparée entre temps. Pour que la cellule puisse continuer à produire pendant la perturbation, il faut trouver une autre solution. Ces simulations ont donc été relancées avec pour objectif de trouver une solution « juste meilleure » que la méthode par champs de potentiel.

4.3. Simulations S3

Dans la simulation S3 l'ILP est configuré pour trouver un maximum de solutions rapidement même s'il ne peut pas en prouver l'optimalité. On espère ainsi diminuer le temps de calcul pour obtenir une meilleure réactivité du système. Les résultats obtenus pour trouver ces « bonnes solutions » sont présentées sur la figure 5.19.

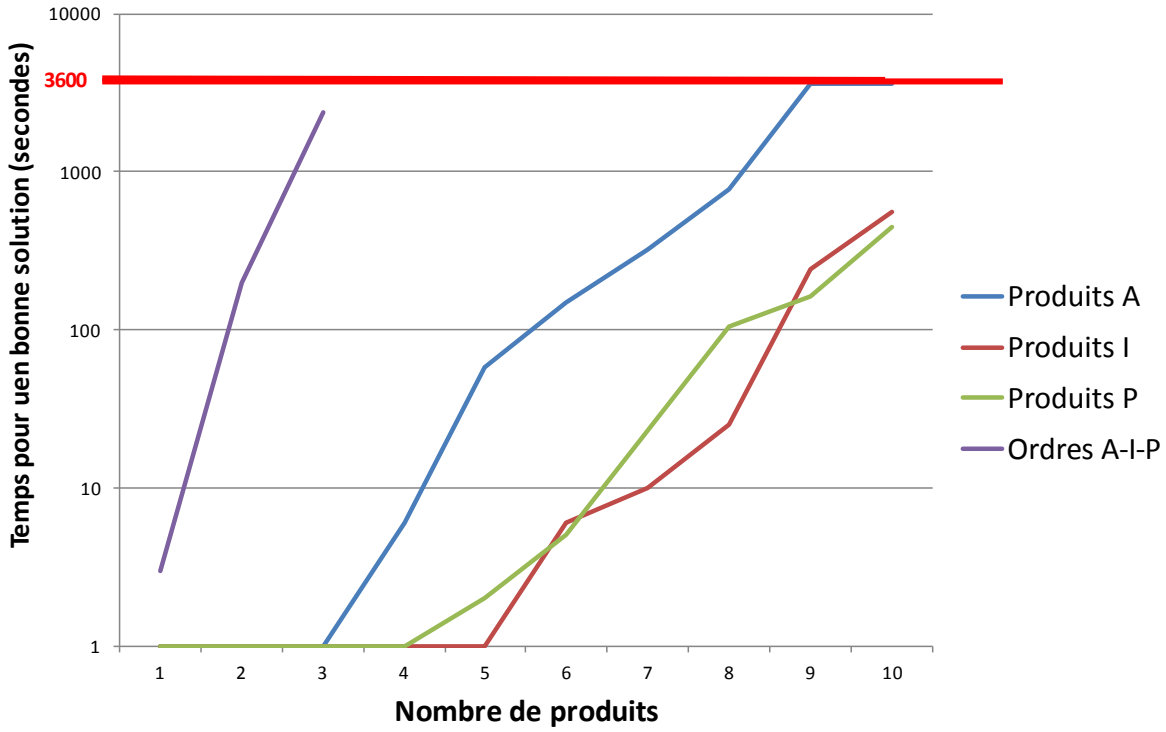


Figure 5.19– Temps pour trouver une bonne solution avec l'ILP (S3)

Cette figure montre que le temps pour trouver cette solution est bien inférieur à celui pour résoudre complètement le problème. Cependant lorsque le produit est complexe (comme le produit A) la limite des 3600 secondes est tout de même atteinte au bout de 9 produits. Pour l'ordre composé des trois produits A-I-P, l'heure de calcul est quasiment atteinte pour seulement 3 ordres. D'autant plus que ces cas restent simples avec des Makespan ne dépassant jamais 10 minutes. Pour un industriel une campagne de 10 minutes est quelque chose de minuscule par rapport à une campagne de 8 heures par exemple.

En plus de cela, si la simulation S1 a montré que dans le pire des cas les champs de potentiels mettent 60 secondes de plus que la solution optimale. Ici l'ILP dépasse souvent 1 minute de calcul. Ainsi si on utilise des champs de potentiel pour réagir à une panne, la production sera terminée avant même que l'ILP ait trouvé une meilleure solution (qu'il faut ensuite appliquer). Ces expérimentations nous montrent clairement, que l'ILP n'est pas capable de réagir rapidement à une perturbation au contraire de l'approche basée sur les champs de potentiel. D'autant plus que si le système subit une perturbation, son modèle (exploité par l'ILP) doit changer (l'état des ressources est différents, celui des produits aussi). Ainsi il faudrait régénérer (automatiquement ou manuellement) un autre modèle avant de pouvoir chercher une solution ce qui prendrait du temps et des ressources supplémentaires.

Ces simulations montrent donc qu'utiliser une approche réactive, par champs de potentiel, pour réagir aux perturbations permet de gagner en performance face à une approche composée uniquement d'un ILP. La simulation S1 avait montré qu'utiliser un ILP permettait d'obtenir de meilleures performances qu'une approche basée sur les champs de potentiel seuls en fonctionnement normal. Ainsi ces deux approches sont complémentaires. Leur association dans une architecture hybride comme ORCA-FMS (ILP en tant qu'optimisation globale et Champs de Potentiel en tant qu'optimisation locale) devrait donc nous permettre d'obtenir un gain de performances à la fois en fonctionnement normal et en fonctionnement perturbé. La section 5 étudie le comportement d'ORCA-FMS.

5. Étude du comportement d'ORCA-FMS

Ce groupe de simulations a pour objectif de valider le comportement de l'architecture ORCA-FMS en fonctionnement normal. Comme la simulation S1 l'avait démontré, les champs de potentiel ne donnent pas les résultats optimaux en fonctionnement normal. D'une part à cause de problème de myopie sur l'équilibrage entre les ressources et d'autre part à cause de l'ordre d'introduction des produits. Le fonctionnement de deux versions d'ORCA-FMS va donc être simulé :

- ORCA-FMS 1 dans laquelle l'optimisation globale fournit uniquement l'ordre d'introduction des produits à l'optimisation locale.
- ORCA-FMS 2 dans laquelle l'intégralité de la séquence optimale des ressources à visiter par le produit est fournie au niveau d'optimisation locale.

5.1. Simulations S4

Les simulations S4 sont lancées pour les ordres de fabrication A-I-P, 2x A-I-P et 3x A-I-P. La table 5.V donne les résultats des simulations.

Table 5.V– Résultats de la simulation S4

Campagne de Production	Optimal (sec)	Simulation (secondes)			Gain / Champs de Pot. (secondes)		Gain / Champs de Pot. (%)	
		Champs de Pot.	ORCA-FMS 1	ORCA-FMS 2	ORCA-FMS 1	ORCA-FMS 2	ORCA-FMS 1	ORCA-FMS 2
AIP	219	247	224	224	23	23	9,3	9,3
2 x AIP	309	345	323	314	22	31	6,4	9
3 x AIP	409	468	428	415	40	53	8,5	11,3

Cette table montre que l'architecture ORCA-FMS permet d'obtenir de meilleurs résultats en fonctionnement normal que les champs de potentiel seuls. Ainsi avec ORCA-FMS 1, l'amélioration des résultats est de 8% en moyenne par rapport à une approche par Champs de Potentiel. L'ordre de production est donc effectivement important et ce premier niveau d'hybridation permet d'améliorer les performances. Le deuxième niveau d'hybridation ORCA-FMS 2 permet de se rapprocher un peu plus de l'optimal en améliorant les performances de 10% en moyenne. L'écart est réduit à 1.8% (en moyenne) par rapport à l'optimal ce qui est vraiment très faible. Pour le scénario avec la campagne A-I-P, les résultats donnés par les deux versions d'ORCA-FMS sont identiques puisque le scénario étant simple, la seule myopie qui apparaît concerne l'ordre de lancement des produits.

Ces simulations nous poussent à penser qu'ORCA-FMS semble remplir ses objectifs en fonctionnement normal, à savoir améliorer les performances de notre approche d'optimisation locale et atteindre ou se rapprocher grandement du comportement optimal pour le modèle définit.

5.2. Simulations S5

La dernière simulation permet d'étudier le comportement d'ORCA-FMS lors d'une perturbation. La ressource R_2 sera dans ce scénario « réservée pour maintenance » suite à un défaut détecté à t=100 secondes (cela correspond au scenario PS9 du benchmark (Trentesaux et al., 2013)). Aucun produit ne pourra lui être affecté après cette date.

Lors des simulations S2 et S3 il a été montré qu'un ILP seul ne peut pas réagir correctement à une perturbation. Ainsi ORCA-FMS doit permettre de corriger ce problème en profitant du niveau d'optimisation locale basé sur les champs de potentiel. Pour vérifier cette hypothèse, cette simulation compare les résultats obtenus par ORCA-FMS 1 et ORCA-FMS 2 avec ceux de l'approche par champs de potentiel seule.

L'objectif était aussi de pouvoir comparer ces résultats à la production la plus rapide considérant cette perturbation. Cependant, un programme linéaire ne permet pas de gérer en dynamique des perturbations et d'obtenir cette solution optimale en termes de *Makespan*. Une contrainte spécifique a donc été ajoutée à l'ILP l'empêchant d'allouer des produits à la ressource R_2 après t= 100 secondes. La perturbation est donc artificiellement connue à l'avance dans l'ILP ce qui permet d'avoir une borne pour le temps de production minimum même si en réalité il est bien évidemment impossible de prévoir une perturbation. Cette solution est appelée ILP* dans les résultats présentés dans la table 5.VI.

Table 5.VI– Résultats de la simulation S5

Campagne de Production	ILP*	Simulation (secondes)			Ecart / ILP* (%)			Gain /champs de pot. (%)	
		Champs de Pot.	ORCA-FMS 1	ORCA-FMS 2	Champs de Pot.	ORCA-FMS1	ORCA-FMS2	ORCA-FMS1	ORCA-FMS2
2 x AIP - M2 en maintenance (t=100)	376	405	388	388	7.7	3.2	3.2	4.2	4.2

Ces résultats montrent que l'architecture ORCA-FMS réagit d'une bonne manière à la perturbation. De plus, avant cette perturbation, la production est optimisée globalement (puisque basée sur l'ILP). Ainsi les résultats obtenus sont meilleurs qu'avec les champs de potentiel seuls en régime perturbé (amélioration de 4.2 %). Il n'y a pas de différence entre les deux versions d'ORCA-FMS puisque la panne arrive tôt et oblige donc les systèmes à passer très tôt en réactif. Les résultats obtenus se situent à 3% de la borne inférieure. Encore une fois cette solution optimale est théorique puisque la perturbation était connue par l'ILP* avant le lancement de la production ce qui ne sera jamais le cas en réalité. La figure 5.20 illustre ces résultats en positionnant sur une échelle de temps les résultats fournis par :

- l'ILP* qui correspondant à l'optimal pour le modèle considéré (en connaissant la perturbation à l'avance),

- ORCA-FMS, qui permet le basculement entre modes autonome (allocation et routage par champs de potentiel) et exécutant (allocation qui suit les recommandations de l'ILP et routage selon les champs de potentiel)

- Les champs de potentiel seuls qui correspondent au fonctionnement d'ORCA-FMS en mode autonome

- L'ILP seul qui correspond au fonctionnement d'ORCA-MS en mode exécutant. Le temps de re-calcul de l'ILP pour ce cas d'étude est supérieur à 3600 secondes

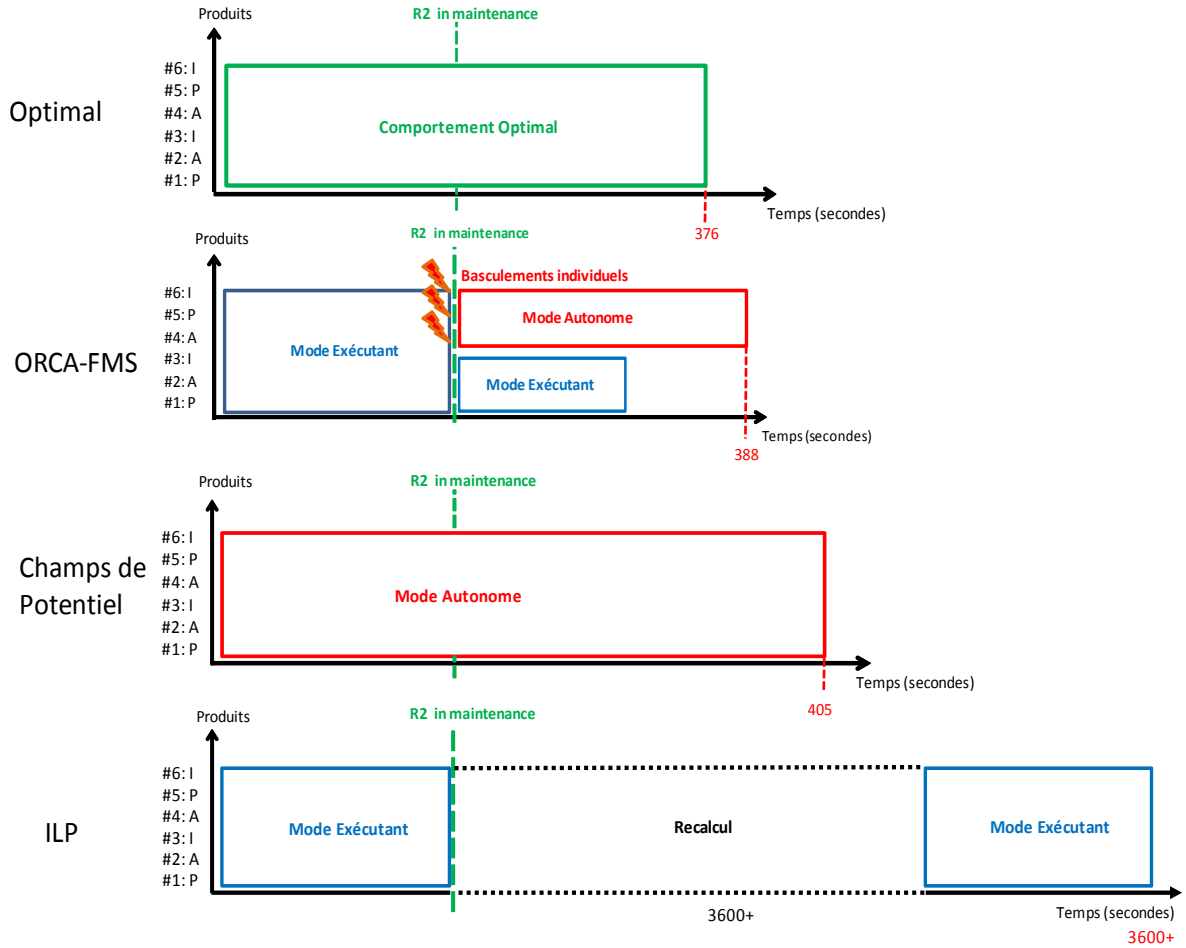


Figure 5.20– Comportement optimal, d'ORCA-FMS et de ses deux couches isolées

En considérant les différences de modèles, nous estimons que les résultats obtenus par ORCA-FMS sur ce cas d'étude sont très encourageants. ORCA-FMS semble ainsi capable de réagir aux perturbations tout en assurant une bonne performance, dans le cadre de l'étude présente.

Des simulations annexes ont été effectuées avec l'intégration de la consommation énergétique. L'objectif était d'étudier comment de nouvelles contraintes pouvait être intégrées dans ORCA-FMS conformément au contexte industriel changeant des systèmes de production. L'**annexe 9** détaille ces simulations supplémentaires. Ces simulations, qui exploitent la possibilité pour les ressources de réduire de manière opportuniste leur consommation si elles ne sont pas utilisées, ont permis de montrer que, dans notre cas d'étude, la prise en compte de l'énergie dans la couche locale d'ORCA-FMS pouvait permettre de gagner entre 15 et 40% d'énergie selon la charge du système sans perdre de performances notable en termes de makespan.

6. Synthèse

Le protocole expérimental présenté section 3.3 a permis de tester l'architecture via deux séries de simulations. Une première série de simulations (macro étape I de la figure 5.9) a permis de souligner les avantages de l'optimisation globale en fonctionnement normal (S1) et ses limites en cas de re-calcul (suite à une perturbation par exemple) par rapport à une optimisation locale (réactivité de l'ordre d'1 heure (S2 et S3) contre 1 seconde pour l'optimisation locale). L'intérêt d'une approche hybride mixant ces deux types d'optimisations a donc été mis en évidence. La deuxième série de simulations (macro étape II de la figure 5.9) a permis de confirmer cet intérêt et de montrer qu'ORCA-FMS bénéficiait des avantages des deux optimisations en améliorant à la fois les performances en termes de performances globales (S4, S5) et de réactivité (S5).

7. Conclusion

Ce chapitre a permis de valider l'architecture ORCA-FMS en simulation sur un cas d'étude. Les premières parties de ce chapitre ont permis de présenter le cas d'étude utilisé, ainsi que le modèle de simulation et le modèle linéaire associés. Le protocole expérimental a ensuite été décrit. Puis les résultats de simulations ont été présentés avec premièrement une étude des couches locales et globales séparées, puis une étude de l'architecture ORCA-FMS complète.

Le comportement de l'architecture a pu être validé sur ce cas d'étude en simulation. Il reste désormais à montrer que cette approche peut être réellement mise en œuvre et que les résultats expérimentaux sur la cellule réelle confirment ceux obtenus en simulation.

Le chapitre VI présente la mise en œuvre réelle de tous les concepts présentés, le protocole expérimental utilisé sur la cellule réelle du cas d'étude et les résultats obtenus.

Chapitre VI : Mise en œuvre réelle d'ORCA-FMS

1. Introduction

Le chapitre précédent a montré l'intérêt de l'architecture ORCA-FMS en simulation. Ce chapitre VI décrit une mise en œuvre effective de l'architecture ORCA-FMS sur la cellule flexible de l'AIP PRIMECA de Valenciennes. Il faut désormais prouver que ce type d'architecture est réalisable avec les technologies actuelles et confirmer les résultats préliminaires (obtenus en simulation) par des résultats expérimentaux (obtenus sur une cellule réelle). Ce chapitre présente tout d'abord la cellule réelle ainsi que les éléments technologiques ayant permis de mettre en œuvre ORCA-FMS. Ensuite la mise en œuvre des entités (ressources et produits) et concepts (approche par champs de potentiel et ILP) est présentée. Finalement un protocole expérimental est détaillé et les résultats obtenus sont présentés, puis analysés.

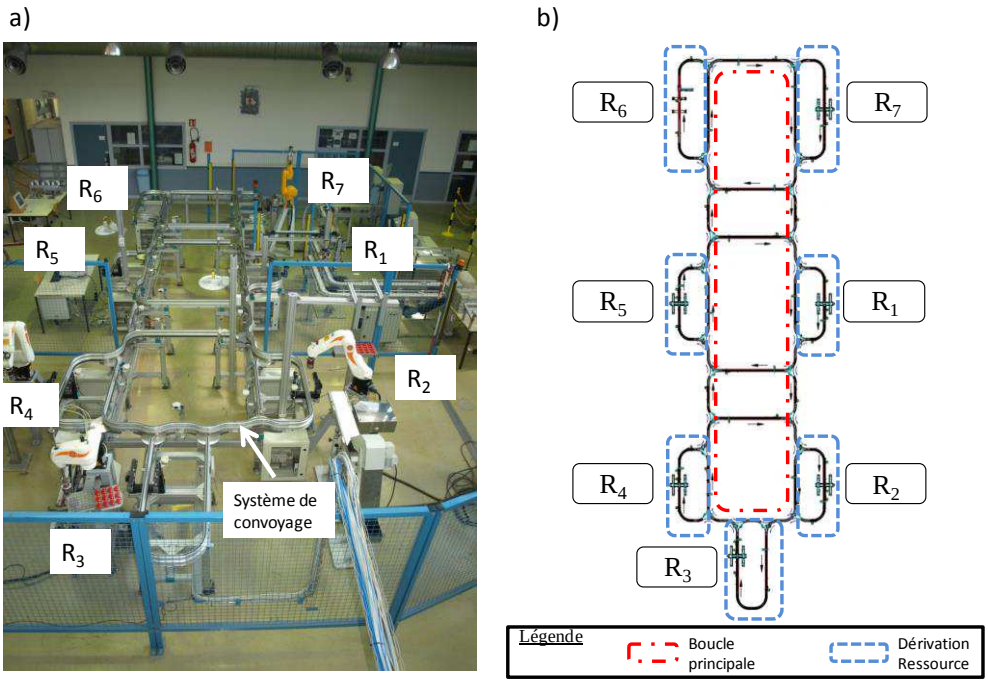
2. Présentation de la cellule flexible AIP PRIMECA

Ce chapitre reprend le cas d'étude du chapitre précédent et présente la cellule flexible en mettant l'accent, cette fois-ci, sur les technologies et les sous-systèmes physiques utilisés.

2.1. Le système de convoyage

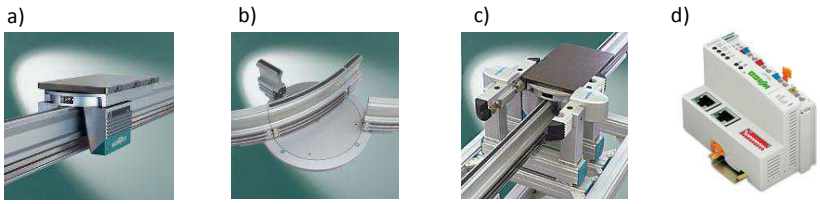
Le système de convoyage est constitué d'un monorail unidirectionnel (Montratec AG, 2013). La cellule est composée d'une boucle principale et de dérivations permettant d'atteindre les ressources. La figure 6.1a présente la cellule réelle et la figure 6.1b sa version schématisée.

Les navettes (Montratec AG, 2013) circulant sur ce système de convoyage sont autopropulsées. Afin d'éviter les collisions un système infrarouge est placé à l'avant de celles-ci. Ce système arrête automatiquement la navette si elle se rapproche trop de la navette la précédant sur le système de convoyage. La figure 6.2a représente une navette de transport sur le système monorail de convoyage. Pour permettre aux navettes d'atteindre les ressources en changeant de monorail, des aiguillages rotatifs sont utilisés. Avant chaque aiguillage un dispositif permet aux navettes de s'arrêter. Une fois l'aiguillage correctement positionné, un ordre est envoyé à la navette pour qu'elle reparte. Un aiguillage est représenté figure 6.2b. Au niveau du poste de travail de chaque ressource un système d'indexage permet de brider la navette pendant que la machine réalise une opération. La figure 6.2c donne une vue d'un système d'indexage.



Figures 6.1a et 6.1b– Cellule flexible réelle et schématisée

Les aiguillages et les systèmes d'indexage sont contrôlés par des automates programmables distribués sur la cellule. Ces automates (Wago, 2013) servent de passerelles entre les demandes provenant des produits ou ressources et le système physique relié sur les entrées / sorties directes de l'automate concerné. Leur petite taille, leur coût réduit et leurs fonctionnalités réseau intégrées en font de parfaits automates pour les systèmes distribués d'automatisation. Sur la cellule il y a au total 18 automates Wago. La figure 6.2d présente un de ces automates.



Figures 6.2a, 6.2b, 6.2c et 6.2d– Navette, aiguillage, système d'indexage et automate.

2.2. Les ressources

Le poste de chargement / déchargement (R_1) est mis en œuvre par un bras manipulateur. Les postes d'assemblages R_2 et R_4 sont matérialisés par des robots 6 axes (Kuka Roboter GMbH, 2013) alors que R_3 est un robot 4 axes (Kuka Roboter GMbH, 2013). L'inspection automatisée (poste R5) est réalisée par un automate de vision (Cognex Corporation, 2013). Le poste R_6 de reprise des produits présentant un défaut de fabrication est constitué d'un système d'indexage (au niveau duquel un opérateur humain peut remédier au défaut) et finalement la ressource optionnelle R_7 est un bras robotisé (Staubli, 2007). La figure 6.3 présente les différents types de ressources.

Figure 6.3– Chargement, Robot 6 axes, Poste de Vision, Robot 4 axes et Bras robotisé

La capacité des ressources R_1, R_2, R_3, R_4, R_5, R_6 et R_7 est d'un produit (réalisation d'une opération pour un produit) avec une file d'attente d'un produit supplémentaire juste avant la ressource.

3. Mise en œuvre des entités

Cette section présente la mise en œuvre réelle des entités ressources et produits d'ORCA-FMS ainsi que la mise en œuvre des communications entre ces entités.

3.1. Mise en œuvre de l'entité ressource

Afin de gérer l'état dans lequel se trouve la ressource (veille, travail…) et l'émission/réception des champs de potentiel, le concept de ressource active est proposé pour mettre en œuvre l'entité « ressource ». Ce concept est détaillé dans (Pach et al., 2013a) et basé sur celui du produit actif présenté dans la section 3.2. Une ressource active contient la ressource (par exemple le robot), son stock de matière première, sa file d'attente et l'API (Automate Programmable Industriel) qui lui fournit une capacité de traitement. La figure 6.4 présente la projection de la vue holonique sur la vue organique de la ressource active.

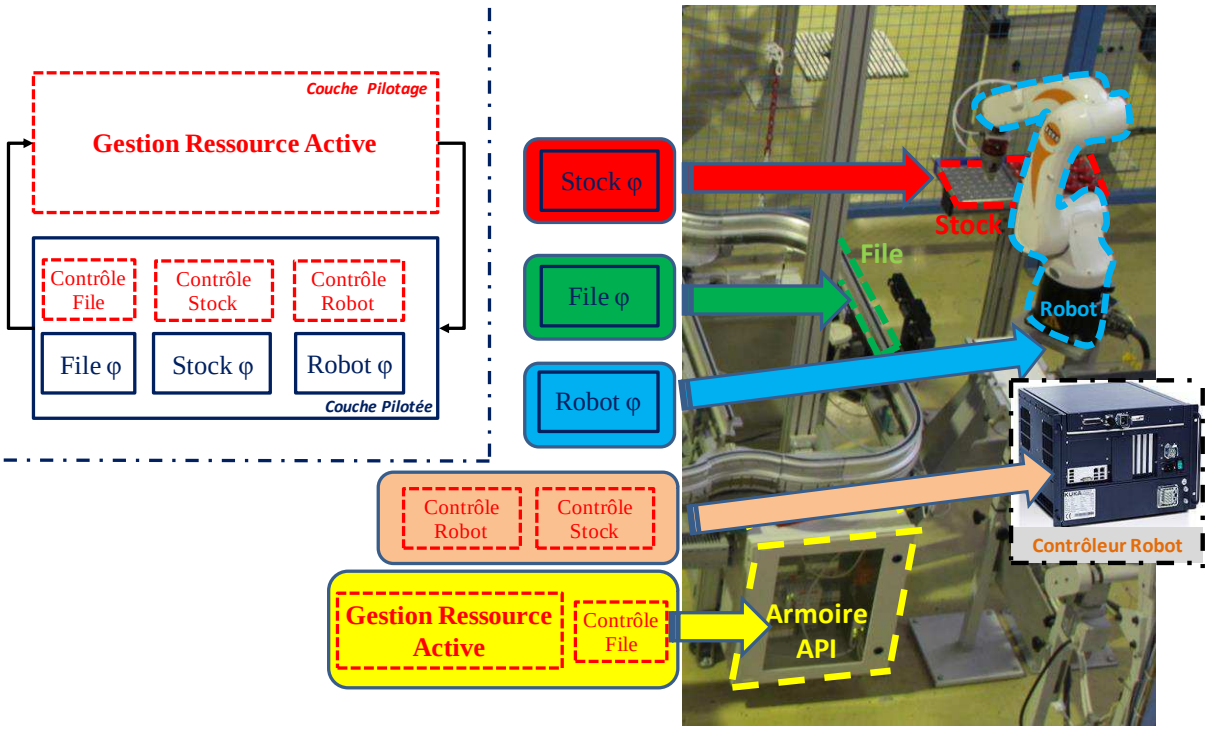


Figure 6.4– Projection de la vue holonique sur la vue organique de la ressource active

Cette ressource active peut être représentée par le holon présenté figure 6.4. La couche pilotée contient tout d'abord les deux quasi-holons (notés holons*) stock et file. Ce sont des holons du niveau le plus bas contenant les parties physiques du stock et de la file d'attente ainsi que le contrôle de bas niveau associé à ces deux éléments. La couché pilotée contient aussi le holon robot avec la partie physique du robot et le contrôle de bas niveau associé.

La couche Pilotage contient quant à elle les traitements permettant de gérer la ressource active au regard du pilotage du système. Il s'agit par exemple de générer, stocker, mettre à jour et propager les champs de potentiel ou de mettre à jour l'état de la machine en fonction des champs de potentiel reçus. Cette couche est dans notre cas intégralement supportée par l'API.

La figure 6.5 illustre le comportement d'une ressource active en reprenant la partie du programme en SFC (*Sequential Function Chart*) qui ère le basculement entre les états « veille » et « prête » de la ressource. Les états correspondant et les transitions de l'algorithme de gestion d'état (chapitre IV) sont rappelés.

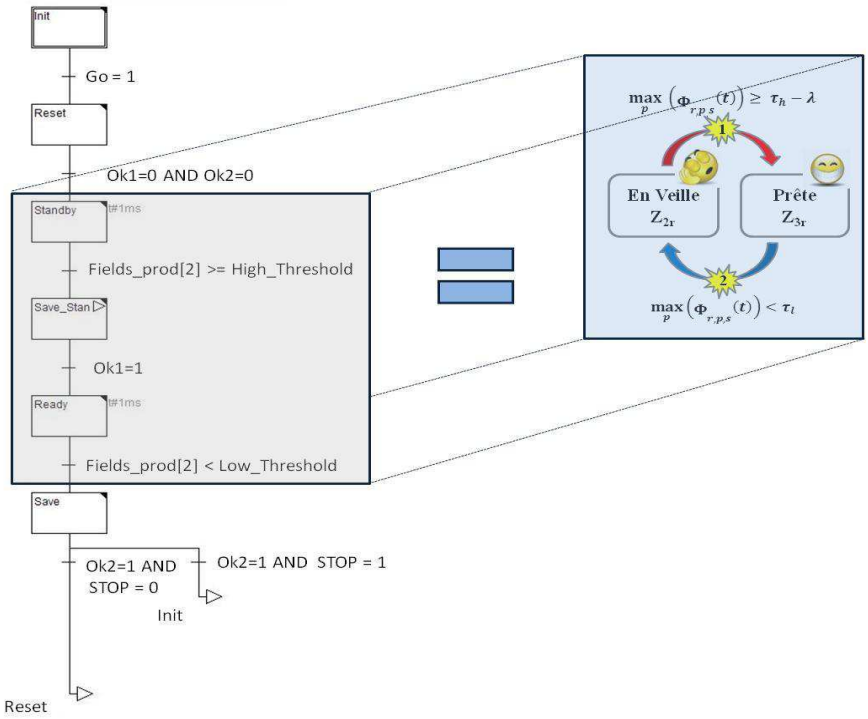


Figure 6.5– Comportement de la ressource active pour le basculement entre états

3.2. Mise en œuvre de l'entité produit

Les entités « produit » étant mis en œuvre par le concept de produit « actif », dans le cadre d'ORCA-FMS, cette cellule flexible relève selon le terme consacré (Sallez et Bajic, 2007), d'un Système Contrôlé par le Produit (SCP). Dans les SCP, le produit est « intelligent » (Kärkkäinen et al., 2003)(Wong et al., 2002) ou « actif » (Sallez et al., 2010). Un produit est actif dans le sens où il possède des capacités de traitement de l'information, de stockage, de communication et de prise de décisions. Chaque produit actif prend donc l'intégralité des décisions d'allocation (si nécessaire) et de routage jusqu'aux ressources qu'il doit atteindre. Afin de prendre ses décisions, le produit actif a besoin de connaître sa localisation sur la cellule. Ainsi chaque navette embarque un tag RFID (*Radio Frequency Identification)* qui est lu (lecture sur une courte distance, i.e. environ 1 cm) à chaque fois qu'il passe devant un lecteur RFID dont la localisation est précisément connue. La technologie RFID est donc ici utilisée de façon originale (un tag lu par plusieurs lecteurs au lieu de l'inverse) afin de permettre au produit de se localiser en fonction de la position du dernier lecteur devant lequel il est passé.

Le produit actif de cette cellule est un holon composé de trois éléments :

- Le produit passif qui est assemblé/fabriqué dans la cellule,
- Une navette (intégrant un tag RFID) lui permettant de se déplacer dans la cellule),
- Un Eeepc (ASUSTeK, 2012) embarqué sur la navette. Cet Eeepc est un ordinateur portable de 11 pouces, possédant 2 Go de RAM et un processeur Intel Atom de 1.5 GHz (capacité de traitement et de décision), une communication WiFi (*Wireless Fidelity* ; (capacité de communication) et un disque dur SSD (capacité de stockage) résistant aux chocs dus aux déplacements dans la cellule. À noter que cet Eeepc, utilisé dans ces travaux de recherche et offrant beaucoup de flexibilité au niveau des déve-

loppements logiciels, serait remplacé par un système embarqué plus compact et définitif pour un produit actif industriel finalisé.

La figure 6.6 représente la projection de la vue holonique sur la vue organique du produit actif.

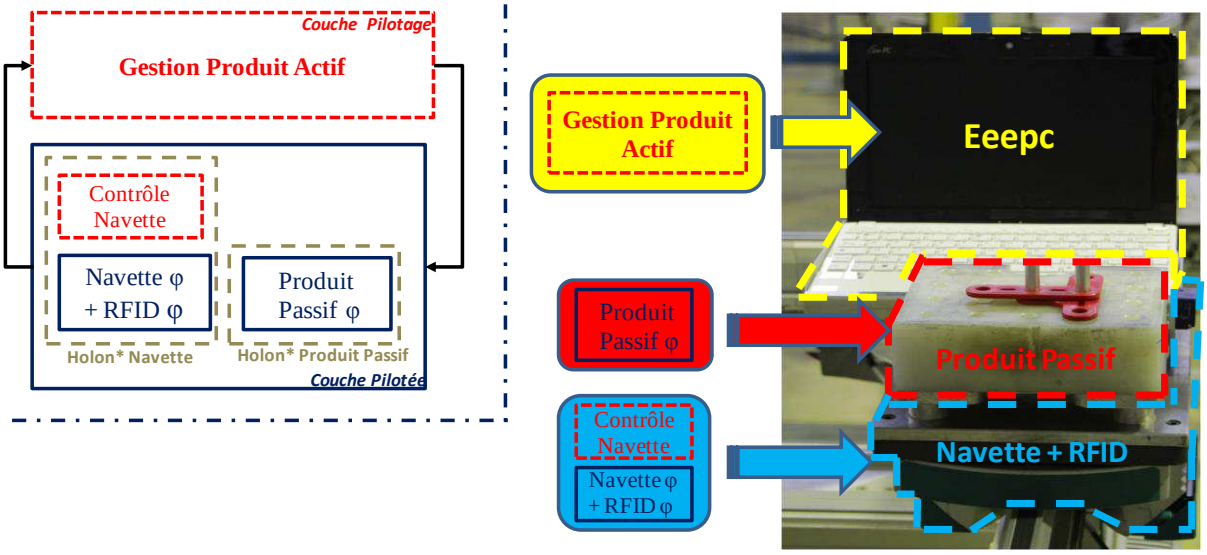


Figure 6.6– Projection de la vue holonique sur la vue organique du produit actif

Le produit actif peut être modélisé via le holon présenté figure 6.6. La couche pilotée contient les deux quasi holons (holons sans autonomie noté holons*) Navette et Produit Passif. Le holon* Produit Passif correspond au produit physique fabriqué. Le holon* Navette contient d'une part les éléments physique de la navette (dont le tag RFID) et d'autre part le contrôle de bas niveau associé qui correspond à la gestion des mouvements élémentaires de la navette (avancer navette, arrêter navette…).

La couche Pilotage contient quant à elle les traitements et données permettant de prendre les décisions de pilotage : l'allocation et le routage, les champs de potentiel captés, la ressource choisie, ou encore la typologie du système de transport. Ces traitements et données sont supportés par l'Eeepc.

La programmation du comportement du produit actif a été réalisée en Java. Ce langage objet est en effet un langage très utilisé par la communauté informatique, il est multiplateforme et de très nombreuses bibliothèques facilitent son utilisation. Par exemple, des bibliothèques de communication Modbus/TCP existent et permettent ainsi de communiquer très facilement avec les automates Wago qui utilisent ce protocole. L'environnement utilisé pour la programmation est Netbeans (Oracle, 2012). Cet environnement illustré figure 6.7 a été choisi car :

- Il est gratuit.
- Il prend peu de ressources et peut donc être installé sur un Eeepc (pour faire du debug).
- Son interface est conviviale et sa prise en main est rapide.
- Il intègre un créateur d'interface graphique.

- Il intègre facilement la gestion de la java-doc (documentation des méthodes utilisateurs).

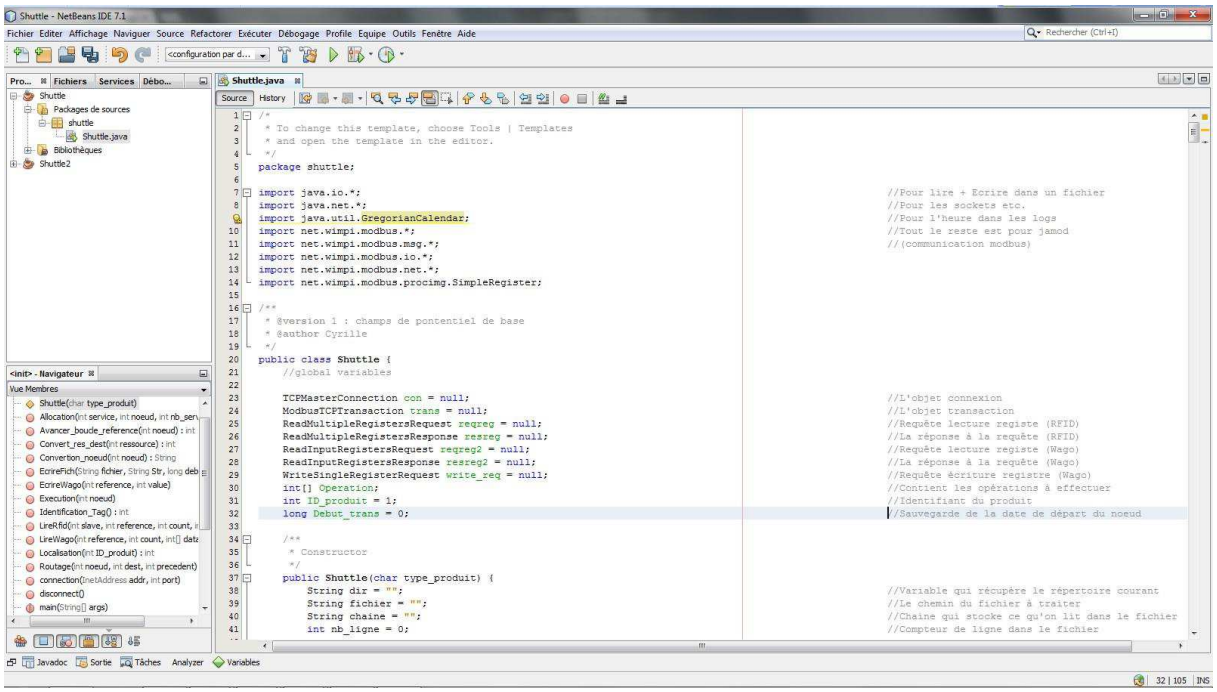


Figure 6.7– Environnement Netbeans

Le comportement de notre produit actif est très proche de celui d'un agent, même s'il n'est pas programmé sur une plateforme multi-agent. Chaque produit actif dispose de son propre programme qui communique avec les autres entités du système : ressources et nœuds décisionnels qui peuvent eux aussi être vus comme des agents. Le passage sur une plateforme agent n'a pas été réalisé car il demandait une adaptation de l'architecture du réseau et de certains composants. Des agents auraient dû être développés pour tous les automates de la cellule qui ne peuvent pas s'intégrer dans l'état dans une plateforme agent. Des agents auraient aussi dû être développés pour communiquer avec les lecteurs RFID. Cependant, l' « agentification » de notre architecture (en cours d'analyse) ne doit pas être écartée et constitue une perspective intéressante.

Lorsqu'un produit actif entre dans la cellule il reçoit la gamme de fabrication concernant le produit à fabriquer. Cet ordre est représenté figure 6.8. La première ligne correspond au type de produit et les lignes suivantes aux opérations à réaliser. Ainsi il est très facile de faire évoluer la gamme de fabrication d'un type de produit ou d'ajouter de nouveaux produits sans modifier le comportement du produit actif.

Figure 6.8– Ordre de fabrication d'un produit A

Le comportement du produit actif représenté sur la figure 6.9 suit les étapes suivantes :

1) Le produit actif s'initialise avec les informations nécessaires telles que le produit passif à réaliser, sa gamme de fabrication ou l'identifiant du tag RFID de la navette à laquelle il est associé. Une fois le produit actif initialisé il entre dans le cycle principal composé des étapes 2–3–4–5.

2) Dans cette étape du cycle, le produit commence par se localiser sur la cellule. Il interroge l'ensemble des lecteurs RFID pour se repérer. S'il ne se trouve pas c'est qu'il n'est pas en face d'un nœud décisionnel. La navette étant autopropulsée et avançant jusqu'au prochain nœud décisionnel, le produit n'a donc qu'à répéter cette étape jusqu'à réussir à trouver sa position dans la cellule.

3) Une fois que le produit actif est localisé, il peut choisir (en mode autonome) ou appliquer le choix (en mode exécutant), la prochaine ressource vers laquelle se rendre. L'allocation dépend du mode de fonctionnement choisi. Dans le cadre de l'architecture ORCA-FMS2 (voir chapitre précédent), le produit commence par lire la ressource choisie par l'ILP. Si elle est disponible, elle est choisie comme destination. Si elle ne l'est pas, le système n'est pas dans l'état prévu et le produit actif doit basculer en mode autonome. L'allocation est alors faite selon les champs de potentiel. Si une ressource est disponible, il la choisit, sinon il doit attendre qu'une ressource capable de réaliser sa prochaine opération se libère. Pour ne pas bloquer la production le produit continue son cheminement sur la boucle principale de la cellule.

4) Dès qu'une destination est choisie, le produit décide du chemin le plus rapide pour s'y rendre. Il utilise pour cela les champs de potentiel puisqu'ils sont propagés en intégrant le temps de transport effectif mis à jour en temps réel. Si le produit se trouve sur la destination, il passe directement à l'étape 5. Sinon il oriente l'aiguillage via l'intermédiaire d'un automate comme décrit dans la partie précédente et se dirige vers le prochain nœud. Il reboucle alors à l'étape 2.

5) Lorsque le produit a atteint sa destination, il peut obtenir le service de la part de la ressource. Il entre donc les informations nécessaires (type de produit, type de service demandé) dans l'automate et attend l'exécution du service. Si ce service est le dernier

de la gamme de fabrication, le produit passe à l'étape 6. Sinon le produit lit le prochain service de la liste et reboucle au niveau de l'étape 3.

6) Dans cette étape le produit à fabriquer est terminé. Il a été déchargé et la navette et l'Eeepc qui le supportait ne sont plus associés à un produit passif (la part virtuelle de l'entité produit actif est détruite). Les informations du produit terminé sont sauvegardées dans des fichiers de log et la navette passe en attente jusqu'à ce que la fabrication d'un nouveau produit soit demandée. Un nouveau plateau (i.e. plaque vierge où est assemblé un produit) sera alors chargé, le comportement du produit actif sera initialisé avec le nouveau produit à fabriquer et un nouveau cycle débutera.

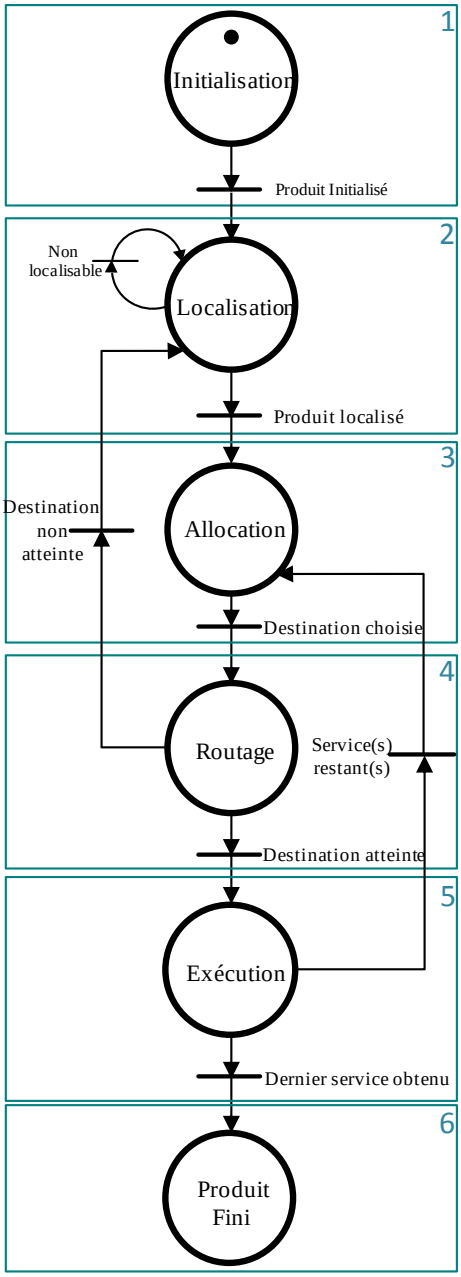


Figure 6.9– Comportement du produit actif

3.3. Mise en œuvre de la communication entre entités

Les entités de la cellule de production (produits et ressources) peuvent communiquer entre eux grâce au réseau présenté dans la figure 6.10. Ce réseau de communication est basé sur un réseau IP (*Internet Protocol*) à la fois filaire et sans-fil. Tous les automates sont câblés sur le réseau Ethernet filaire, ce qui permet des échanges d'informations directs et la réalisation de tâches synchronisées. Ils utilisent le protocole industriel Modbus / TCP (*Transmission Control Protocol*) qui est intégré nativement dans les automates Wago.

Les aiguillages sont câblés sur les entrées/sorties logiques de certains automates. Ainsi toute entité peut changer la position d'un aiguillage via le réseau Ethernet. L'automate, en plus de servir de passerelle pour commander l'aiguillage, stocke localement toutes les informations concernant cet aiguillage comme par exemple son état. Il contribue, en la facilitant, à la prise de décision (allocation et routage) des produits actifs.

Les ressources (robots d'assemblages, automate de vision…) sont elles aussi reliées aux entrées/sorties logiques des automates concernés. Ainsi les opérations fournies par les ressources sont lancés par l'automate gérant la ressource active concernée dès qu'un produit actif est en place ce qui assure la synchronisation entre produits et ressources actifs. Les informations concernant l'opération demandée par le produit actif sont envoyées à la ressource active via l'automate.

Les produits actifs se connectent à ce réseau grâce à un routeur WiFi industriel. Elles peuvent ainsi obtenir les informations propres aux aiguillages et aux ressources en contactant l'automate souhaité. Ils contrôlent l'ensemble des éléments de la cellule au travers de cette communication WiFi en utilisant un protocole Modbus / TCP. Finalement, pour se localiser dans la cellule, chaque produit actif est équipe d'un tag RFID comme illustré sur la figure 6.6. Des lecteurs répartis sur la cellule permettent de lire les tags des produits actifs circulant sur la cellule. L'Eeepc du produit actif peut ensuite interroger les lecteurs RFID pour se localiser précisément dans la cellule. Bien entendu il s'agit d'une version de laboratoire du produit actif. L'objectif était de réaliser un produit actif « prototype » avec des éléments commercialisés et connus de tous. L'Eeepc permet en plus d'afficher des données pour les démonstrations et de débuguer facilement les programmes.

Dans une version « entreprise », il serait remplacé par une carte de calcul intégrée directement dans la navette. Les lecteurs RFID seraient eux aussi embarqués et seuls des tags RFID se trouveraient sur la cellule. Ainsi le produit actif serait totalement inclus dans la navette. Un prototype de cette version « entreprise » est en cours de réalisation.

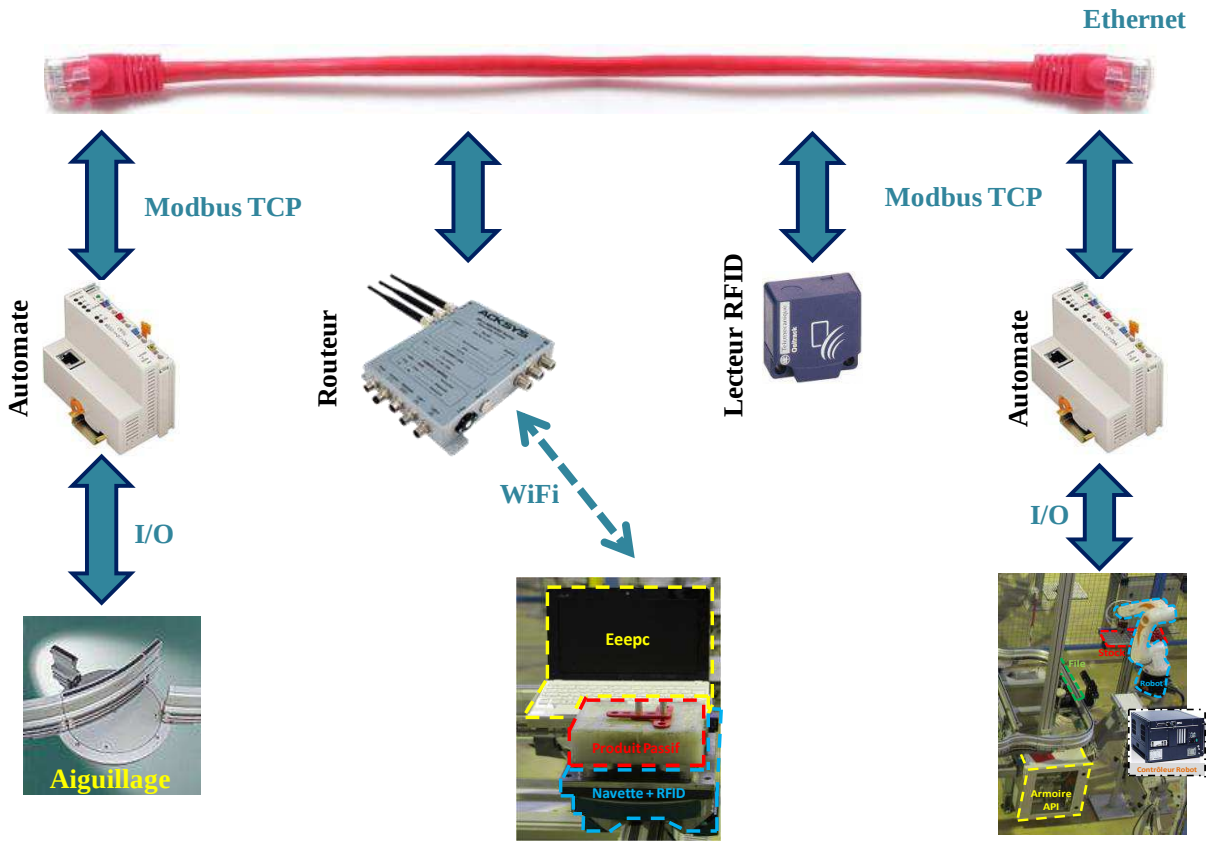


Figure 6.10– Réseau de communication

4. Mise en œuvre des concepts

Cette partie présente la mise en œuvre réelle des concepts intégrés dans ORCA-FMS : l'approche par champs de potentiel et l'ILP.

4.1. Mise en œuvre des Champs de Potentiel

L'approche générique par champs de potentiel a été présentée dans le chapitre III. Cette section va présenter la mise en œuvre de cette approche dans le cadre de la cellule AIP PRIME-CA de Valenciennes.

Émission et propagation des champs

Dans cette mise en œuvre des champs de potentiel, les ressources émettent les champs de potentiel pour attirer les produits. Des champs sont donc générés au niveau de chaque ressource par l'intermédiaire d'automates Wago. Un programme automate permet de créer un les champs de potentiel en fonction du champ de potentiel initial de la ressource, de la file d'attente actuelle et de la file d'attente maximum.

Le champ initial est fixé à 200 comme en simulation. Cette valeur a été fixée par expérimentation pour que même avec une file d'attente de 1 le champ reste visible sur l'ensemble de la boucle de convoyage principale. La file d'attente maximum pour toutes les ressources (de R_1 à R_7) est de 1 et la capacité des ressources est de 1 produit en traitement (comme en simula-

tion). Une fois que ce champ est créé par l'automate responsable d'une ressource particulière, il est propagé aux automates le précédant (compte tenu du sens de progression des navettes dans la cellule) sur le convoyeur comme illustré sur la figure 6.11. Cette propagation est réalisée grâce à l'utilisation des « variables réseaux » des automates. Ces variables permettent des échanges réguliers et systématiques entre automates.

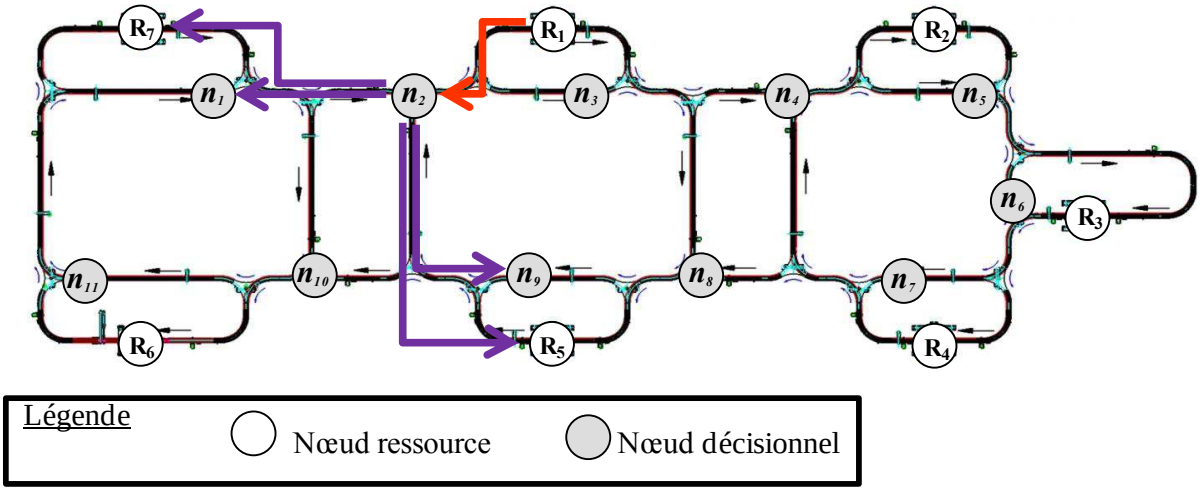


Figure 6.11– Émission et propagation des champs

Sur la figure 6.11, on voit que l'automate responsable de la ressource active R_1 créé et émet un champ de potentiel (flèche rouge). Ce champ de potentiel est envoyé au nœud précédent qui est ici n_2. L'automate en charge de n_2 va ensuite propager ce champ aux nœuds précédents (flèches mauves) qui sont n_1 et R_7 d'une part, mais aussi n_9 et R_5 d'autre part (et ainsi de suite).

Comme indique dans le chapitre IV (section 4), la propagation des champs de potentiel est affectée par la distance séparant l'émetteur du récepteur. Ainsi les automates qui reçoivent un champ de potentiel du nœud suivant doivent tenir compte de la distance entre eux et le nœud précédent. Cependant, dans une cellule flexible, la notion de distance entre deux nœuds est relative. En effet, en fonction de la vitesse des navettes et de la présence de ralentissements dus à l'occupation des tronçons de convoyage, l'opération de transport peut prendre plus ou moins de temps. C'est pourquoi, les champs sont réduits du temps effectif de transport entre deux nœuds qui est mis à jour dynamiquement. En effet dès qu'un produit actif sort d'un tronçon, il met à jour, dans l'automate correspondant à ce tronçon, le temps qu'il a mis pour le parcourir. Ainsi à chaque instant, les automates disposent du temps de transport effectif pour rejoindre le nœud suivant

Procédure d'échange entre le produit actif et un nœud décisionnel

Lorsqu'un produit actif arrive sur un nœud décisionnel (point de décision de routage situé juste avant un aiguillage), la procédure (présentée figure 6.12) est suivie.

1) Le produit actif transmet le temps de transport effectif depuis le nœud précédent au nœud actuel.
2) Le nœud actuel met à jour sa table des temps de transport pour régénérer les champs.
3) Le produit lit alors les champs de potentiel « captés » au niveau du nœud actuel.

4) Il choisit sa destination en prenant le champ le plus attractif et la transmet au nœud.
5) Le nœud décisionnel positionne l'aiguillage en correspondance avec le choix du produit actif.

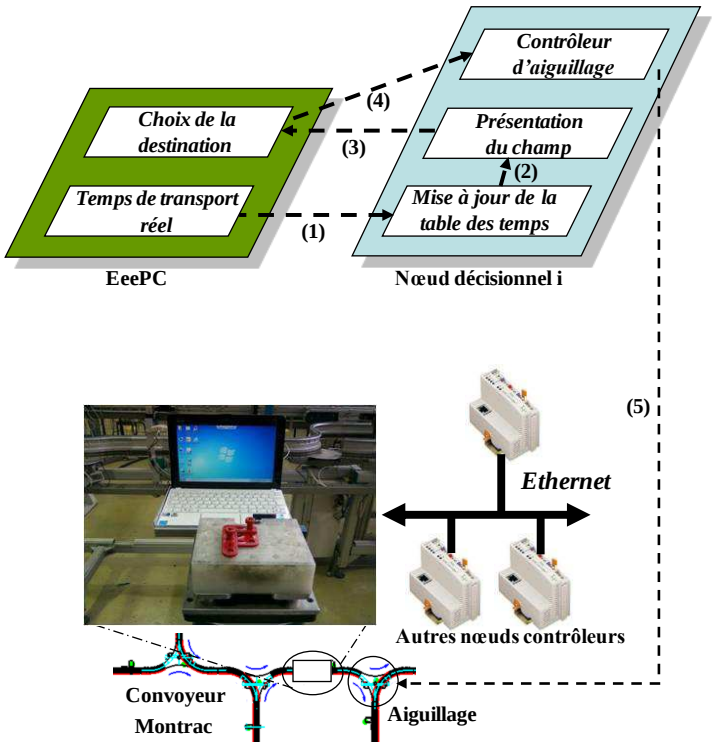


Figure 6.12– Procédure d'échange entre produit actif et nœud décisionnel

4.2. Mise en œuvre de l'ILP

Le modèle linéaire utilisé est le même que celui utilisé en simulation (cf. section 3.3 du chapitre V).

Étant donné que la résolution du modèle par Cplex est réalisée hors-ligne, les résultats peuvent être stockés dans un fichier texte et transmis au produit actif au moment du lancement de la production. Les ressources sur lesquelles le produit doit se rendre sont donc directement ajoutées dans le fichier de configuration de la gamme de fabrication comme sur la figure 6.13. Le chiffre après chaque opération représente l'identifiant de la ressource sur laquelle l'opération doit être faite.

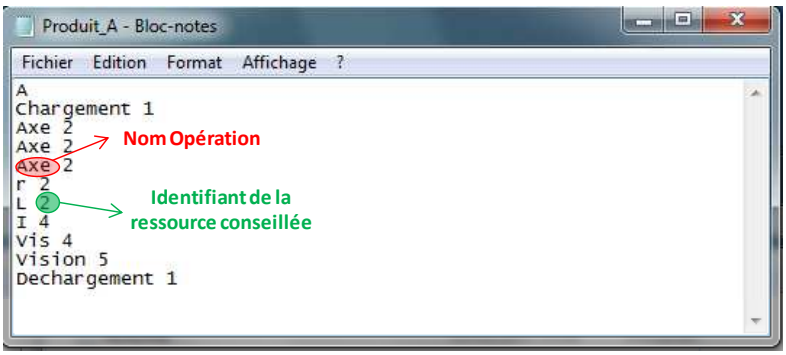


Figure 6.13– Gamme de fabrication d'un produit A contenant l'allocation basée sur l'ILP

L'**annexe 10** présente un retour d'expérience sur la mise en œuvre des entités et concepts d'ORCA-FMS dans un contexte industriel. La section suivante présente le protocole expérimental et les expérimentations réelles réalisées.

5. Étude du comportement d'ORCA-FMS en expérimentation

L'objectif de cette partie est de tester le comportement d'ORCA-FMS en expérimentation réelle. Ce comportement validé en simulation dans le chapitre V a été mis en œuvre sur la cellule flexible de l'AIP PRIMECA de Valenciennes.

5.1. Protocole expérimental

Pour valider le fonctionnement de cette approche en expérimentation les données expérimentales sur les ressources, produits et système de transport sont semblables à celles utilisées en simulation (présentée dans la partie 5 du chapitre V).

Les expérimentations réalisées dans ce chapitre sont décomposées en deux parties comme illustré sur la figure 6.14 :

- E1 : Ces expérimentations (#1 de la figure 6.14) compareront les résultats du comportement d'ORCA-FMS 1 & 2 en fonctionnement normal à une expérimentation avec les champs de potentiel seuls, puis aux résultats de l'ILP. Les résultats expérimentaux seront aussi comparés aux résultats obtenus en simulation S4.

- E2 : Ces expérimentations (#2 de la figure 6.14) testeront le comportement d'ORCA-FMS 1 & 2 en fonctionnement perturbé. Les résultats obtenus seront comparés aux résultats fournis par les champs de potentiel seuls, l'ILP et ceux obtenus lors de la simulation S5.

Les expérimentations du groupe E1 seront lancées avec les ordres de fabrication A-I-P, 2x A-I-P, puis 3x A-I-P alors qu'E2 sera lancé avec l'ordre 2x A-I-P. L'indicateur de performance utilisé pour ces expérimentations est le *Makespan*.

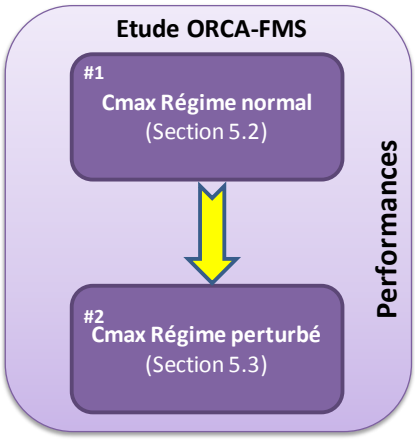


Figure 6.14– Protocole expérimental des expérimentations réelles

5.2. Expérimentation E1

Ce groupe d'expérimentations a pour objectif de montrer que le comportement de l'architecture ORCA-FMS en fonctionnement normal obtenu en simulation se confirme sur la cellule réelle. Les deux niveaux d'hybridation ORCA-FMS 1 & 2 sont donc comparés aux résultats expérimentaux obtenus avec une approche par champs de potentiel uniquement et à ceux fournis par l'ILP pour le modèle utilisé. Tous les résultats sont disponibles dans la table 6.I. La table 6.II rappelle les résultats obtenus lors de la simulation S4.

Table 6.I– Résultats de l'expérimentation E2

Campagne de Production	Optimal (sec)	Experimentation (secondes)			Gain / Champs de Pot. (secondes)		Gain / Champs de Pot. (%)	
		Champs de Pot.	ORCA-FMS 1	ORCA-FMS 2	ORCA-FMS 1	ORCA-FMS 2	ORCA-FMS 1	ORCA-FMS 2
AIP	219	290	272	272	18	18	6,2	6,2
2 x AIP	309	397	387	376	10	21	2,5	5,3
3 x AIP	409	514	488	478	26	36	5,1	7

Table 6.II– Rappel des résultats de la simulation S4

Campagne de Production	Optimal (sec)	Simulation (secondes)			Gain / Champs de Pot. (secondes)		Gain / Champs de Pot. (%)	
		Champs de Pot.	ORCA-FMS 1	ORCA-FMS 2	ORCA-FMS 1	ORCA-FMS 2	ORCA-FMS 1	ORCA-FMS 2
AIP	219	247	224	224	23	23	9,3	9,3
2 x AIP	309	345	323	314	22	31	6,4	9
3 x AIP	409	468	428	415	40	53	8,5	11,3

La première chose à souligner est qu'il existe un écart entre les résultats de simulation et les expérimentations réelles. Le simulateur n'inclut pas toutes les contraintes de la cellule réelle (vitesse variable des navettes, temps de communication / décision, etc.). Ainsi les résultats obtenus en expérimentation réelle sont légèrement différents de ceux obtenus en simulation. L'ILP étant utilisé hors ligne il donne les mêmes résultats. La table 6.III suivante présente l'écart entre la simulation et l'expérimentation réelle.

Table 6.III– Différences entre simulations et expérimentations

Campagne de Production	Différence Expérimentation /simulation (%)			Différence Réelle Expérimentation /simulation (%)	
	Champs de Pot.	ORCA-MS 1	ORCA-MS 2	ORCA-MS 1	ORCA-MS 2
AIP	14,83	17,65	17,65	2,82	2,82
2 x AIP	13,10	16,54	16,49	3,44	3,39
3 x AIP	8,95	12,30	13,18	3,35	4,23

La différence entre les résultats des champs de potentiel en simulation et expérimentation permet de quantifier la part de l'erreur due aux différences entre le modèle de simulation et la réalité. Les deux dernières colonnes de la table 6.III présentent la différence entre expérimentation et simulation pour ORCA-FMS, après retrait de cette part liée aux modèles. Cette différence « réelle » est en définitive très faible (maximum 4%).

La différence « réelle » est aussi due à une différence de modèles, ici entre le modèle linéaire et la réalité. Le modèle linéaire permettant à l'ILP de calculer l'optimal n'intègre pas non plus les contraintes non considérées par le simulateur. Ainsi les résultats obtenus par l'ILP correspondent réellement à l'optimal pour le simulateur mais certaines différences apparaissent avec le système réel. Dans ORCA-FMS 1 & 2, la couche basse de pilotage se base sur les résultats de l'ILP mais à cause des différences de modèles les choix conseillés par l'ILP peuvent au final ne pas être optimaux en réalité. Une perspective possible est d'améliorer la qualité des modèles de simulation et linéaire (ILP) afin de se rapprocher des expérimentations réelles même si certaines différences semblent difficilement modélisables (notamment les temps de transport variables ou les temps de traitement).

Cependant, les tendances dégagées en simulation se confirment en expérimentation réelle. ORCA-FMS 1 permet de gagner en moyenne 5% en termes de Cmax par rapport à l'approche par champs de potentiel et ORCA-FMS 2 un peu plus de 6%. ORCA-FMS permet donc bien d'améliorer les performances de la cellule réelle par rapport à une approche uniquement basée sur les champs de potentiel.

5.3. Expérimentation E2

Cette dernière expérimentation permet de vérifier le comportement d'ORCA-FMS en fonctionnement perturbé en expérimentation. Comme pour la simulation S5 la ressource R_2 sera « réservée pour maintenance » suite à un défaut détecté à t=100 secondes. La table 6.IV présente les résultats des expérimentations E2. La table 6.V rappelle les résultats obtenus lors de la simulation S5.

Table 6.IV– Résultats de l'expérimentation E2

Campagne de Production	ILP*	Expérimentation (secondes)			Ecart / ILP* (%)			Gain /champs de pot. (%)	
		Champs de Pot.	ORCA-FMS 1	ORCA-FMS 2	Champs de Pot.	ORCA-FMS1	ORCA-FMS2	ORCA-FMS1	ORCA-FMS2
2 x AIP - M2 en maintenance (t=100)	376	467	460	460	24.2	22.3	22.3	1.5	1.5

Table 6.V– Rappel des résultats de la simulation S5

Campagne de Production	ILP*	Simulation (secondes)			Ecart / ILP* (%)			Gain /champs de pot. (%)	
		Champs de Pot.	ORCA-FMS 1	ORCA-FMS 2	Champs de Pot.	ORCA-FMS1	ORCA-FMS2	ORCA-FMS1	ORCA-FMS2
2 x AIP - M2 en maintenance (t=100)	376	405	388	388	7.7	3.2	3.2	4.2	4.2

Comme pour la simulation, il n'y a pas de différence entre ORCA-FMS 1 & 2 puisque la panne survient très tôt dans la production. L'amélioration de performance par rapport aux champs de potentiel est faible à cause des différences de modèles qui sont discutées dans la partie 4.2. Cependant, ORCA-FMS remplit son rôle en permettant de réagir à une perturbation.

6. Conclusion

Ce chapitre a présenté une expérimentation d'ORC-FMS sur une cellule de production pédagogique mais réelle. Après avoir présenté les éléments technologiques qui concernent la cellule, ce chapitre a présenté la façon dont ont été mis en œuvre les concepts de produits actifs, ressources actives et les approches utilisées dans ORCA-FMS (i.e. ILP et champs de potentiel).

Les expérimentations de ce chapitre ont été divisées en deux catégories. Une première (E1) permettant de valider le comportement d'ORCA-FMS en régime normal et une seconde (E2) en régime perturbé. Les tendances obtenues en simulation sont toutes confirmées en expérimentation. ORCA-FMS améliore les performances en fonctionnement normal tout en permettant de réagir en cas de perturbation. La cellule réelle étant très difficilement modélisable, la solution optimale calculée par l'ILP diffère légèrement de la solution réelle mais permet tout de même un gain notable de performance malgré les différences de modèles.

Des expérimentations annexes ont été effectuées afin de confirmer les simulations illustrant l'intégration de la consommation énergétique dans ORCA-FMS. L'**annexe 11** détaille ces expérimentations supplémentaires. Les résultats obtenus en expérimentation confirment ceux des simulations correspondantes tout en validant la faisabilité de cette approche intégrant l'énergie.

Le chapitre suivant présente les conclusions et perspectives de cette thèse.

Conclusion générale et perspectives

Conclusion générale

L'objectif de cette thèse est de contribuer au contrôle de la myopie dans les architectures de pilotage des systèmes. Afin d'y parvenir une architecture de pilotage hybride nommé ORCA est proposée. Cette architecture ORCA est appliquée dans cette thèse pour le pilotage des systèmes flexibles de production sous l'appellation ORCA-FMS. Cette dernière architecture basée sur l'utilisation conjointe de la programmation linéaire et d'une approche par champs de potentiel est conçue pour permettre un pilotage (i.e. gestion de l'allocation des produits aux ressources et de leur routage) optimisé et réactif.

Nous avons présenté le contexte du travail de thèse dans le premier chapitre. L'évolution du contexte industriel soumet les systèmes de production à de nouvelles contraintes comme la *mass customization* et l'intégration de l'énergie dans les décisions de pilotage. Afin de s'adapter à ces nouvelles contraintes le système doit être réactif (avec un temps de réponse court exprimé en secondes/minutes). Cependant, cette réactivité se fait généralement au détriment des performances globales du système à cause de phénomènes tels que la myopie. Ce contexte nous a amené à étudier les architectures de pilotage hybrides (mixant hétérarchie et hiérarchie) qui sont potentiellement réactives tout en permettant le contrôle de la myopie quand nécessaire.

Dans le deuxième chapitre, un état de l'art des architectures de pilotage hybride a été proposé. Cet état de l'art laisse apparaître quatre sous-classes d'architectures hybrides et souligne l'intérêt des architectures dites « dynamiques ». Un second état de l'art a ensuite permis de détailler les architectures de pilotage hybride dynamiques existantes et notamment d'étudier le mécanisme de basculement leur permettant de passer d'un pilotage « hiérarchique » à un pilotage « hétérarchique ». Ces mécanismes ont été positionnés selon une typologie comportant trois axes : l'homogénéité du basculement, la localisation de la décision de basculement et les circonstances du déclenchement du basculement.

Dans le troisième chapitre, a été proposée l'architecture ORCA qui répond aux spécifications reprises dans les deux premiers chapitres. ORCA est générique et permet un pilotage optimisé et réactif de systèmes. La démarche de conception d'ORCA ainsi que sa structure en trois couches ont été présentées dans ce chapitre. Un modèle holonique générique permettant de faire le lien entre les éléments de l'architecture ORCA et les éléments du système physique a aussi été proposé. Ce modèle a pour objectif d'accompagner la mise en œuvre effective de l'architecture ORCA.

Dans le quatrième chapitre, l'architecture ORCA a été appliquée au pilotage des systèmes flexibles de production. Cette application d'ORCA, nommée ORCA-FMS, utilise un modèle linéaire (i.e. ILP) afin d'assurer l'optimisation de la performance globale et une approche par champs de potentiel afin d'être réactif aux évènements inattendus. Après présentation de chacune des deux approches, le mécanisme de basculement permettant à l'une ou l'autre de prendre en charge le pilotage du système a été détaillé et justifié. Finalement, la modélisation de l'architecture ORCA-FMS a été présentée.

Le cinquième chapitre a permis de valider l'architecture ORCA-FMS en simulation. À l'aide de la plateforme multi-agents NetLogo, un modèle de simulation a été développé. Ce modèle a permis de vérifier l'optimisation et la réactivité d'ORCA-FMS sur le cas d'étude de la cellule flexible AIP PRIMECA de Valenciennes. En accord avec un benchmark (récemment élaboré (Trentesaux et al., 2013)), des scénarios statiques et dynamiques (i.e. intégrant un évènement inattendu) ont permis d'éprouver ORCA-FMS. Les résultats obtenus ont permis de valider la pertinence d'ORCA-FMS pour ce cas d'étude précis.

Le sixième chapitre a détaillé la mise en œuvre d'ORCA-FMS sur la cellule flexible AIP PRIMECA de Valenciennes. Ce chapitre a tout d'abord présenté les éléments utilisés lors de la mise en œuvre (Automates Programmables Industriels, produits actifs, ressources actives, etc.). La mise en œuvre des deux approches utilisées (ILP et champs de potentiel) a ensuite été détaillée afin de présenter et justifier l'ajustement des paramétrages utilisés. Finalement, des expérimentations réelles ont permis de valider les résultats obtenus en simulation et de confirmer la pertinence d'ORCA-FMS sur le cas d'étude de la cellule AIP PRIMECA de Valenciennes. Cette mise en œuvre a aussi permis de valider la faisabilité de l'utilisation de l'architecture ORCA-FMS sur un système très proche d'un système industriel.

Perspectives

Ce travail de thèse ouvre de nombreuses perspectives que ce soit à court, moyen ou long terme.

Perspectives à court terme : Les perspectives à court termes concernent l'amélioration des modèles ou approches utilisés ainsi que la diversification des scénarios de validation testés. Les améliorations envisagées sont les suivantes :

- Utiliser une heuristique ou méta-heuristique à la place de l'ILP pour mettre en œuvre la couche de contrôle global d'ORCA-FMS. Cela permettrait d'accroître la réactivité du niveau global et par exemple permettre de recalculer l'ordonnancement en pleine production et faciliter la mise en œuvre du « switch back ».

- Tester l'architecture sur des scénarios plus complexes. Même si pour les champs de potentiel des productions de 50 produits ont été testées (cf. table A.V), les scenarios de validation d'ORCA-FMS se limitent à 10 produits. Ceci est essentiellement dû à deux choses : d'une part l'incapacité de l'ILP à fournir une solution rapide pour des

productions plus complexes et d'autre part la limite de 10 ressources de transport disponibles sur le système réel.

- Tester l'architecture dans un environnement très perturbé avec un nombre de pannes plus important et des pannes plus variées afin d'étudier la robustesse de l'approche.

- Valider une application d'ORCA avec une couche de contrôle local différente comme par exemple les *Delegate MAS*, la stigmergie ou une base de connaissance dynamique proposant des solutions aux évènements inattendus basées sur les expériences passées. Cette validation permettrait de confirmer la généricité d'ORCA et de comparer les résultats obtenus avec les champs de potentiel.

- Étudier la compatibilité entre les entités en mode « exécutant » et « autonome » d'ORCA-FMS. Dans les scénarios de validation de ce travail de thèse, il arrivait que certaines entités en mode autonome « vole » une ressource choisie par une entité en mode exécutant. Ceci faisait basculer cette dernière en mode autonome et donc avait tendance à dégrader la performance globale du système. L'instauration de règles de priorités par exemple est une piste à approfondir pour corriger ce problème.

- Valider le fonctionnement de l'architecture ORCA-FMS avec une fonction objectif différente, prenant en compte notamment des *due dates.* Ceci passe vraisemblablement par le changement de l'approche de contrôle local, puisque les champs de potentiel ne semblent pas adaptés à ce type d'objectif et du modèle du contrôle global qui ne contient aucune variable liées aux avances et retards.

- Mettre en œuvre ORCA-FMS à l'aide d'un langage créé pour les systèmes distribués afin de favoriser la scalabilité et la tolérance aux pannes du système. Une première piste très prometteuse est le langage Erlang (Erlang Web, 2013).

- Obtenir des résultats d'approches concurrentes sur le benchmark développé afin de positionner les travaux de cette thèse et de déterminer les points à améliorer.

Perspectives à moyen terme : À moyen terme, cette thèse permet de mettre en valeur des axes de recherche prometteurs :

- Étudier le contrôle de la myopie au niveau stratégique du système de production (planification, logistique, finances)

- Compléter l'étude sur la myopie en proposant des moyens permettant de l'observer et la quantifier.

- Étudier l'impact d'un mécanisme de décision de basculement non atomique, c'est-à-dire pouvant être réparti entre plusieurs entités, éventuellement localisés sur des couches de contrôle différentes.

- Intégrer d'autres fonctions dans le niveau pilotage 1 géré par l'architecture ORCA comme la maintenance, l'approvisionnement ou la gestion de la qualité.

- Appliquer les travaux de cette thèse à un système utilisé en industrie. En effet, même si le système utilisé est très proche d'un système industriel, il n'est pas directement connecté à la réalité et au marché. Une validation de ces travaux sur un système de production réellement connecté aux marchés serait des plus instructives, à l'instar de P2000+ (Bussmann et Schild, 2000).

- Appliquer ORCA à d'autres domaines que le pilotage des FMS. Des premiers pas ont été faits en parallèle de cette thèse pour l'application d'ORCA aux systèmes hospitaliers ou à la gestion d'un cross-dock.

- Étudier l'intégration de capacité de communication de proche en proche entre les entités, et des principes de coopération et d'évitement de situation non-coopératives à détecter.

- Réaliser un guide pour faciliter la mise en œuvre d'ORCA et le choix des différents éléments la composant (optimiseurs globaux et locaux, mécanisme de basculement, etc.) en fonction du domaine d'application, du système physique et de ses objectifs.

Perspectives à long terme : Les perspectives à long terme concerne le mécanisme de basculement « retour » (i.e. revenir du mode autonome au mode exécutant). Dans ce travail de thèse, un premier pas a été fait pour mettre en œuvre réellement ce mécanisme retour, ce qui a soulevé les perspectives suivantes :

- Concevoir un modèle réaliste de prédiction de l'état de la couche système afin de fournir avec précision l'état du système à un instant t+1 à la couche de contrôle global. Le problème se situe notamment au niveau des entités en mode autonome qui réagissent en temps réel aux évènements inattendus et dont le comportement n'est donc pas prévisible au premier abord. Ces entités continuant d'évoluer pendant le recalcul de l'optimiseur global, il faut estimer leur comportement.

- Trouver un moyen pour déterminer le point de synchronisation, à partir duquel toutes les entités du système reviennent en mode exécutant. Ce point de synchronisation n'est pas simple à définir car il dépend :
    - De la précision du modèle de prédiction,

- De la vitesse de re-calcul de l'optimiseur global,
- De l'estimation faite du temps de recalcul de l'optimiseur global,
- De la capacité de l'optimiseur global à recalculer un ordonnancement à partir d'un état du système différent de l'état initial.

Liste des acronymes

ADACOR : ADAptive holonic COntrol aRchitecture
API : Automate Programmable Industriel
BMS : Bionic Manufacturing Systems
CH : Control Hiérarchique
CIM : Computer Integrated Manufacturing
CG : couche de Contrôle Global
CL : couche de Contrôle Local
Cmax : Makespan
CP : Champs de Potentiel
CPG : Champs de Potentiel Généré
CPI : Champs de Potentiel Initial
CPR : Champs de Potentiel Reçu
CS : Couche Système
CSP : Constraint Satisfaction Problems
CSS : Centralized Scheduling System
DE : Déclenchement sur Évènement
DHe : Dynamique et Hétérogène
DHo : Dynamique et Homogène
DM : Déclenchement Mixte
D-MAS : Delegate Multi-Agent System
DMES : Distributed Manufacturing Execution System
DP : Déclenchement Périodique
EP : Entité Produit
EP : Entité Resource
ERP : Enterprise Resource Planning
FA : File d'Attente actuelle
FIFO : First In First Out
Fmax : File d'attente maximale
FJSP : Flexible Job-Shop Problem
FMS : Flexible Manufacturing Systems (équivalent à SFP)
FrMS : Fractal Manufacturing Systems
HCBA : Holonic Component-Based Architecture
HMS : Holonic Manufacturing Systems
HE : déclenchement HEterogène
HN : contrôle Hétérarchique Négocié

HNN : control Hétérarchique Non Négocié
HO : déclenchement HOmogène
ILP : Integer Linear Programming
IP : Internet Protocol
IMS : Intelligent Manufacturing Systems
MAS : Multi-Agent Systems
MES : Manufacturing Execution System
MILP : Mixed-Integer Linear Programming
NB : basculement localisé au Niveau Bas
NH : basculement localisé au Niveau Haut
OG : Optimiseur Global
OL : Optimiseur Local
ORCA : Architecture for an Optimized and Reactive Control
ORCA-FMS : ORCA pour le pilotage d'un Flexible Manufacturing Systems
PAC : Programmable Automation Controller
PLC : Programmable Logic Controller
PROSA : Product Resource Order Staff Architecture
PROSIS : Product, Resource, Order and Simulation Isoarchic Structure
RFID : Radio Frequency IDentification
RMS : Reconfigurable Manufacturing Systems
SCADA : System Control And Data Acquisition
SCP : Système Contrôlé par le Produit
SEI : Service Espace Informationnel
SEP : Service Espace Physique
SFC : Sequential Function Chart
SFI : Service Forme Informationnel
SFP : Service Forme Physique
SOA : Service Oriented Architecture
SSD : Solid-State Disk
STI : Service Temps Informationnel
STP : Service Temps Physique
SFP : Systèmes Flexibles de Production (équivalent à FMS)
SHe : Statique et Hétérogène
SHo : Statique et Homogène
TCP : Transmission Control Protocol
TTE : Temps de Transport Effectif
WiFi : Wireless Fidelity
WIP : Work In Progress
YAMS : Yet Another Manufacturing System

Références bibliographiques

ABC NetMarketing, 1997. http://www.definitions-marketing.com/Definition-Mass-customization (accessed 10.14.13).

Adam, E., Mandiau, R., 2007. Flexible Roles in a Holonic Multi-Agent System, in: Mařík, V., Vyatkin, V., Colombo, A.W. (Eds.), Holonic and Multi-Agent Systems for Manufacturing, Lecture Notes in Computer Science. Springer Berlin Heidelberg, 59–70.

Adam, E., Zambrano, G., Pach, C., Berger, T., Trentesaux, D., 2011. Myopic Behaviour in Holonic Multiagent Systems for Distributed Control of FMS. Trends Pr. Appl. Agents Multiagent Syst., 91–98.

Agha, G., 1997. Abstracting interaction patterns: A programming paradigm for open distributed systems. Formal Methods for Open Object-Based Distributed Systems, IFIP Transactions 1, 135–153.

Agha, G., Mason, I.A., Smith, S.F., Talcott, C.L., 1997. A foundation for actor computation. J. Funct. Program. 7, 1–72.

Aissani, N., 2010. Pilotage adaptatif et réactif pour un système de production à flux continu: application à un système de production pétrochimique (PhD Thesis), Université de Valenciennes et du Hainaut-Cambresis.

Andersson, M.R., Sandholm, T.W., 2001. Leveled commitment contracts with myopic and strategic agents. J. Econ. Dyn. Control 25, 615–640.

Andreev, V., Batishchev, S., Skobelev, P., Vittikh, V., 2003. Methods and toolset for designing multi-agent systems for decision making support. J. Russ. Acad. Sci., Computer and Systems Sciences International, 126–137.

Anwar, M.F., Nagi, R., 1998. Integrated scheduling of material handling and manufacturing activities for just-in-time production of complex assemblies. Int. J. Prod. Res. 36, 653–681.

ASUSTeK, 2012. http://eeepc.asus.com/ (accessed 10.14.13).

Atzori, L., Iera, A., Morabito, G., 2010. The internet of things: A survey. Comput. Networks 54, 2787–2805.

Babiceanu, R.F., Chen, F.F., 2006. Development and Applications of Holonic Manufacturing Systems: A Survey. J. Intell. Manuf. 17, 111–131.

Baker, A.D., 1998. A survey of factory control algorithms that can be implemented in a multi-agent heterarchy: dispatching, scheduling, and pull. J. Manuf. Syst. 17, 297–320.

Barbosa, J., Leitão, P., Adam, E., Trentesaux, D., 2012b. Nervousness in Dynamic Self-organized Holonic Multi-agent Systems, in: Pérez, J.B., Sánchez, M.A., Mathieu, P., Rodríguez, J.M.C., Adam, E., Ortega, A., Moreno, M.N., Navarro, E., Hirsch, B., Lopes-Cardoso, H., Julián, V. (Eds.), Highlights on Practical Applications of Agents and Multi-Agent Systems, Advances in Intelligent and Soft Computing. Springer Berlin Heidelberg, 9–17.

Barbosa, J., Leitao, P., Trentesaux, D., Adam, E., 2011. Enhancing ADACOR with biology insights towards reconfigurable manufacturing systems, in: IECON 2011 - 37th Annual Conference on IEEE Industrial Electronics Society. Presented at the IECON 2011 - 37th Annual Conference on IEEE Industrial Electronics Society, 2746–2751.

Bauer, B., Müller, J.P., Odell, J., 2001. Agent UML: A formalism for specifying multiagent software systems. Int. J. Softw. Eng. Knowl. Eng. 11, 207–230.

Bilge, Ü., Ulusoy, G., 1995. A Time Window Approach to Simultaneous Scheduling of Machines and Material Handling System in an FMS. Oper. Res. 43, 1058–1070.

Böhnlein, D., Schweiger, K., Tuma, A., 2011. Multi-agent-based transport planning in the newspaper industry. Int. J. Prod. Econ. 131, 146–157.

Bojic, I., Lipic, T., Kusek, M., Jezic, G., 2011. Extending the JADE agent behaviour model with JBehaviourTrees Framework, in: Agent and Multi-Agent Systems: Technologies and Applications. Springer, 159–168.

Bolton, R., Tyler, S., 2008. PQLI Engineering Controls and Automation Strategy. J. Pharm. Innov. 3, 88–94.

Bongaerts, L., Valckenaers, P., Van Brussel, H., Wyns, J., 1995. Schedule execution for a holonic shop floor control system. Adv. Summer Inst. ASI'95 Life Cycle Approaches Prod. Syst. Lisbon Portugal.

Borangiu, T., Raileanu, S., Trentesaux, D., Berger, T., Iacob, I., 2012. Distributed manufacturing control with extended CNP interaction of intelligent products. J. Intell. Manuf., 1–11, doi:10.1007/s10845-013-0740-3.

Boufaied, A., 2003. Contribution à la surveillance distribuée des systèmes à événements discrets complexes (PhD Thesis), Université Paul Sabatier-Toulouse III.

Boyer, S.A., 2009. SCADA: supervisory control and data acquisition. International Society of Automation.

Brandl, D., 2012. Practical Applications of the ISA 95 standard.

Bussmann, S., McFarlane, D.C., 1999. Rationales for holonic manufacturing control, in: Proc. of Second Int. Workshop on Intelligent Manufacturing Systems. 177–184.

Bussmann, S., Schild, K., 2000. Self-organizing manufacturing control: An industrial application of agent technology, in Proceedings of Fourth International Conference On MultiAgent Systems, 87–94.

Buzacott, J.A., Yao, D.D., 1986. Flexible manufacturing systems: a review of analytical models. Manag. Sci., 890–905.

Camalot, J.P., 2000. Aide à la décision et à la coopération en gestion du temps et des ressources (PhD Thesis), Institut National des Sciences Appliquées de Toulouse.

Cardin, O., Castagna, P., 2009. Using online simulation in Holonic manufacturing systems. Eng. Appl. Artif. Intell. 22, 1025–1033.

Cardin, O., Mebarki, N., Pinot, G., 2013. A study of the robustness of the group scheduling method using an emulation of a complex FMS. Int. J. Prod. Econ. 146, 199–207.

Caumond, A., Lacomme, P., Moukrim, A., Tchernev, N., 2009. An MILP for scheduling problems in an FMS with one vehicle. Eur. J. Oper. Res. 199, 706–722.

Cavalieri, S., Garetti, M., Macchi, M., Taisch, M., 2000. An experimental benchmarking of two multi-agent architectures for production scheduling and control. Comput. Ind. 43, 139–152.

Chirn, J.L., McFarlane, D.C., 2000. A holonic component-based approach to reconfigurable manufacturing control architecture, in Proceedings of 11th International Workshop On Database and Expert Systems Applications, 219–223.

Christensen, J.H., 1994. Holonic manufacturing systems: initial architecture and standards directions, in Proceedings of 1st Euro Wkshp Holonic Manuf. Syst.

Chu, Y., You, F., Wassick, J.M., 2014. Hybrid method integrating agent-based modeling and heuristic tree search for scheduling of complex batch processes. Comput. Chem. Eng. 60, 277–296.

Clair, G., Kaddoum, E., Gleizes, M.-P., Picard, G., 2008. Self-regulation in self-organising multi-agent systems for adaptive and intelligent manufacturing control, in Proceedings of Second IEEE International Conference On Self-Adaptive and Self-Organizing Systems, 107–116.

Cognex Corporation, 2013. http://www.cognex.com/ (accessed 10.14.13).
Conway, R.W., Maxwell, W.L., Miller, L.W., 2003. Theory of scheduling. Dover Publications.
Cox, J.F., Blackstone, J.H., Spencer, M.S., 1995. APICS dictionary.
Cox, J.S., Durfee, E.H., 2003. Discovering and exploiting synergy between hierarchical planning agents, in: Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems. 281–288.
Da Silveira, G., Borenstein, D., Fogliatto, F.S., 2001. Mass customization: Literature review and research directions. Int. J. Prod. Econ. 72, 1–13.
Darling, N., Hamilton, S., Toyokawa, T., Matsuda, S., 2002. Naturally occurring mentoring in Japan and the United States: Social roles and correlates. Am. J. Community Psychol. 30, 245–270.
David, R., Alla, H., 1992. Petri nets and Grafcet: tools for modelling discrete event systems, Englewood Cliffs, NJ: Prentice-Hall.
De Guillebon, B., Bihouix, P., 2012. Quel futur pour les métaux?: Raréfaction des métaux: un nouveau défi pour la société. EDP Sciences.
Dilts, D.M., Boyd, N.P., Whorms, H.H., 1991. The evolution of control architectures for automated manufacturing systems. J. Manuf. Syst. 10, 79–93.
Dindeleux, E., 1992. Proposition d'un modèle et d'un système interactif d'aide à la décision pour la conduite d'atelier (PhD Thesis), Université de Valenciennes et du Hainaut-Cambresis.
Dorigo, M., Bonabeau, E., Theraulaz, G., 2000. Ant algorithms and stigmergy. Future Gener. Comput. Syst. 16, 851–871.
Duffie, N.A., 1996. Heterarchical control of highly distributed manufacturing systems. Int. J. Comput. Integr. Manuf. 9, 270–281.
Duffie, N.A., Prabhu, V.V., 1994. Real-time distributed scheduling of heterarchical manufacturing systems. J. Manuf. Syst. 13, 94–107.
Erceau, J., Ferber, J., 1991. L'intelligence artificielle distribuée. La recherche 750–758.
Erl, T., 2004. Service-oriented architecture. Prentice Hall Englewood Cliffs.
Erlang Web, 2013. http://www.erlang.org/ (accessed 10.14.13).
Esquirol, P., Lopez, P., Fargier, H., Schiex, T., 1995. Constraint programming. JORBEL Belg. J. Oper. Res. Stat. Comput. Sci. 35, 5–36.
Fattahi, P., Mehrabad, M.S., Jolai, F., 2007. Mathematical modeling and heuristic approaches to flexible job shop scheduling problems. J. Intell. Manuf. 18, 331–342.
Ferber, J., 1995. Les systèmes multi-agents: vers une intelligence collective. Informatique. Intell. Artif. InterÉditions.
Ferber, J., Gutknecht, O., 1998. A meta-model for the analysis and design of organizations in multi-agent systems, in Proceedings of International Conference On Multi Agent Systems, 128–135.
Ferber, J., Gutknecht, O., Michel, F., 2004. From agents to organizations: an organizational view of multi-agent systems. Agent-Oriented Softw. Eng. IV, 443–459.
Fischer, K., Schillo, M., Siekmann, J., 2003. Holonic multiagent systems: A foundation for the organisation of multiagent systems. Holonic Multi-Agent Syst. Manuf. 1083–1084.
Garey, M.R., Johnson, D.S., 1979. Computers and Intractability: A Guide to the Theory of NP-completeness. WH Freeman and Company, New York.
Gascueña, J.M., Garijo, F.J., Fernández-Caballero, A., Gleizes, M.-P., Glize, P., 2012. Implementation and Assessment of Robot Team Cooperation Models Using Deliberative Control Components, in Proceedings of Advances in Artificial Intelligence–IBERAMIA 2012. Springer, 412–421.

Gaud, N., Galland, S., Hilaire, V., Koukam, A., 2008. An organisational platform for holonic and multiagent systems. PROMAS-6 AAMAS 8, 111–126.

Gill, A., 1970. Finite-State Machines. IEEE Trans. Comput. 19.

Giret, A., Botti, V., 2004. Holons and agents. J. Intell. Manuf. 15, 645–659.

Gomes, M.C., Barbosa-Povoa, A.P., Novais, A.Q., 2005. Optimal scheduling for flexible job shop operation. Int. J. Prod. Res. 43, 2323–2353.

Grabot, B., 2011. Prospective Productique 2011.

Grabot, B., 2012. Prospective Productique 2012.

Harel, D., 1987. Statecharts: A visual formalism for complex systems. Sci. Comput. Program. 8, 231–274.

Hayes-Roth, B., 1985. A blackboard architecture for control. Artif. Intell. 26, 251–321.

Heragu, S.S., Graves, R.J., Byung-In Kim, St Onge, A., 2002. Intelligent agent based framework for manufacturing systems control. IEEE Trans. Syst. Man Cybern. Part A, 32, 560–573.

Herrera, C., 2011. Cadre générique de planification logistique dans un contexte de décisions centralisées et distribuées (PhD Thesis), Université Henri Poincaré-Nancy I.

Hewitt, C., 1977. Viewing control structures as patterns of passing messages. Artif. Intell. 8, 323–364.

Hoc, J.-M., Amalberti, R., 1994. Diagnostic et prise de décision dans les situations dynamiques. Psychol. Française 39, 177–192.

Horn, P., 2001. Autonomic computing: IBM's Perspective on the State of Information Technology.

Huang, Y., 2012. The Green Manufacturing - The Development Trend of Machinery Manufacturing Industry in the 21st Century. Adv. Mater. Res. 503-504, 202–205.

IBM ILOG CPLEX Optimization Studio, 2013 http://www-01.ibm.com/software/integration/optimization/cplex-optimization-studio/ (accessed 10.14.13).

IMS2020.net , 2011. http://data.fir.de/projektseiten/ims2020/files/IMS2020_Brochure_KAT1-5.pdf (accessed 10.14.13).

Jarvis, J., Jarvis, D., Martin, A., 2011. A goal-based approach to holonic manufacturing, in: Lectures Notes in Artificail Intelligence. Presented at the Holomas 2011, Springer, 205–214.

Jarvis, J., Ronnquist, R., McFarlane, D., Jain, L., 2006. A team-based holonic approach to robotic assembly cell control. J. Netw. Comput. Appl. 29, 160–176.

Johnston, K., 2009. Extending the marketing myopia concept to promote strategic agility. J. Strat. Mark. 17, 139–148.

Kadera, P., Tichy, P., 2009. Plan, Commit, Execute Protocol in Multi-agent Systems, in: Mařík, V., Strasser, T., Zoitl, A. (Eds.), Holonic and Multi-Agent Systems for Manufacturing. Springer Berlin Heidelberg, Berlin, Heidelberg, 155–164.

Kadri, F., Pach, C., Chaabane, S., Berger, T., Trentesaux, D., Tahon, C., Sallez, Y., 2013. Modelling and management of the strain situations in hospital systems using ORCA approach. Presented at the International Conference on Industrial Engineering and Systems Management, Rabat.

Kärkkäinen, M., Holmström, J., Främling, K., Artto, K., 2003. Intelligent products–a step towards a more effective project delivery chain. Comput. Ind. 50, 141–151.

Kenneth, N.M., Frank, R.S., Buzacott, J.A., 1995. A review of hierarchical production planning and its applicability for modern manufacturing. Prod. Plan. Control 6, 384–394.

Kephart, J.O., Chess, D.M., 2003. The vision of autonomic computing. Computer 36, 41–50.

Khatib, O., 1986. Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. Int. J. Robot. Res. 5, 90–98.

Koestler, A., 1967. The ghost in the machine. 1967. Lond. Hutchinson.

Krafzig, D., Banke, K., Slama, D., 2005. Enterprise SOA: service-oriented architecture best practices. Prentice Hall PTR.

Kristensen, B.B., Østerbye, K., 1996. Roles: conceptual abstraction theory and practical language issues. Theory Pr. Object Syst. 2, 143–160.

Kuka Roboter GMbH, 2013. http://www.kuka-robotics.com/fr/ (accessed 10.14.13).

Langer, T., Weber, M., 2005. Myopic prospect theory vs. myopic loss aversion: how general is the phenomenon? J. Econ. Behav. Organ. 56, 25–38.

Le Moigne, J.-L., 1994. La théorie du système général: théorie de la modélisation. Presses Universitaires de France-PUF.

Lee, J.H., Kim, C.O., 2008. Multi-agent systems applications in manufacturing systems and supply chain management: a review paper. Int. J. Prod. Res. 46, 233–265.

Leitão, P., 2004. An agile and adaptive holonic architecture for manufacturing control (PhD Thesis), Faculty of Engineering of University of Porto.

Leitão, P., 2009. Agent-based distributed manufacturing control: A state-of-the-art survey. Eng. Appl. Artif. Intell. 22, 979–991.

Leitão, P., Alves, J., Pereira, A.I., 2010. Solving myopia in real-time decision-making using Petri nets models' knowledge for service-oriented manufacturing systems, in Proceedings of Preprints of the IFAC Workshop on Intelligent Manufacturing Systems (IMS'10). Lisboa, Portugal. 155–160.

Leitao, P., Colombo, A.W., Restivo, F.J., 2005. ADACOR: a collaborative production automation and control architecture. Intell. Syst. IEEE 20, 58–66.

Leitão, P., Restivo, F., 2006. ADACOR: A holonic architecture for agile and adaptive manufacturing control. Comput. Ind. 57, 121–130.

Leon, A., 2007. Enterprise resource planning. Tata McGraw-Hill Education.

Lesourne, J., 2011. Énergie: nouvelle donne économique et politique. À propos du livre de Jean-Pierre Hansen et Jacques Percebois, Énergie. Économie et politiques. Futuribles 103–107.

Liu, J., MacCarthy, B.L., 1997. A global MILP model for FMS scheduling. Eur. J. Oper. Res. 100, 441–453.

Mackworth, A.K., 1977. Consistency in networks of relations. Artif. Intell. 8, 99–118.

Mancebo, F., 2010. Le développement durable. Armand Colin.

Mandelbrot, B.B., 1982. The fractal geometry of nature. Times Books.

Marik, V., Lazanskı, J., 2007. Industrial applications of agent technologies. Control Eng. Pr. 15, 1364–1380.

Marík, V., McFarlane, D., 2005. Industrial Adoption of Agent-Based Technologies. IEEE Intell. Syst. 20, 27–35.

Mataric, M.J., 1992. Minimizing complexity in controlling a mobile robot population, in Proceedings of IEEE International Conference On Robotics and Automation, 830–835.

Mati, Y., Xiaolan Xie, 2003. A polynomial algorithm for a two-job shop scheduling problem with routing flexibility. Presented at the 2003 IEEE International Conference on Robotics and Automation, 157–162.

Maturana, F., Shen, W., Norrie, D.H., 1999. MetaMorph: an adaptive agent-based architecture for intelligent manufacturing. Int. J. Prod. Res. 37, 2159–2173.

Mbobi, M., Boulanger, F., 2006. Le Paradigme Acteur Dans La Modelisation Des Systemes Embarques, in: Canadian Conference on Electrical and Computer Engineering, CCECE '06. Presented at the Canadian Conference on Electrical and Computer Engineering, 418–421.

McFarlane, D.C., Bussmann, S., 2003. 13. Holonic Manufacturing Control: Rationales, Developments and Open Issues. Agent-Based Manuf. Adv. Holonic Approach 303–326.

Mealy, G.H., 1955. A method for synthesizing sequential circuits. Bell Syst. Tech. J. 34, 1045–1079.

Ministère de l'Industrie, 2010. Technologies clés 2015 - 85 technologies clés dans sept secteurs économiques.

Monteiro, T., Ladet, P., 2001. Formalisation de la coopération dans le pilotage distribué des flux interentreprises, Application à une entreprise de production de biens. J. Eur. Syst. Autom. 35, 933–962.

Montratec AG, 2013. http://www.montratec.com (accessed 10.14.13).

Moore, E.F., 1956. Gedanken-experiments on sequential machines. Autom. Stud. 34, 129–153.

Morel, G., Valckenaers, P., Faure, J.M., Pereira, C.E., Diedrich, C., 2007. Manufacturing plant control challenges and issues. Control Eng. Pr. 15, 1321–1331.

Mp2cosmeticsolutions, 2011. http://www.mp2cosmeticsolutions.com (accessed 10.14.13).

Nagalingam, S.V., Lin, G.C.I., 2008. CIM—still the solution for manufacturing industry. Robot. Comput.-Integr. Manuf. 24, 332–344.

Novas, J.M., Van Belle, J., Saint Germain, B., Valckenaers, P., 2013. A Collaborative Framework between a Scheduling System and a Holonic Manufacturing Execution System, in: Service Orientation in Holonic and Multi Agent Manufacturing and Robotics, Studies in Computer Intelligence. Springer, 3–17.

Okino, N., 1989. Bionic manufacturing systems-modelon based approach. Presented at the CAM-I 18th Annual International Conference, New Orleans, Lousiana, 485–492.

Okino, N., 1993. Bionic manufacturing systems, in: Conference on Flexible Manufacturing Systems, Past, Present-Future (Ed: J. Peklenik), Ljubljana: Faculty of Mechanical Engineering. 73–95.

Oracle, 2012. http://fr.netbeans.org/ (accessed 10.14.13).

Ottaway, T.A., Burns, J.R., 2000. An adaptive production control system utilizing agent technology. Int. J. Prod. Res. 38, 721–737.

Ounnar, F., Pujo, P., 2010. Isoarchic and Multi-criteria Control of Supply Chain Network, in: Benyoucef, L., Grabot, B. (Eds.), Artificial Intelligence Techniques for Networked Manufacturing Enterprises Management. Springer London, 161–180.

Ounnar, F., Pujo, P., Mekaouche, L., Giambiasi, N., 2008. Integration of a flat holonic form in an HLA environment. J. Intell. Manuf. 20, 91–111.

Ou-Yang, C., Lin, J.S., 1998. The development of a hybrid hierarchical/heterarchical shop floor control system applying bidding method in job dispatching. Robot. Comput.-Integr. Manuf. 14, 199–217.

Pach, C., Bekrar, A., Zbib, N., Sallez, Y., Trentesaux, D., 2012a. An effective potential field approach to FMS holonic heterarchical control. Control Eng. Pr. 20, 1293–1309.

Pach, C., Berger, T., Sallez, Y., Adam, E., Trentesaux, D., 2013. Reactive and Energy-aware Scheduling of Flexible Manufacturing Systems Using Potential Fields. Comput. Ind., Special Issue on ICT for Sustainability in Industry, doi: 10.1016/j.compind.2013.11.008 .

Pach, C., Berger, T., Sallez, Y., Trentesaux, D., 2012b. Instantiation of the Open-Control Concept in FMS based on Potential Fields. Presented at the 8th Annual Conference of the IEEE Industrial Electronics Society, Montreal.

Pach, C., Berger, T., Sallez, Y., Trentesaux, D., 2013b. Effective, energy-aware control of a production system: a potentiel fields approach. Presented at the Intelligent Manufacturing System 2013, Sao Paulo.

Pach, C., Berger, T., Sallez, Y., Trentesaux, D., Deneux, D., 2012c. Holo-Gen : modèle de contrôle holonique générique- mise en oeuvre sur un SFP. Presented at the 13ème Colloque National AIP PRIMECA, Le Mont Dore.

Pach, C., Zambrano, G., 2011. Vers la résolution de la myopie, pour un pilotage performant des Systèmes Flexibles de Production ?, in: Journées Nationales MACS 2011. Marseille.

Pach, C., Zambrano, G., Adam, E., Berger, T., Trentesaux, D., 2011a. Roles-based MAS applied to the control of intelligent products in FMS. Lect. Note Comput. Sci., HOLOMAS 6867, 185–194.

Pach, C., Zambrano, G., Berger, T., Sallez, Y., Trentesaux, D., 2011b. Maitrise de la myopie des systèmes flexibles de production industriels, in: 12e Colloque National AIP PRIMECA. Le Mont Dore.

Parunak, H.V.D., Irish, B.W., Kindrick, J., Lozo, W., 1985. Fractal actors for distributed manufacturing control. Eng. Knowl.-Based Syst. 653–660.

Paschos, V., 2005. Optimisation combinatoire. 1, Concepts fondamentaux. Hermès Sci. Lavoisier Paris.

Pechoucek, M., Marík, V., 2008. Industrial deployment of multi-agent technologies: review and selected case studies. Auton. Agents Multi-Agent Syst. 17, 397–431.

Peterson, J.L., 1981. Petri net theory and the modeling of systems. Prentice-Hall Englewood Cliff NJ.

Petri, C.A., 1980. Introduction to general net theory, in: Net Theory and Applications. Springer, 1–19.

Pinedo, M.L., 2012. Scheduling: theory, algorithms, and systems. Springer.

Polajnar, J., Nalbandyan, N., Alemi, O., Polajnar, D., 2012. An Interaction Protocol for Mutual Assistance in Agent Teamwork, in: Complex, Intelligent and Software Intensive Systems (CISIS), 2012 Sixth International Conference On. pp. 6–11.

Pujo, P., Kieffer, J.P., 2002. Fondements du pilotage des systèmes de production. Hermès Science Publications.

Puterman, M.L., 1987. Dynamic Programming, in: Encyclopedia of Physical Science and Technology. Academic Press, 438–463.

Raileanu, S., Parlea, M., Borangiu, T., Stocklosa, O., 2012. A JADE Environment for Product Driven Automation of Holonic Manufacturing. Serv. Orientat. Holonic Multi-Agent Manuf. Cont., 265–277.

Rasmussen, J., 1986. Information Processing and Human-Machine Interaction. An Approach to Cognitive Engineering.

Reb, J., Connolly, T., 2009. Myopic regret avoidance: Feedback avoidance and learning in repeated decision making. Organ. Behav. Hum. Decis. Process. 109, 182–189.

Rolón, M., Martínez, E., 2012. Agent-based modeling and simulation of an autonomic manufacturing execution system. Comput. Ind. 63, 53–78.

Rutten, L., Valckenaers, P., 2013. Self-organizing Prediction in Smart Grids through Delegate Multi-Agent Systems, in: Trends in Practical Applications of Agents and Multiagent Systems. Springer, 95–102.

Ryu, K., 2004. Fractal-based reference model for self-reconfigurable manufacturing systems (PhD Thesis), Pohang University of Science and Technology.

Ryu, K., Jung, M., 2003. Agent-based fractal architecture and modelling for developing distributed manufacturing systems. Int. J. Prod. Res. 41, 4233–4255.

Sallez, Y., Bajic, E., 2007. Projet SCP http://scp-gdr-macs.cran.uhp-nancy.fr/ (accessed 10.14.13).

Sallez, Y., Berger, T., Deneux, D., Trentesaux, D., 2010. The lifecycle of active and intelligent products: The augmentation concept. Int. J. Comput. Integr. Manuf. 23, 905–924.

Sallez, Y., Berger, T., Trentesaux, D., 2009. A stigmergic approach for dynamic routing of active products in FMS. Comput. Ind. 60, 204–216.

Samsudin, K., Ahmad, F.A., Mashohor, S., 2011. A highly interpretable fuzzy rule base using ordinal structure for obstacle avoidance of mobile robot. Appl. Soft Comput. 11, 1631–1637.

Sauer, O., 2008. Automated engineering of manufacturing execution systems–a contribution to "adaptivity" in manufacturing companies, in: 5th International Conference on Digital Enterprise Technology Nantes, France.

Scattolini, R., 2009. Architectures for distributed and hierarchical Model Predictive Control – A review. J. Process Control 19, 723–731.

Senehi, M.K., Kramer, T.R., 1998. A framework for control architectures. Int. J. Comput. Integr. Manuf. 11, 347–363.

Shen, W., Hao, Q., Yoon, H.J., Norrie, D.H., 2006. Applications of agent-based systems in intelligent manufacturing: An updated review. Adv. Eng. Informatics 20, 415–431.

Shen, W.-M., Will, P., Galstyan, A., Chuong, C.-M., 2004. Hormone-inspired self-organization and distributed control of robotic swarms. Auton. Robots 17, 93–105.

Shivanand, H.K., Benal, M.M., Koti, V., 2006. Flexible manufacturing system. New Age International.

Skobelev, P., 2011. Multi-Agent Systems for Real Time Resource Allocation, Scheduling, Optimization and Controlling: Industrial Applications, in: Mařík, V., Vrba, P., Leitão, P. (Eds.), Holonic and Multi-Agent Systems for Manufacturing, Lecture Notes in Computer Science. Springer Berlin Heidelberg, 1–14.

Smith, R.G., 1980. The contract net protocol: High-level communication and control in a distributed problem solver. Comput. IEEE Trans. 100, 1104–1113.

Sorici, A., Picard, G., Boissier, O., Santi, A., Hübner, J.F., 2012. Multi-Agent Oriented Reorganisation within the JaCaMo infrastructure, in: Proceedings of The Third International Workshop on Iinfraestructures and Tools for Multiagent Systems: ITMAS. pp. 135–148.

Spreadshirt, 2002. http://www.spreadshirt.fr/ (accessed 10.14.13).

Staubli, 2007. http://www.staubli.fr/ (accessed 10.14.13).

Suárez, S., Leitao, P., Adam, E., 2013. Holonic Recursiveness with Multi-Agent System Technologies, in: Trends in Practical Applications of Agents and Multiagent Systems. Springer, 103–111.

Taghezout, N., Zaraté, P., 2008. Negotiation Process for Multi-Agent DSS for Manufacturing System, in: Zaraté, P., Belaud, J.P., Camilleri, G. (Eds.), Collaborative Decision Making: Perspectives and Challenges. P. IOS Press, 49–60.

Tawegoum, R., Castelain, E., Gentina, J.C., 1994. Hierarchical and dynamic production control in flexible manufacturing systems. Robot. Comput.-Integr. Manuf. 11, 327–334.

Tharumarajah, A., 1996. Comparison of the bionic, fractal and holonic manufacturing system concepts. Int. J. Comput. Integr. Manuf. 9, 217–226.

Tharumarajah, A., Wells, A.J., Nemes, L., 1998. Comparison of emerging manufacturing concepts, in: Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference On. pp. 325–331.

Thomas, P., Thomas, A., 2011. De la nécessité des bonnes informations dans les systèmes contrôlés par les produits, in: 7ème Conférence Internationale Conception et Production Intégrées, CPI'2011. Presented at the 7ème Conférence Internationale Conception et Production Intégrées, CPI'2011.

Trentesaux, D., 2002. Pilotage hétérarchique des systèmes de production, in: Habilitation à Diriger Les Recherches. Université de Valenciennes et du Hainaut-Cambrésis.

Trentesaux, D., 2009. Distributed control of production systems. Eng. Appl. Artif. Intell. 22, 971–978.

Trentesaux, D., Pach, C., Bekrar, A., Sallez, Y., Berger, T., Bonte, T., Leitão, P., Barbosa, J., 2013. Benchmarking Flexible Job-Shop Scheduling and Control Systems. Control Eng. Pr. 21, 1204–1225.

Trentesaux, D., Pesin, P., Tahon, C., 2000. Distributed artificial intelligence for FMS scheduling, control and design support. J. Intell. Manuf. 11, 573–589.

Trentesaux, D., Tahon, C., Ladet, P., 1998. Hybrid production control approach for JIT scheduling. Artif. Intell. Eng. 12, 49–67.

Ueda, K., 1992. A concept for bionic manufacturing systems based on DNA-type information, in: Proceedings of the IFIP TC5/WG5. 3 Eight International PROLAMAT Conference on Human Aspects in Computer Integrated Manufacturing. 853–863.

Ueda, K., Hatono, I., Fujii, N., Vaario, J., 2000. Reinforcement Learning Approaches to Biological Manufacturing Systems. CIRP Ann. - Manuf. Technol. 49, 343–346.

Ueda, K., Vaario, J., Ohkura, K., 1997. Modelling of biological manufacturing systems for dynamic reconfiguration. CIRP Ann.-Manuf. Technol. 46, 343–346.

Valckenaers, P., Van Brussel, H., Verstraete, P., Saint Germain, B., 2007. Schedule execution in autonomic manufacturing execution systems. J. Manuf. Syst. 26, 75–84.

Van Brussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., Peeters, P., 1998. Reference architecture for holonic manufacturing systems: PROSA. Comput. Ind. 37, 255–274.

Van Dyk, L., 2000. Manufacturing execution systems (PhD Thesis), University of Pretoria.

Vrba, P., Kadera, P., Jirkovskỳ, V., Obitko, M., Ma\vrík, V., 2011. New trends of visualization in smart production control systems, in: Holonic and Multi-Agent Systems for Manufacturing. Springer, 72–83.

Wago, 2013. http://www.wago.com (accessed 10.14.13).

Warnecke, H.J., 1993. The Fractal Company: A Revolution in Corporate Culture. Springer-Verlag Berlin and Heidelberg GmbH & Co. K.

Wilensky, U., 1999. Center for connected learning and computer-based modeling, Northwestern University. Evanston, IL. http://ccl.northwestern.edu/netlogo/ (accessed 10.14.13).

Wong, C.Y., McFarlane, D., Ahmad Zaharudin, A., Agarwal, V., 2002. The intelligent product driven supply chain, in: Systems, Man and Cybernetics, 2002 IEEE International Conference On.

Wooldridge, M., Jennings, N.R., Kinny, D., 2000. The Gaia methodology for agent-oriented analysis and design. Auton. Agents Multi-Agent Syst. 3, 285–312.

Wyns, J., 1999. Reference architecture for Holonic Manufacturing Systems-the key to support evolution and reconfiguration (PhD Thesis), Katholieke Universiteit Leuven.

Yang, T., Ma, J., Hou, Z.G., Peng, G., Tan, M., 2008. A Multi-agent Architecture Based Cooperation and Intelligent Decision Making Method for Multirobot Systems, in: Neural Information Processing, 376–385.

Zambrano, G., Aissani, N., Bekrar, A., Trentesaux, D., 2012. A Holonic Approach to Myopic Behavior Correction for the Allocation Process in Flexible-Job Shops Using Recursiveness. Serv. Orientat. Holonic Multi-Agent Manuf. Control, 115–128.

Zambrano, G., Pach, C., Aissani, N., Bekrar, A., Berger, T., Trentesaux, D., 2013. The control of myopic behavior in semi-heterarchical production systems: A holonic framework. Eng. Appl. Artif. Intell. 26, 800–817.

Zambrano, G., Pach, C., Aissani, N., Berger, T., Trentesaux, D., 2011. An approach for temporal myopia reduction in Heterarchical Control Architectures, in IEEE International Symposium on Industrial Electronics, 1767–1772.

Zbib, N., Pach, C., Sallez, Y., Trentesaux, D., 2012. Heterarchical production control in manufacturing systems using the potential fields concept. J. Intell. Manuf. 23, 1649–1670.

ANNEXES

Table des Matières Annexes

Table des Figures Annexes

Table des Tableaux Annexes

Annexe 1 : Illustrations de l'environnement changeant des systèmes de production

Annexe 1.1 : Mass customization

La *mass customization* est aussi appelée « sur-mesure de masse » (Da Silveira et al., 2001). Ce principe est défini par l'ABC NetMarketing (ABC NetMarketing, 1997) de la manière suivante.

> La *mass customization* consiste à proposer un produit de consommation relativement courante créé sur mesure en fonction des caractéristiques et préférences de chaque acheteur.
>
> ABC Netmarketing,

Cette démarche, pour réussir, ne doit pas engendrer de surcout par rapport au produit standard (non sur mesure). Généralement, cette pratique est associée à internet. Ainsi, la commande et la personnalisation du produit par le client peuvent par exemple être automatiquement envoyées vers le système de production sans obligatoirement passer par un intermédiaire humain. Ceci peut concerner des vêtements personnalisés comme la création de tee-shirt (spreadshirt 2002) ou des produits cosmétiques (Mp2cosmeticsolutions, 2011). Mais le meilleur exemple est certainement celui de l'automobile. Ainsi, une voiture fabriquée à des milliers d'exemplaires peut comporter des millions de variantes possibles au niveau du type de motorisation, de couleur de carrosserie, d'intérieur, de jantes ou encore d'équipements optionnels (ex : lecteur CD, climatisation). Afin de permettre cette personnalisation, le système de production doit pouvoir être paramétré, s'adapter et réaliser différents produits sans modification majeure de son architecture de pilotage. La figure A1 présente un exemple de site internet permettant de paramétrer, et donc personnaliser, sa future voiture.

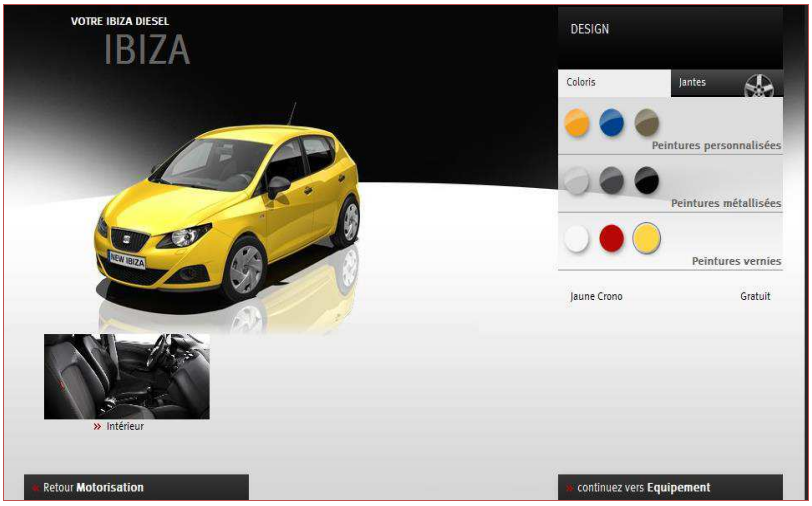


Figure A.1– Exemple de site internet permettant de personnaliser son produit

Annexe 1.2 : L'énergie

Le comportement des industriels au regard de l'énergie change pour plusieurs raisons. Tout d'abord, l'énergie qui était une constante du système avec les énergies fossiles devient une variable avec les énergies renouvelables. En effet, la production d'énergie renouvelable est la plupart du temps variable (selon l'ensoleillement, le vent) et difficilement stockable. Ainsi le système de production doit, par exemple, pouvoir adapter sa consommation à l'énergie disponible.

Ensuite les entreprises ont désormais comme critère de plus en plus critique de minimiser leur consommation d'énergie. D'une part car le coût de l'énergie ne cesse d'augmenter et que la part du coût énergétique dans le coût total de fabrication d'un produit devient de plus en plus importante. D'autre part car les clients sont de plus en plus sensibles aux entreprises qui respectent l'environnement, notamment d'un point de vue énergétique. C'est ce qui est appelé le *green manufacturing* (Huang, 2012). De plus, l'aspect respect de l'environnement par les entreprises est d'autant plus important que les collectivités locales et nationales, favorisent ces entreprises et rémunérèrent cette prise de conscience environnementale. Ceci accentue encore l'importance de l'énergie dans les stratégies de décisions de l'entreprise.

Annexe 2 : Illustration de la myopie

Annexe 2.1 : Myopie individuelle

Un exemple de myopie individuelle inspiré de est présenté dans la figure A.2.

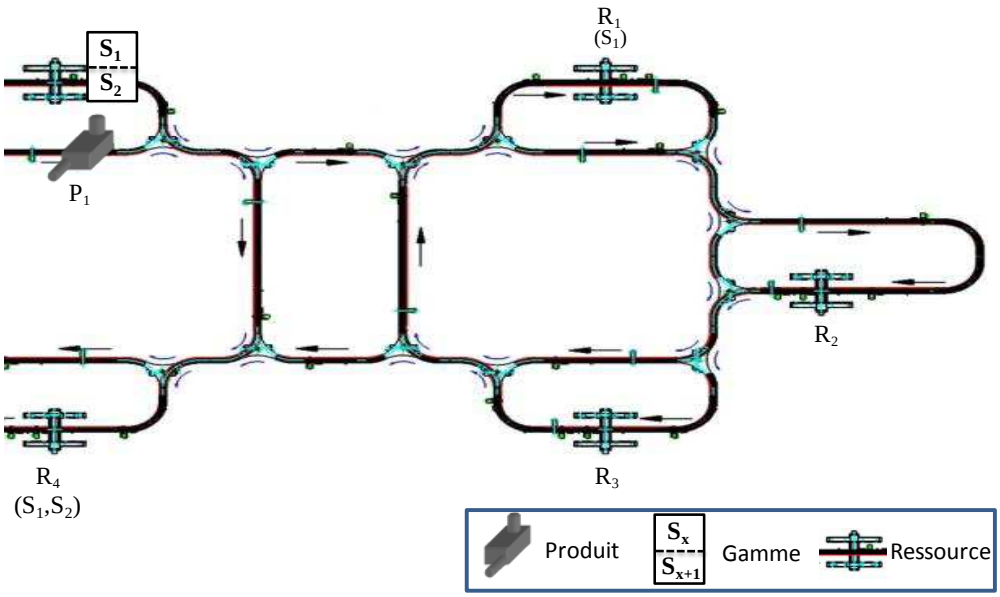


Figure A.2– Exemple de Myopie individuelle : situation initiale

Sur cette figure, le produit P1 a besoin des services S1 et S2. Sur les 4 ressources/machines présentes dans la cellule, R4 est la seule ressource à fournir le service S2 alors que R1 et R4 peuvent fournir S1. Les flèches représentent le sens de circulation des produits. P1 (compte tenu de sa gamme de fabrication) doit se diriger vers R1 ou R4. Les deux ressources sont libres et comme le produit ne cherche qu'à obtenir S1 à l'instant t, il choisit, par exemple, R1. Une fois que le service S1 est fourni, la cellule se trouve dans la situation présentée figure A3.

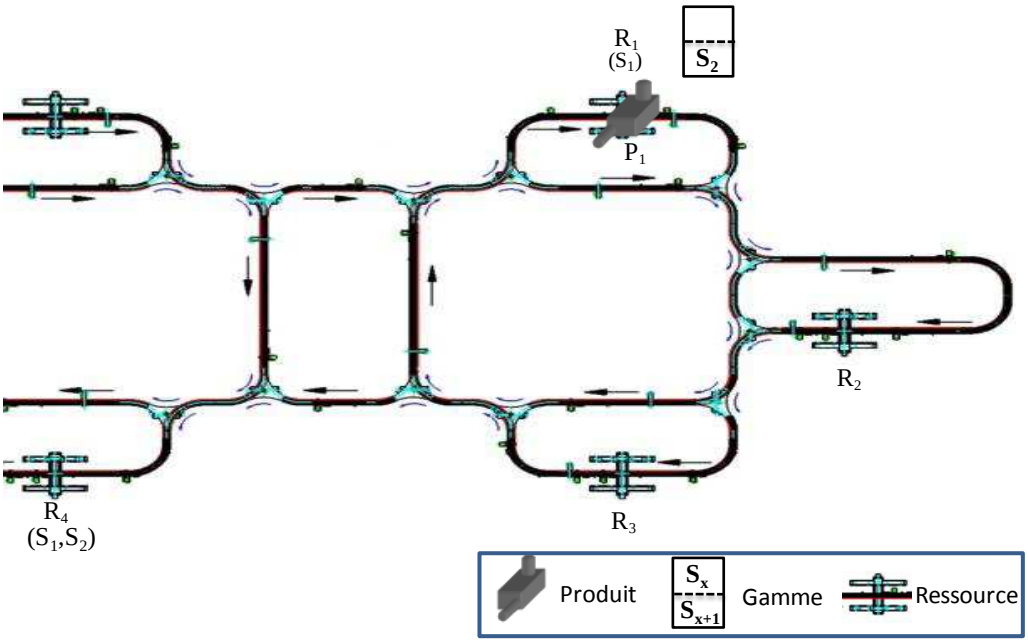


Figure A.3– Exemple Myopie individuelle : service S1 fourni

Le produit cherche désormais à obtenir le service S_2 ; hors comme R_4 est la seule ressource pouvant fournir ce service, le produit va devoir traverser toute la cellule pour l'obtenir. Si le produit avait pris en compte l'information concernant le second service à obtenir, il aurait dû choisir directement R_4 qui lui aurait permis d'obtenir tous les services souhaités. Ici le produit a pris une mauvaise décision car il n'a pas pris en compte son futur. Il s'agit de la myopie individuelle.

Annexe 2.2 : Myopie collective

Un exemple de myopie collective est présenté dans la figure A.4.

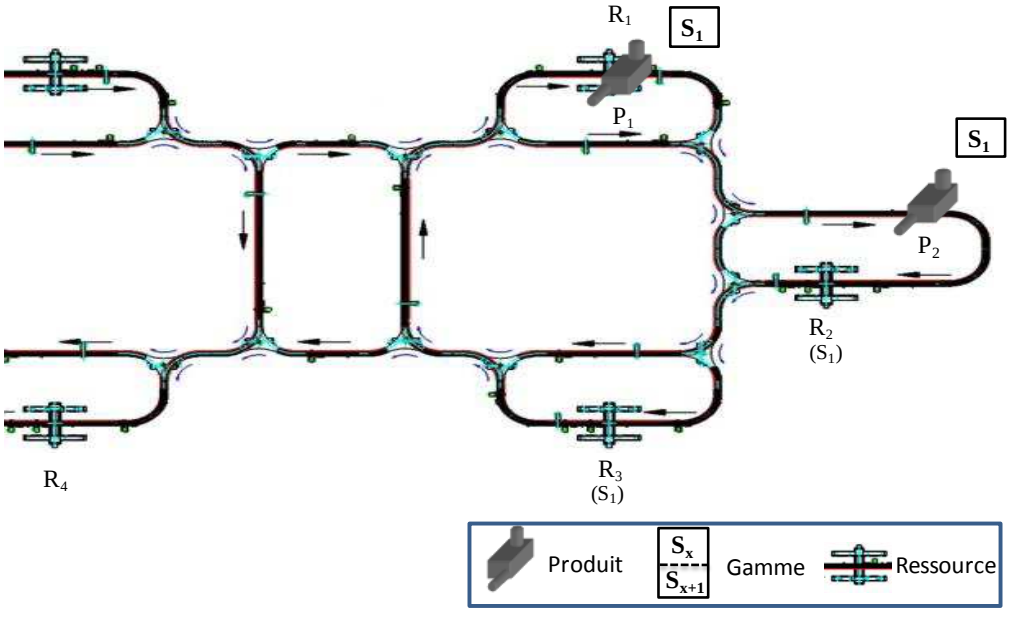


Figure A.4– Exemple de Myopie collective : situation initiale

Ici deux produits P_1 et P_2 ont besoin du service S_1 qui peut être fourni par R_2 et R_3. Les produits n'interagissent pas entre eux (i.e. aucune communication directe ou indirecte). Le produit P_2 se dirige vers R_2. P_1 vient d'obtenir son service précédent et doit maintenant choisir, pour obtenir son nouveau service S_1, entre R_2 et R_3. Il ne se base que sur l'état des ressources puisqu'il n'a aucune information sur les autres produits du système. Étant donné que P_2 n'est pas encore arrivé sur R_2, cette ressource est libre tout comme R_3. P_1 choisit R_2 qui est la ressource la plus proche. Cependant, quelques instants plus tard survient la situation présentée figure A.5.

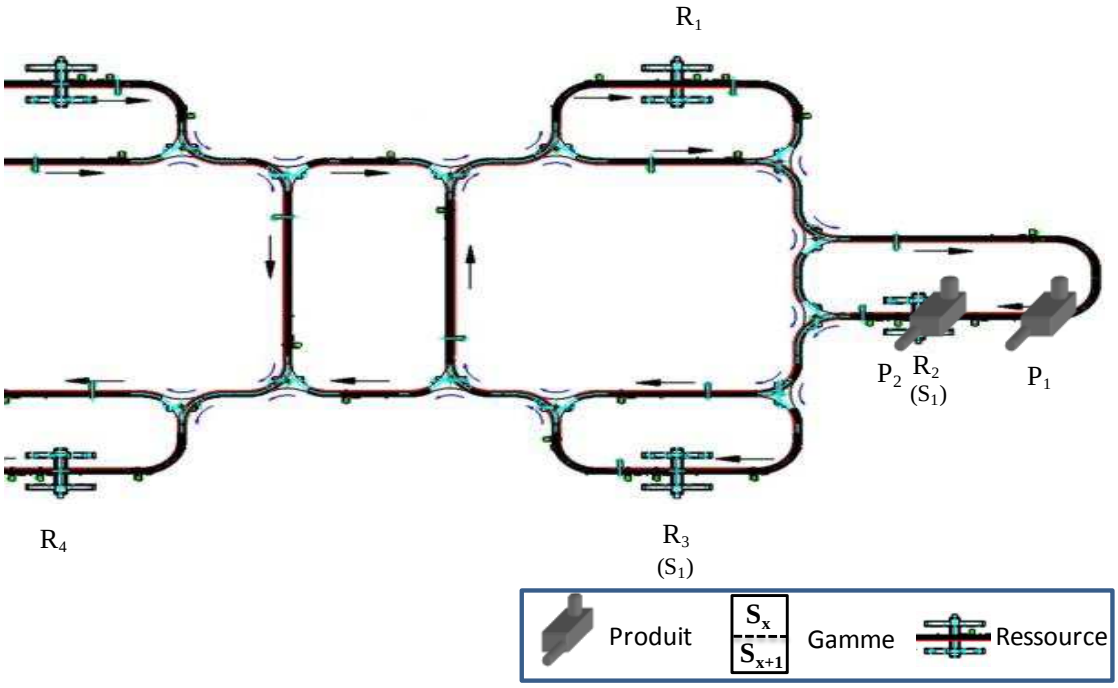


Figure A.5– Exemple de Myopie collective : produit P_1 en attente

P_1 s'est dirigé vers R_2 mais entre temps P_2 est arrivé sur R_2. Ainsi P_1 est bloqué dans la file d'attente de R_2 alors qu'il aurait pu obtenir son service sur R_3 qui est toujours libre. Sa décision était donc mauvaise car il n'avait pas l'information sur le futur de P_2 et sur sa destination. Il s'agit ici de la myopie collective.

Annexe 3 : Les fonctions principales entre niveau stratégique et système physique

Cette annexe présente les fonctions principales qui permettent le lien entre niveau stratégique et système physique. Celles-ci seront en effet impactées par le travail de thèse.

La fonction ordonnancement joue un rôle crucial au niveau des performances des systèmes de production (Pinedo, 2012). L'ordonnancement a été notamment défini dans la littérature dans (Conway et al., 2003; Cox et al., 1995; Pinedo, 2012). Pinedo le définit comme :

> L'allocation des ressources aux tâches sur une durée prédéfinie en ayant pour but d'optimiser un ou plusieurs objectifs.
>
> Pinedo, 2012.

Ainsi l'ordonnancment permet d'optimiser la production au regard des objectifs du système et est donc en grande partie responsable de l'efficacité du système. Il est de plus directement impacté par les choix du niveau stratégique (notamment la plannification globale cf. (Pinedo, 2012)). Il doit donc être intègré à ce travail de thèse visant à adapter les performances du système conformément aux nouvelles stratégies industrielles présentées dans l'introduction.

La fonction conduite est ensuite responsable de la mise en oeuvre de l'ordonnancement et fait le lien entre celui-ci et la commande. La fonction conduite est notamment définie par (Aissani, 2010; Cox et al., 1995; Dindeleux, 1992). Aissani la définit sous les termes suivants.

> La conduite correspond au niveau décisionnel du pilotage dont le rôle principal est la mise en œuvre des tâches ordonnancées conformément aux décisions planifiées par le niveau prévisionnel. Elle assure la flexibilité quotidienne pour faire face aux fluctuations du système.
>
> Aissani, 2010.

En cas de fluctuations dans le système (perturbations ou incertitude), la conduite adapte localement l'ordonnancement afin de correspondre à la nouvelle situation. Ainsi dans un travail de thèse devant permettre de réagir à l'incertitude la prise en compte et l'étude de cette fonction est primordiale.

Finalement, la dernière fonction impactée par ce travail de thèse est la commande. Elle est notamment définie par (Aissani, 2010;Cox et al., 1995; Dindeleux, 1992). Aissani la définit de la manière suivante.

> La fonction commande décode les tâches à exécuter et envoie les commandes au système physique. Elle englobe notamment la commande des équipements de production.
>
> Aissani, 2010.

Cette fonction permet donc de commander le système physique afin de suivre les ordres provenant de l'ordonnancement et de la conduite. C'est la fonction de plus bas niveau permettant d'actionner les éléments physiques du système de production (comme par exemple la sortie d'un vérin ou la mise en rotation d'un moteur électrique).

Annexe 4 : Modélisation des entités des systèmes de production

Cette annexe présente les différents modèles d'entités existants dans la littérature des systèmes de production.

Annexe 4.1 : État de l'art sur les modèles d'entités

Le modelon

Le premier modèle présenté est le modelon, qui a été proposé par Okino (Okino, 1993, 1989) et Ueda (Ueda, 1992; Ueda et al., 1997). Un modelon représente l'entité de base des *Bionic Manufacturing System* (BMS) et se fonde sur les principes biologiques tels que l'auto-organisation, l'auto-développement, l'adaptation et l'évolution (Ryu, 2004). Il est composé de modelons de niveau inférieurs, d'opérateurs (enzymes) et d'une mémoire ou environnement commun(e). Certains BMS utilisent la notion de cellule qui est très proche d'un modelon mais dont les fonctions sont représentées par des informations biologiques (Ueda et al., 2000).

Le fractal

Les *Fractal Manufacturing Systems* (FrMS) utilisent un autre modèle : le fractal. Ce terme se base sur un concept géométrique inventé par Mandelbrot (Mandelbrot, 1982) qui a été pour la première fois introduit dans le domaine manufacturier par Warnecke (Warnecke, 1993) sous le nom de *Fractal Factory*. Il définit le fractal de la manière suivante.

> Un fractal est une entité sociale agissant de manière indépendante dont les buts et la performance peuvent être précisément décrits.
>
> Warnecke, 1993.

Les caractéristiques principales d'une *Fractal Factory* sont les suivantes (Ryu et Jung, 2003) :

- Les fractals sont auto-similaires (décomposition tout-partie) et fournissent des services,
- Les fractals utilise l'auto-organisation,
- Le système de buts (buts sociaux) qui émerge des buts individuels de chaque fractal ne contient aucune contradiction. Les buts sociaux doivent être atteints,
- Les fractals sont mis en réseau via un système d'information et de communication efficace,

- La performance d'un fractal est soumis à une estimation et une évaluation constantes.

L'acteur

Un autre modèle présent dans la littérature est la modélisation orientée acteurs. Ce paradigme est issus des travaux de Agha (Agha, 1997; Agha et al., 1997) même si le terme acteur a été utilisé pour la première fois par Hewitt (Hewitt, 1977) pour représenter une entité intelligente autonome dans le domaine de l'intelligence artificielle. Un système d'acteurs est un système dans lequel chaque composant autonome communique avec un ensemble d'autres composants qui lui sont connectés (Mbobi et Boulanger, 2006). Le système est décomposé d'un point de vue de ses actions.

L'agent

Un autre modèle très présent dans la littérature est l'agent, qui est utilisé dans les *Multi-Agent Systems* (MAS). Un agent est défini par Erceau (Erceau et Ferber, 1991) comme une entité ayant :

- La capacité d'agir sur elle-même et sur son environnement,
- La capacité d'utiliser une représentation partielle de cet environnement,
- La capacité de communiquer avec d'autres agents,
- La capacité de poursuivre un objectif individuel.

Selon Jacques Ferber (Ferber, 1995), un MAS est un ensemble d'agents représentant les entités du système sous forme d'entités informatiques. Le MAS permet aux agents de posséder les caractéristiques suivantes (Boufaied, 2003) :

- **Autonomie :** un agent peut opérer sans intervention externe et exerce un certain contrôle sur son comportement.
- **Coopération :** les agents interagissent entre eux afin d'accomplir un but commun.
- **Réactivité et Pro-activité :** les agents perçoivent leur environnement et répondent rapidement aux changements qui s'y opèrent. Les agents sont aussi capables de prendre des initiatives afin de contrôler leur comportement.
- **Adaptation** : les agents peuvent s'adapter à l'évolution de leur architecture de contrôle pour continuer à exécuter leur comportement.

L'autonomic element

Le concept *d'autonomic computing* et *d'autonomic elemen*t associé a été proposé par IBM en 2001 comme une perspective pour la technologie de l'information (Horn, 2001). Le principe étant de proposer des systèmes informatiques qui se gèrent eux-mêmes à partir des objectifs de haut niveau données par l'administrateur. *L'autonomic computing* est basé sur quatre fonctionnalités (Kephart et Chess, 2003) :

- **Auto-configuration** : les composants (*autonomic elements*) se configurent automatiquement et les systèmes suivent des règles de haut-niveau. Le reste du système s'adapte automatiquement et harmonieusement.
- **Auto-optimisation :** Les composants et les systèmes cherchent continuellement des opportunités pour améliorer leur performance et leur efficacité.

- **Auto-réparation :** Le système détecte, diagnostique et répare automatiquement les problèmes logiciel et matériel.
- **Auto-protection :** Le système se défend automatiquement contre les attaques malveillantes ou les pannes en cascade. Il utilise des avertissements afin d'anticiper et de prévenir les pannes globales du système.

Le holon

Le dernier modèle de la littérature est le holon, utilisé dans les *Holonic Manufacturing Systems* (HMS). Les principes holoniques sont basés sur les travaux d'Arthur Koestler (Koestler, 1967) qui a défini un holon comme une entité qui est à la fois une partie et un ensemble et qui est capable de créer des structures complexes avec d'autres holons. Ce paradigme a été adapté au domaine de la production par le consortium IMS (*Intelligent Manufacturing Systems*). De cette adaptation quatre définitions caractérisant les HMS ont été proposées :

- **Holon** : Élément autonome et coopératif d'un système manufacturier qui permet de transformer, transporter, stocker et/ou valider de l'information et des objets physiques. Le holon est composé d'une partie informationnelle et d'une partie physique. Un holon peut être une partie d'un autre holon (principe de récursivité).
- **Autonomie** : Capacité d'un holon à créer et contrôler l'exécution de ses plans et/ou stratégies.
- **Coopération** : Processus par lequel un groupe de holons développe mutuellement des plans acceptables et les exécute.
- **Holarchie :** Ensemble de Holons qui coopèrent pour atteindre des objectifs communs. L'holarchie définit les règles de base pour la coopération des holons et limite donc leur autonomie.

La dualité entre l'individu (le holon, qui est autonome et possède ses propres objectifs) et la communauté (l'holarchie, qui limite l'autonomie pour améliorer le comportement global afin d'atteindre un objectif global) à l'origine de la myopie est ici clairement identifiée.

Outre ces caractéristiques, deux éléments distinguent les holons : la prise en compte des interactions avec l'humain et la volonté d'être isomorphe au système réel. La figure A.6 extraite de (McFarlane et Bussmann, 2003) illustre ces deux éléments en présentant l'architecture générique d'un holon.

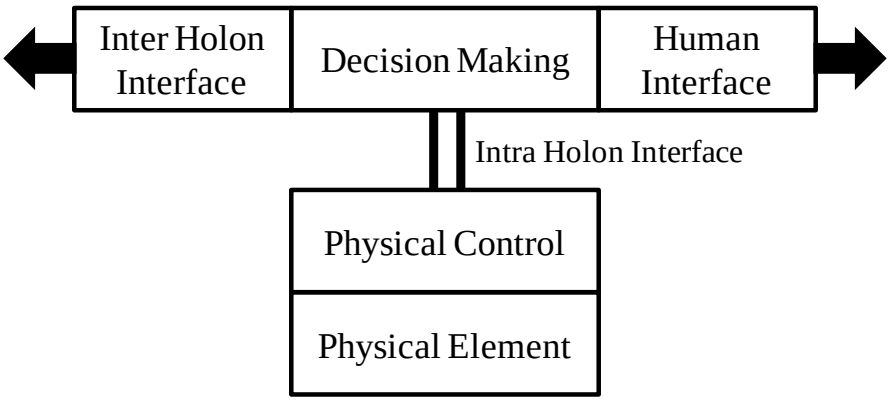


Figure A.6– Architecture générique d'un holon

La partie suivante justifie le choix du modèle holonique notamment en le comparant avec les autres modèles existants.

Annexe 4.2 : Comparaison des modèles

La table A.I inspirée de (Tharumarajah et al., 1998) résume les caractéristiques principales de tous les modèles présentés dans l'annexe 4.1.

Table A.I– Comparaison des caractéristiques des modèles

	BMS	FrMS	HMS	Autonomic Computing	Acteur	MAS
Entités	**Modelon ou Cellule**	**Fractal**	**Holon**	**Autonomic Element**	**Acteur**	**Agent**
Autonomie	auto-developpement, adaptation et évolution	définition de ses propres buts et adaptation via la vitalité	Capable de créer et contrôler l'exécution de ses plans et stratégies	auto-configuration, auto-réparation, auto-optimization et auto-protection	Suite à un message reçu, peut prendre des décisions locales	Peut opérer sans intervention externe et exerce un certain contrôle sur son comportement
Coopération	Au travers de l'environnement partagé des cellules et coordinateurs enzymatiques	Le but du système émerge des buts individuels de chaque fractal via négociation	Développement mutuel de plans acceptables au sein de l'holarchie	Interactions au travers de l'autonomic computing framework	Au travers d'échanges de messages asynchrones	Intéractions entre agents afin d'accomplir un but commun sauf en cas de compétition
Récursivité	Composé de modelon de niveaux inférieures et d'enzymes	auto-similarité	entité qui est à la fois une partie et un ensemble composé d'autres holons	-	-	-
Intégration directe du Système Physique dans le modèle	Représentation des éléments physiques par des modelons	Réprésentation des éléments physique par des sous-fractals	Représentation des éléments physiques par des holons terminaux physiques	-	-	-

Les modèles basés modelon (ou cellule), fractal et holon ont des caractéristiques et des objectifs assez proches comme le démontrent différentes comparaisons de la littérature (Ryu, 2004; Tharumarajah, 1996; Tharumarajah et al., 1998). Ainsi même si ces modélisations viennent de domaines différents tels que la biologie (Modelon), les mathématiques (Fractal) et l'étude du comportement (Holon), leurs caractéristiques sont similaires.

Les autres modèles (*autonomic element*, acteur et agent) ont une vision plus « informatique » des entités et permettent le lien direct avec la programmation de celles-ci. Ainsi ils ne sont pas à mettre en concurrence avec les trois modèles comparés ci-dessus mais sont complémentaires et utilisables lors de la mise en œuvre. Par exemple, les modèles holonique et agent partagent beaucoup de caractéristiques (Giret et Botti, 2004). Ainsi, il arrive fréquemment que les MAS soient utilisés lors de la mise en œuvre d'un HMS et certains chercheurs ont regroupé ce type d'approche sous le nom *Holonic Multi-Agent Systems* (Fischer et al., 2003). Certains auteurs comme (Andreev et al., 2003; Skobelev, 2011) considèrent d'ailleurs que les systèmes holoniques fournissent aux systèmes multi-agent une solide ossature générique pour des agents bio-inspirés et une structure avancée pour réaliser des « systèmes de systèmes » complexes.

Annexe 5 : La pyramide ERP-MES-Shop Floor

Les ERP et MES sont présentés respectivement dans (Leon, 2007) et (Van Dyk, 2000). Le *Shop Floor*, en français Niveau Atelier, correspond au système piloté de la figure 1.4. Le niveau *Shop Floor* peut être décomposé en fonction de l'importance de certains éléments. La figure A.7 présente un cas particulier de cette pyramide décomposant le niveau bas (niveau Atelier/*Shop Floor*) en deux sous niveaux : un niveau contrôle automate programmable industriel (PLC/PAC) et un niveau pour les capteurs et actionneurs. Un niveau supervision (SCADA présenté dans (Boyer, 2009)) est dans cet exemple intercalé entre le MES et le système piloté.

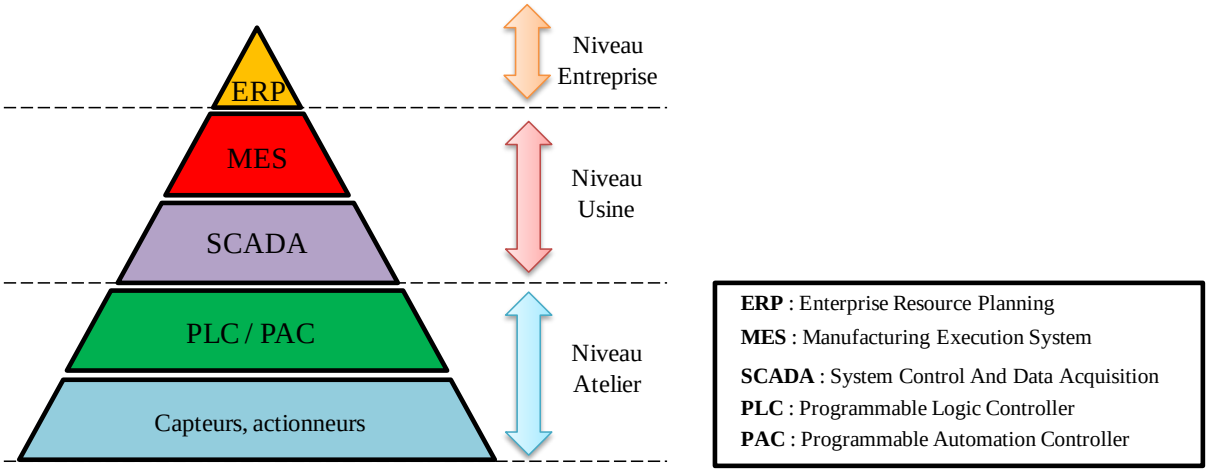


Figure A.7– Exemple de pyramide CIM

Dans cette pyramide, l'ERP gère les aspects financiers, le cycle de vie des produits et/ou la prise en compte des besoins clients (contrôle global sur un horizon temporel long). Le MES fait le lien entre l'ERP et le contrôle de bas niveau du système de production et prend en charge la fonction pilotage (allocation des ressources, routage des produits), s'occupe de la logistique ou encore de l'analyse des performances du système. Le niveau SCADA réalise la supervision du système, les PLCs et les capteurs & actionneurs réalisent eux les contrôles/commandes de bas niveau.

Annexe 6 : Le processus décisionnel de basculement considéré comme atomique

Le processus décisionnel, déterminant si le basculement d'une entité est nécessaire, est inspiré des travaux de Rasmussen et HOC (Hoc et Amalberti, 1994; Rasmussen, 1986). Selon ces auteurs, un processus décisionnel peut être décomposé en cinq étapes comme illustré figure A.8.

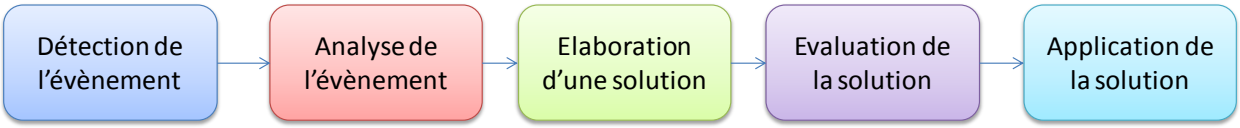


Figure A.8– Processus décisionnel (inspiré de Hoc et Amalberti, 1994; Rasmussen, 1986).

Dans cette thèse, ces cinq étapes sont considérées comme indissociables et localisées au sein de la même entité. Ainsi le processus décisionnel de basculement est considéré comme atomique.

Annexe 7 : Rapport entre complexité et avancée technologique (Garey et Johnson 1979)

La table A.II illustre le rapport entre complexité d'un problème et gain apporté par l'augmentation de la puissance de l'ordinateur résolvant ce problème. Les **Ni** représentent la taille de la plus grosse instance du problème solvable en 1 heure. Ainsi lorsque le problème est de complexité **N,** utiliser un ordinateur 1000 fois plus rapide permet de traiter des instances du problème de taille 1000 fois plus importante, toujours en 1 heure. Par contre si le problème est de complexité 3^N un ordinateur 1000 fois plus rapide ne permettra que d'augmenter la taille de l'instance solvable en 1 heure de 6.

Table A.II– Complexité et avancée technologique

Complexité	Ordinateur de référence	Ordinateur 100 fois plus rapide	Ordinateur 1000 fois plus rapide
N	N1	100*N1	1000*N1
N^2	N2	10*N2	31,6*N2
N^3	N3	4,64*N3	10*N3
2^N	N4	N4+6,64	N4+9,97
3^N	N5	N5+4,19	N5+6,29

L'exploration complète de l'espace des solutions d'un problème d'ordonnancement d'un Job-Shop Flexible est exponentielle. Ainsi la résolution de ce problème est positionnée dans les deux dernières lignes de ce tableau.

Annexe 8 : Modèle linéaire

$Z = min\ Cmax,$

s.t. $Cmax \geq t(j,h) + PT(j,h)$

$t(j,h) + PT(j,h) + \sum_{i,x} Tr(j,h,i,x) * TT(i,x) \leq t(j,h_{+1})$

$Tm(i,l_{+1}) \geq Tm(i,l) + \sum_{j,h} PT(j,h) * X(i,j,h,l)$

$Tr(j,h,i,x) = \begin{cases} 1\ if\ y(i,j,h) + y(x,j,h_{+1}) = 2 \quad (\forall\ x \neq i\ ) \\ 0 \text{ sinon} \end{cases}$

$y(i,j,h) \leq a(i,j,h)$

$\sum_{j,h} X(i,j,h,l) \leq 1$

$\sum_{i} y(i,j,h) = 1$

$\sum_{i} X(i,j,h,l) = 1$

$Tm(i,l) \leq t(j,h) + \big(1 - X(i,j,h,l)\big) * BN$

$Tm(i,l) + \big(1 - X(i,j,h,l)\big) * BN \geq t(j,h)$

$\sum_{l} X(i,j,h,l) = y(i,j,h)$

$t(j,h) + BN * b(j,h,k,m) \leq t(k,m) + BN$

$\sum_{j} z(j,k) \leq MJ - 1 \quad (\forall\ j \neq k\ )$

$z(j,k) \geq b(k,0,j,u) + b(j,0,k,0) - 1$

$z(j,k) \leq 1 - b(k,0,j,0) + b(j,u,k,u)$

$z(j,k) \geq b(k,0,j,0) + b(j,u,k,u) - 1$

$Tr(j,h,i,x), y(i,j,h), X(i,j,h,l), b(j,h,k,m), z(j,k), t(j,h), Tm(i,l), Cmax \in \mathrm{N}$

$Tr(j,h,i,x), y(i,j,h), X(i,j,h,l), b(j,h,k,m), z(j,k) \in [0, 1]$

$t(j,h), Tm(i,l), Cmax \geq 0$

Annexe 9 : Intégration de l'énergie dans ORCA-FMS en simulation

Cette annexe présente les simulations ayant permis de valider la possibilité de prise en compte de la consommation énergétique du système dans la couche de contrôle local d'ORCA-FMS. La première partie de cette annexe présente tout d'abord le réglage des paramètres utilisé dans les simulations présentées dans cette annexe. Puis la deuxième partie présente l'étude de contrôlabilité des indicateurs de performance et la troisième l'étude pour des simulations « à grande échelle ».

Annexe 9.1 : Hypothèses, paramètres et indicateurs de performances

Plusieurs hypothèses sont faites dans les simulations présentées dans cette annexe. Tout d'abord, les ordres des clients ne sont pas prévus à l'avance, empêchant tout ordonnancement prédictif. Deuxièmement, il n'y a pas de problème de qualité sur les produits, le résultat de l'inspection finale est donc toujours positif. Troisièmement, la consommation de chaque ressource est considérée comme constante tant que la ressource se trouve dans le même état de fonctionnement. De plus, la consommation pour fournir un service est la même pour toutes les ressources pouvant fournir ce service. Quatrièmement, la consommation de R_1 et R_5 dans les états « prête » et « en travail » est négligée de part la nature de ces ressources (respectivement pneumatique et système de vision). Ainsi les consommations considérées sont celles de R_2, R_3 et R_4. Et finalement, il est défini que les ressources utilisées dans ces simulations ne nécessité pas de temps de préparation pour fournir leurs services ($\lambda = 0$).

La table A.III présente le paramétrage utilisé pour le modèle de champs de potentiel.

Table A.III– Paramétrage du modèle de champs de potentiel.

Paramètre	Valeur	Commentaires
μ_r	200	Permet aux champs de se propager dans toute la cellule
μ_p	200	Permet aux champs de se propager dans toute la cellule
$\tau_{h,r}$	-	Ajusté pour chaque scénario
$\tau_{l,r}$	-	Ajusté pour chaque scénario
λ	0	Aucun temps de préparation considéré
Z_{1r}	0	Les machines éteintes n'émettent pas de champs de potentiel
Z_{2r}	1	Les machines en veille attirent faiblement les produits
Z_{3r}	5	Les machines prêtes attirent fortement les produits pour favoriser les machines déjà réveillées
Z_{4r}	$5 * (1 – (Q_r(t) / Q_{r\_max}))$	Même attractivité que pour Z_{3R} mais celle-ci est diminuée par les produits sur la machine ou en file d'attente.
Z_{1p}	0	Les produits hors du système n'émettent pas de champs
Z_{2p}	0	Les produits en veille n'émettent pas de champs
Z_{3p}	1	Les produits émettent un champ lorsqu'ils sont actifs
Z_{4p}	1	Les produits émettent un champ lorsqu'ils sont en cours d'assemblage.

Selon le modèle de champs de potentiel présenté chapitre II, les paramètres de contrôle sont $\tau_{h,r}$ et $\tau_{l,r}$. Les performances du système en fonction de ses paramètres sont étudiées par trois indicateurs de performances :

- le Cmax, qui évalue le temps pris pour compléter la production (efficacité),
- la consommation d'énergie globale (efficience),
- le nombre de basculement entre les états d'une ressource nécessitant une coupure ou une mise en marche de l'alimentation de puissance. Cet état est mesuré d'une part pour prévenir les dégâts en cas de ressources qui peuvent être endommagées par des basculements successifs, et d'autre part il permet d'avoir une idée de la nervosité du système (cf. (Barbosa et al., 2012a)).

La partie suivante présente la première série de simulation (SA1) correspondante à l'étude de contrôlabilité du modèle de champs de potentiel. Cette étude a pour objectif de montrer que les résultats en termes de Cmax, consommation énergétique et nombre de basculements peuvent être contrôlés à partir des valeurs de $\tau_{h,r}$ et $\tau_{l,r}$.

Annexe 9.2 : SA1- étude de la contrôlabilité des performances

Ces simulations sont effectuées en utilisant l'ordre de fabrications « AIPAIP » (il s'agit des ordres de fabrication #B0 du benchmark (Trentesaux et al., 2013)). Les seuils des ressources ($\tau_{h,r}$ et $\tau_{l,r}$) sont fixés à des valeurs entre 0 (aucun « champ produit » perçu) et 200 (le champs est maximum), par pas de 25. Le pas est affiné entre 190 et 200 pour certains scenarios afin d'obtenir des solutions spécifiques. Les résultats obtenus sont fournis dans la table A.IV.

Pour les colonnes correspondant aux seuils, si seulement une valeur est fournie, cela signifie que les stations sont réglées sur le même seuil (i.e. $\tau_{l,R2}= \tau_{l,R3}= \tau_{l,R4}$). Si trois valeurs sont

fournies, elles correspondent respectivement aux valeurs de $\tau_{l,R2}$, $\tau_{l,R3}$, et $\tau_{l,R4}$ (comme à partir de la simulation #50 table A.IV). Les temps sont donnés en seconde et les consommations en Wattheure. La colonne #1 représente le numéro de la simulation. Les colonnes #2 à #7 présentent les valeurs des seuils testées. La colonne 8 donne le makespan obtenu. Les colonnes #9, #10 et #11 fournissent les consommations de chaque ressource, alors que les colonnes #12 et #13 fournissent la consommation en Wattheure et en pourcentage du scenario initial (simulation #1), respectivement. Les colonnes #14, #15 et #16 présentent le nombre de basculement de chaque ressource. Finalement, la dernière colonne #17 fournit la somme de tous ces basculements. Les cellules colorées représentent les sorties des simulations.

Table A.IV– Résultats des simulations de contrôlabilité

Simulation	$\tau_{h,R2}$	$\tau_{h,R3}$	$\tau_{h,R4}$	$\tau_{l,R2}$	$\tau_{l,R3}$	$\tau_{l,R4}$	Cmax (s)	Conso. R2 (W.h)	Conso. R3 (W.h)	Conso. R4 (W.h)	Conso. Globale (W.h)	Conso. Globale (%)	Bascul. R2	Bascul. R3	Bascul. R4	Total des Bascul.
#1	**0**			**0**			**333**	**32.29**	**30.63**	**31.46**	**94.38**	-	**1**	**1**	**1**	**3**
#2	25			0			357	33.96	31.04	26.25	91.25	96.69%	1	1	1	3
#3				25			357	25.35	21.60	22.92	69.86	74.03%	1	1	1	3
#4	50			0			357	33.96	31.04	26.25	91.25	96.69%	1	1	1	3
#5				25			357	25.35	21.60	22.92	69.86	74.03%	1	1	1	3
#6				50			357	25.35	21.60	22.92	69.86	74.03%	1	1	1	3
#7	75			0			357	33.96	31.04	26.25	91.25	96.69%	1	1	1	3
#8				25			357	25.35	21.60	22.92	69.86	74.03%	1	1	1	3
#9				50			357	25.35	21.60	22.92	69.86	74.03%	1	1	1	3
#10				75			357	25.35	22.78	22.92	71.04	75.28%	1	1	1	3
#11	100			0			357	33.96	31.04	26.25	91.25	96.69%	1	1	1	3
#12				25			357	25.35	21.60	22.92	69.86	74.03%	1	1	1	3
#13				50			357	25.35	21.60	22.92	69.86	74.03%	1	1	1	3
#14				75			357	25.35	22.78	22.92	71.04	75.28%	1	1	1	3
#15				100			357	25.35	21.60	22.92	69.86	74.03%	1	1	1	3
#16	125			0			357	33.96	31.04	26.25	91.25	96.69%	1	1	1	3
#17				25			357	25.35	21.60	22.92	69.86	74.03%	1	1	1	3
#18				50			357	25.35	21.60	22.92	69.86	74.03%	1	1	1	3
#19				75			357	25.35	22.78	22.92	71.04	75.28%	1	1	1	3
#20				100			357	25.35	22.78	22.92	71.04	75.28%	1	1	1	3
#21				125			357	25.35	22.78	22.92	71.04	75.28%	1	1	1	3
#22	150			0			357	33.96	31.04	26.25	91.25	96.69%	1	1	1	3
#23				25			357	25.35	21.60	22.92	69.86	74.03%	1	1	1	3
#24				50			357	25.35	21.60	22.92	69.86	74.03%	1	1	1	3
#25				75			357	25.35	22.78	22.92	71.04	75.28%	1	1	1	3
#26				100			357	25.35	21.60	22.92	69.86	74.03%	1	1	1	3
#27				125			357	25.35	22.78	22.92	71.04	75.28%	1	1	1	3
#28				150			357	25.35	21.60	22.92	69.86	74.03%	1	1	1	3
#29	175			0			357	33.96	31.04	26.25	91.25	96.69%	1	1	1	3
#30				50			357	25.35	21.60	22.92	69.86	74.03%	1	1	1	3
#31				75			357	25.35	22.78	22.92	71.04	75.28%	1	1	1	3
#32				100			357	25.35	22.78	22.92	71.04	75.28%	1	1	1	3
#33				125			357	25.35	22.78	22.92	71.04	75.28%	1	1	1	3
#34				150			357	25.35	21.60	22.92	69.86	74.03%	1	1	1	3
#35				175			357	25.35	21.60	22.92	69.86	74.03%	1	1	1	3
#36	200			0			357	32.57	28.96	25.21	86.74	91.91%	1	1	1	3
#37				**25**			**357**	**24.58**	**20.14**	**22.50**	**67.22**	**71.23%**	**1**	**1**	**1**	**3**
#38				**50**			**357**	**24.58**	**20.14**	**22.50**	**67.22**	**71.23%**	**1**	**1**	**1**	**3**
#39				**75**			**357**	**24.58**	**20.14**	**22.50**	**67.22**	**71.23%**	**1**	**1**	**1**	**3**
#40				**100**			**357**	**24.58**	**20.14**	**22.50**	**67.22**	**71.23%**	**1**	**1**	**1**	**3**
#41				**125**			**357**	**24.58**	**20.14**	**22.50**	**67.22**	**71.23%**	**1**	**1**	**1**	**3**
#42				**150**			**357**	**24.58**	**20.14**	**22.50**	**67.22**	**71.23%**	**1**	**1**	**1**	**3**
#43				**175**			**357**	**24.58**	**20.14**	**22.50**	**67.22**	**71.23%**	**1**	**1**	**1**	**3**
#44				**180**			**357**	**24.58**	**20.14**	**22.50**	**67.22**	**71.23%**	**1**	**1**	**1**	**3**
#45				**185**			**357**	**24.58**	**20.14**	**22.50**	**67.22**	**71.23%**	**1**	**1**	**1**	**3**
#46				**189**			**357**	**24.58**	**20.14**	**22.50**	**67.22**	**71.23%**	**1**	**1**	**1**	**3**
#47				**190**			**357**	**24.58**	**20.07**	**22.50**	**67.15**	**71.16%**	**1**	**2**	**1**	**4**
#48				**195**			**357**	**24.58**	**20.00**	**22.36**	**66.94**	**70.93%**	**1**	**3**	**3**	**7**
#49				**200**			**357**	**24.44**	**20.00**	**22.22**	**66.67**	**70.64%**	**3**	**3**	**5**	**11**
#50				**195**	**189**	**195**	**357**	**24.58**	**20.14**	**22.50**	**67.22**	**71.23%**	**1**	**1**	**1**	**3**
#51						**190**	**357**	**24.58**	**20.14**	**22.50**	**67.22**	**71.23%**	**1**	**1**	**1**	**3**
#52						**191**	**357**	**24.58**	**20.14**	**22.50**	**67.22**	**71.23%**	**1**	**1**	**1**	**3**
#53						**192**	**357**	**24.58**	**20.14**	**22.50**	**67.22**	**71.23%**	**1**	**1**	**1**	**3**
#54						**193**	**357**	**24.58**	**20.14**	**22.50**	**67.22**	**71.23%**	**1**	**1**	**1**	**3**
#55						**194**	**357**	**24.58**	**20.14**	**22.36**	**67.08**	**71.08%**	**1**	**1**	**3**	**5**

L'objectif de cette étude n'est pas de trouver la meilleure solution en combinant makespan, consommation et basculements, mais de montrer la contrôlabilité des critères de performances en fonction des paramètres afin de montrer l'intégration de la consommation énergé-

tique dans cette approche de pilotage. Ainsi, identifier la « meilleure « solution dans ce problème multicritère n'a aucun sens puisque cela dépendant de la méthode d'agrégation des critères et des préférences de l'utilisateur final. Cependant, une analyse de Pareto peut être menée afin de souligner la contrôlabilité. Cette analyse permet de fournir les choix non dominés aux responsables de production. Ces choix non dominés sont représentés en gras et le numéro de ces scénarios sont encadrés d'étoiles dans la table A.IV.

La simulation #1 est un scénario dans lequel les ressources ne basculent pas en veille. Ainsi la consommation est maximale (94.38 W.h), le C_{max} est minimal (333s) et le nombre de basculement est minimal (3) signifiant que les ressources se réveillent au début de la production et ne passent jamais en veille. La simulation #49 représente la simulation avant la consommation minimal (66.67W.h = 30% d'économie d'énergie), mais implique un nombre maximal de basculements (11). De plus due à la priorité donnée aux ressources déjà éveillés, le C_{max} est légèrement plus important que pour la simulation #1 (357 au lieu de 333s). Les simulations #37 à #46 et #50 à #54 sont équivalentes, avec les mêmes résultats pour les trois indicateurs. Ces simulations fournissent un nombre de basculements minimum (3), une consommation relativement faible (67.22W.h) et un C_{max} de 357 secondes. Finalement, les simulations #47 à #49 et #55 fournissent des résultats intermédiaires entre la consommation et le nombre de basculements. La figure A.9 présente la frontière de Pareto construite à partir des points non dominés. Le point correspond à la simulation #48 est mis en valeur à titre d'illustration.

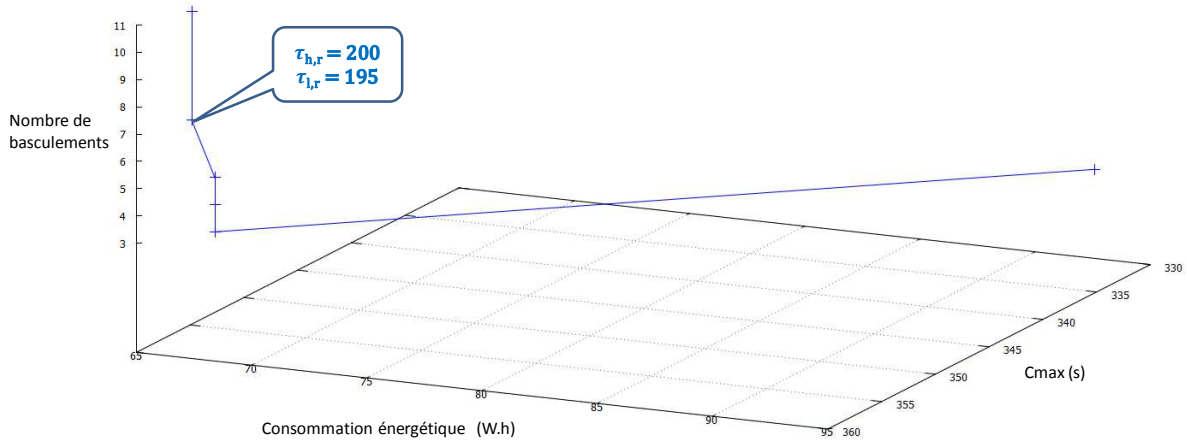


Figure A.9– Frontière de Pareto

Annexe 9.3 : SA2 - simulations à grande échelle

Le but de ces simulations (SA2) est d'étudier le comportement du modèle de champs de potentiel avec des simulations à grande échelle. L'ordre de fabrication utilisé est « PIA ». Trois productions différentes sont réalisées avec 10x l'ordre « PIA » (i.e. 30 produits). Pour le premier scenario, les ordres entre dans le système aussi tôt que possible ; puis pour le deuxième scénario (10x PIA //150 sec) ils entrent dans le système toutes les 150 secondes et toutes les 200 secondes pour le troisième scénario (10x PIA //200 sec). Cela permet d'évaluer la consommation énergétique avec différents taux de charge du système. Finalement, un quatrième scénario est lancé avec 50 ordres « PIA ».

Dans ces simulations, trois combinaisons de seuils $\tau_{h,r}$ et $\tau_{l,r}$ sont utilisés :

- $\tau_{h,R2} = \tau_{h,R3} = \tau_{h,R4} = 0$ et $\tau_{l,R2} = \tau_{l,R3} = \tau_{l,R4} = 0$, ce qui représente l'approche sans gestion de l'énergie, les ressources étant toujours actives.
- $\tau_{h,R2} = \tau_{h,R3} = \tau_{h,R4} = 200$ et $\tau_{l,R2} = \tau_{l,R3} = \tau_{l,R4} = 200$, dans lequel les seuils sont équivalents au « champ de potentiel produit » maximum. Ainsi les ressources sont prêtes lorsqu'un produit arrive sur elles et passent en veille dès qu'elles sont libres.
- $\tau_{h,R2} = \tau_{h,R3} = \tau_{h,R4} = 200$ et $\tau_{l,R2} = 195$, $\tau_{l,R3} = 189$, & $\tau_{l,R4} = 195$, dans lequel les ressources sont prêtes lorsqu'un produit arrive sur elles mais elles ne basculent plus en veille tant qu'un produit est dans leur file d'attente.

Les résultats de ces simulations sont présentés Table A.V. Les temps sont donnés en secondes et les consommations en Wattheure.

Table A.V– Résultats des simulations à grande échelle

Scenario	$\tau_{h,R2}$	$\tau_{h,R3}$	$\tau_{h,R4}$	$\tau_{l,R2}$	$\tau_{l,R3}$	$\tau_{l,R4}$	Cmax (sec)	Cmax (%)	Conso. R2 (W.h)	Conso. R3 (W.h)	Conso. R4 (W.h)	Conso. Globale W.h)	Conso. Globale (%)	Bascul. R2	Bascul. R3	Bascul. R4	Total des Bascul.
10 x PIA	0			0			1369	-	136.7	136.7	136.7	410.2	-	1	1	1	3
10 x PIA	200			200			1398	102.1	115.6	104.4	113.3	333.3	81.3%	15	15	25	55
10 x PIA				195	189	195	1398	102.1	116.5	105.4	114.9	336.9	82.1%	1	1	6	8
10x PIA // 150 sec	0			0			1626	-	158.8	150.4	154.6	463.8	-	1	1	1	3
10x PIA // 150 sec	200			200			1622	99.8	155.6	88.9	89.6	334.0	72.0%	30	10	20	60
10x PIA // 150 sec				195	189	195	1622	99.8	159.5	88.9	89.6	338.0	72.9%	7	10	10	27
10x PIA // 200 sec	0			0			2091	-	220.2	161.9	178.5	560.6	-	1	1	1	3
10x PIA // 200 sec	200			200			2072	99	155.6	88.9	88.9	333.3	59.5%	30	10	20	60
10x PIA // 200 sec				195	189	195	2072	99	158.4	88.9	89.6	336.9	60.1%	10	10	10	30
50x PIA	0			0			6192	-	644.2	638.3	632.5	1915.0	-	1	1	1	3
50x PIA	200			200			6241	100.8	654.7	471.1	542.9	1668.8	87.1%	98	64	128	290
50x PIA				195	189	195	6241	100.8	649.3	486.9	549.5	1685.7	88.0%	2	4	29	35

Les résultats de ces simulations montrent que même pour des simulations à grande échelle, le modèle est toujours viable et contrôlable. Avec un système surchargé, les temps pendants lesquels les ressources ne travaillent pas sont logiquement rares, mais la consommation d'énergie du système est tout de même réduite : environs 19% de gain pour 10 ordres et 13% pour 50 ordres. La différence en termes de Cmax sans gestion de l'énergie (i.e. scénario avec les seuils à 0) est faible : 2% seulement pour 10 ordres et moins d'1% pour 50 ordres. Pour les scénarios dans lesquels l'arrivée des ordres est espacée dans le temps, le gain d'énergie est plus grand à cause d'un taux d'utilisation des ressources plus faible. Ainsi le gain est d'environs 28% pour le scénario avec les ordres espacés de 150 secondes et d'environs 40% pour celui avec 200 secondes d'écart entre chaque ordre.

Toutes ces simulations ont permis de montrer que les seuils $\tau_{h,r}$ et $\tau_{l,r}$ intégrés dans le modèle de champs de potentiel permettent au gestionnaire de production d'ajuster les performances du système en termes d'efficacité (C_{max}) et d'efficience (Consommation d'énergie, nombre de basculements). La contrôlabilité de ces indicateurs de performances a été clairement illustrée dans cette étude. Les simulations à grande échelle ont souligné le fait que le modèle proposé est toujours viable et intéressant avec un grand nombre d'ordres de fabrication.

Annexe 10 : Retour d'expérience sur la mise en œuvre

Lors de la mise en œuvre, autant que possible, nous avons choisi d'utiliser uniquement des équipements déjà commercialisés, des protocoles et standards bien établis. Ainsi, les APIs (Wago) utilisent des langages automates courants (SFC et langage structuré) et l'Eeepc un langage objet usuel (JAVA). Les communications sont réalisées via WiFi avec le protocole Modbus/TCP et la localisation est basée sur l'usage de tags RFID (géolocalisation par RFID). Tous ces éléments sont classiques d'un point de vue industriel.

Cependant, une évolution des usages est nécessaire au niveau industriel. Premièrement, une évolution du mode de pensée « d'un point de vue centralisé à un point de vue distribué ». En effet, dans l'architecture ORCA-FMS, les décisions sont prises par plusieurs entités (Eeepcs et APIs), et le contrôle de bas niveau est réalisé par de nombreux APIs au lieu de 1 ou 2 habituellement. Deuxièmement, les équipements doivent évoluer. Le système nécessite plusieurs petits APIs au lieu d'un seul automate centralisé puissant. Le déploiement et le débogage d'APIs distribués est alors plus complexe. Ainsi, l'utilisation de l'IEC61499 par exemple est conseillée à moyen terme. De plus, le produit intelligent n'est pas encore commercialisé pour le moment, donc il doit encore être « fait maison ». Dans cette thèse, il est composé d'un Eeepc, d'un tag RFID et d'un système de transport classique.

En plus des résultats obtenus par les expérimentations de cette thèse, l'évolution vers le mode de pensée distribué fournit d'autres avantages. Le système est devenu « auto-adaptatif » et donc gère lui-même les perturbations. Cela signifie une réduction drastique des coûts d'ingénierie afférant à la prise en compte de modes de fonctionnement pour faire face à ces perturbations. Finalement, utiliser un langage « évolué » (comme par exemple un langage orienté objet) pour le pilotage du système facilite la mise en œuvre d'entités intelligentes.

Annexe 11 : Intégration de l'énergie dans ORCA-FMS en expérimentation

Cette section présente les scenarios ayant permis de valider la prise en compte de la consommation énergétique du système dans la couche de contrôle local d'ORCA-FMS en expérimentation réelle.

Annexe 11.1 : Paramètres et indicateurs de performances

Ces expérimentations reprennent les paramètres et hypothèses des simulations de l'annexe 8. Ainsi, l'ordre de fabrication choisi est toujours « AIPAIP ». Les résultats sont présentés pour les combinaisons de seuils de basculement ($\tau_{h,r}$ and $\tau_{l,r}$) suivantes :

- $\tau_{h,R2} = \tau_{h,R3} = \tau_{h,R4} = 0$ et $\tau_{l,R2} = \tau_{l,R3} = \tau_{l,R4} = 0$, correspondant à l'approche sans gestion de l'énergie avec des ressources ne basculant pas en veille..
- $\tau_{h,R2} = \tau_{h,R3} = \tau_{h,R4} = 200$ et $\tau_{l,R2} = \tau_{l,R3} = \tau_{l,R4} = 200$, dans lequel les seuils sont équivalents au champs de potentiel produit maximum. Ainsi les ressources sont prêtes lorsqu'un produit arrive sur elles et passent en veille dès qu'elles sont libres.
- $\tau_{h,R2} = \tau_{h,R3} = \tau_{h,R4} = 200$, $\tau_{l,R2} = 195$, $\tau_{l,R3} = 189$ et $\tau_{l,R4} = 195$, dans lequel les ressources sont prêtes lorsqu'un produit arrive sur elles mais elles ne basculent plus en veille tant qu'un produit est dans leur file d'attente.
- $\tau_{h,R2} = \tau_{h,R3} = \tau_{h,R4} = 200$, $\tau_{l,R2} = 195$, $\tau_{l,R3} = 189$ et $\tau_{l,R4} = 190$, scenario quasiment identiquement au précédent, seul le seuil de R4 étant changé. Ce changement permet de n'obtenir qu'un seul basculement sur R4 et donc d'obtenir le même nombre de basculement que pour l'approche sans gestion de l'énergie tout en consommant moins.

Les indicateurs de performances suivis sont les mêmes que pour la simulation à savoir le Cmax, la consommation d'énergie et le nombre de basculements.

Annexe 11.2 : Étude de la Contrôlabilité

La table A.VI présente les résultats obtenus pour l'étude de la contrôlabilité de l'approche par champs de potentiel intégrant l'énergie. Les résultats obtenus en expérimentation sont légèrement différents de ceux de simulation. Il y a en effet des contraintes réelles difficiles à prendre en compte en simulation, comme par exemple la vitesse de certaines navettes (jamais les mêmes) qui n'est pas constante dans les courbes. Ainsi le Cmax est proposé aussi en pourcentage du Cmax du scénario de référence (tous les seuils à 0). Cette vue normalisée

permet de comparer les résultats expérimentaux et de simulation en atténuant les légères différences de modèle. La table A.VI rappelle les résultats de la simulation S1 correspondant pour faciliter la comparaison. Une colonne est ajouté (Colonne #14), afin d'avoir un aperçu des résultats pour le même scénario répété pendant une production de 12 heures (répétition d'environs 110 fois l'ordre de fabrication). Cela permet d'avoir une vue plus « réaliste » de l'énergie économisée car les scénarios de base testés sont assez courts.

Table A.VI– Résultats de l'étude de contrôlabilité

	$\tau_{h,R2}$	$\tau_{h,R3}$	$\tau_{h,R4}$	$\tau_{l,R2}$	$\tau_{l,R3}$	$\tau_{l,R4}$	Cmax (s)	Cmax (%)	Conso. R2 (W.h)	Conso. R3 (W.h)	Conso. R4 (W.h)	Conso. Globale (W.h)	Conso. pour 12 Heures (KW.h)	Conso. Globale (%)	Bascul. R2	Bascul. R3	Bascul. R4	Total des Bascul.
Expérimentation	0			0			397	-	36.74	35.07	35.90	107.71	11.72	-	1	1	1	3
	200			200			421	106%	24.44	20.00	22.22	66.67	7.25	61.90%	3	3	5	11
	200			195	189	195	421	106%	25.28	21.25	23.19	69.72	7.59	64.73%	1	1	3	5
	200			195	189	190	421	106%	25.28	21.25	26.94	73.47	7.99	68.21%	1	1	1	3
	$\tau_{h,R2}$	$\tau_{h,R3}$	$\tau_{h,R4}$	$\tau_{l,R2}$	$\tau_{l,R3}$	$\tau_{l,R4}$	Cmax (s)	Cmax (%)	Conso. R2 (W.h)	Conso. R3 (W.h)	Conso. R4 (W.h)	Conso. Globale (W.h)	Conso. pour 12 Heures (KW.h)	Conso. Globale (%)	Bascul. R2	Bascul. R3	Bascul. R4	Total des Bascul.
Simulation	0			0			333	-	32.29	30.63	31.46	94.38	10.27	-	1	1	1	3
	200			200			357	107%	24.44	20.00	22.22	66.67	7.25	70.64%	3	3	5	11
	200			195	189	195	357	107%	24.58	20.14	22.36	67.08	7.30	71.08%	1	1	3	5
	200			195	189	190	357	107%	24.58	20.14	25.21	69.93	7.61	74.10%	1	1	1	3

En considérant les valeurs normalisées du Cmax et de la consommation d'énergie, les résultats des expérimentations sont cohérents par rapport à ceux obtenus en simulation. Ainsi le scénario de référence en expérimentation est similaire à celui de simulation. Le nombre de basculement est de 3 (chaque ressource se réveille au début de la production), le Cmax est de 397 secondes (64 secondes de plus qu'en simulation) et la consommation est de 107.71 Wattheure (13W.h de plus qu'en simulation à cause du Cmax plus important). Pour le scénario avec les seuils à 200, le Cmax est légèrement supérieur à celui du scénario de référence (24 secondes de plus), la consommation est réduite (62% de la consommation de référence) et le nombre de basculements est maximum (11). Ce comportement correspond à celui de simulation avec un gain d'énergie légèrement plus important, notamment à cause d'un Cmax plus grand. Finalement, les deux autres scénarios donnent des résultats intermédiaires avec une consommation légèrement supérieure à celle du scénario de référence mais moins de basculements.

Cette mise en œuvre réelle montre la pertinence des résultats de simulation obtenus sur ce cas d'étude (cf. chapitre V section 2) et est un premier pas montrant la faisabilité effective de la prise en compte de l'énergie dans ORCA-FMS.

RESUME

Cette thèse contribue au contrôle de la myopie dans les systèmes flexibles de production (SFP). La myopie apparaît lorsque des entités décisionnelles prennent des décisions locales à partir d'une quantité d'information limitée. Cette prise de décision permet de réagir rapidement aux aléas mais induit une performance globale non optimale. Ainsi, ce phénomène doit être contrôlé afin d'obtenir des architectures de pilotage plus performantes.

Après une étude du phénomène de myopie dans d'autres domaines, nous définissons la myopie dans les SFP. Un état de l'art sur les différents types d'architectures permet de retenir les architectures de pilotage hybride mixant hiérarchie et hétérarchie pour contrôler la myopie.

Une typologie des architectures de pilotage hybride est ensuite réalisée avant qu'une nouvelle architecture ne soit proposée : ORCA. Après avoir été présentée, ORCA est déclinée au pilotage des SFP (ORCA-FMS). ORCA-FMS combine deux approches: un modèle linéaire (ILP) et une approche par champs de potentiel.

ORCA-FMS est ensuite appliquée au cas d'étude de la cellule flexible de l'AIP PRIMECA de Valenciennes. Premièrement un modèle de simulation, le plus proche possible du cas d'étude réel est présenté. Il permet d'éprouver l'architecture dans l'environnement de simulation NetLogo. Deuxièmement, afin de valider la pertinence des comportements observés en simulation, l'architecture est mise en œuvre sur la cellule réelle à l'aide du concept de produit actif. Les équipements industriels utilisés pour cette mise en œuvre, le protocole expérimental, ainsi que les résultats obtenus sont détaillés et discutés.

MOTS CLES : Pilotage Hybride, Myopie, Réactivité, Performance Globale, Champs de Potentiel, ILP, Produit Actif, FMS

ORCA: a Hybrid Architecture for the control of myopia in Flexible Manufacturing Systems control.

ABSTRACT

This thesis deals with the control of myopia in Flexible Manufacturing Systems (FMS). Myopia arises when decisional entities take local decisions using limited amount of information. This decision making targets a fast reactivity under perturbations but compromises the overall performance. Thus this phenomenon should be controlled to obtain more efficient control architectures.

After a presenting the related literature in myopia in other domains, myopic behavior in FMS is defined. An analysis of state-of-the-art regarding different types of control architectures determined that hybrid control architectures, mixing hierarchy and heterarchy, are the best option to control myopia. Therefore, a thorough study on hybrid control architectures is presented. Afterwards, a new architecture is proposed: ORCA. ORCA first described and then applied to FMS control (ORCA-FMS). ORCA-FMS combines two approaches: a linear model (ILP) and a potential fields approach.

ORCA-FMS is then applied to the case study of the flexible cell of Valenciennes' AIP PRIMECA. First, a simulation model, as close as possible to the real case study is presented. It allows testing the architecture in the simulation environment NetLogo. Then, to validate the behaviors observed in simulation, the architecture is implemented on the real cell using the active product concept. The industrial equipment used for the implementation, the experimental protocol and the results are detailed and discussed.

KEYWORDS: Hybrid Control, Myopia, Reactivity, Overall Performance, Potential Fields, ILP, Active Product, FMS