Pilotage adaptatif et réactif pour un système de production à flux continu: application à un système de production pétrochimique

Nassima Aissani

Numéro d'ordre UVHC: 10/44

Doctorat

en co-tutelle présentée par

M^me AISSANI Nassima

pour l'obtention du titre de Docteur de

L' Université de Valenciennes et du Hainaut-Cambrésis
Préparée au laboratoire TEMPO, équipe PSI
Discipline : Automatique
Spécialité : Automatique et génie informatique

ET L'Universite D'Oran
Discipline : Informatique
Spécialité : Informatique et Automatique

Intitulé :

Pilotage adaptatif et réactif pour un système de production à flux continu: application à un système de production pétrochimique

Soutenu le : 02 /12/ 2010 devant le jury composé de:

Henri PIERREVAL	**Professeur** à l'IFMA de Clermont-Ferrand, Rapporteur
Abdelkader BENYETTOU	**Professeur** à l'Université Mohamed Boudiaf USTO, Rapporteur
Bernard ESPINASSE	**Professeur** à l'Université d'Aix-Marseille 3, examinateur
Mahmoud BOUFAIDA	**Professeur** à l'Université Mentouri de Constantine, examinateur
Bouziane BELDJILALI	**Professeur** à l'Université d'Oran, Co-directeur de thèse
Damien TRENTESAUX	**Professeur** à l'Université de Valenciennes et du Hainaut-Cambrésis, Co-directeur de thèse

REMERCIEMENTS

Le travail de recherche présenté dans ce mémoire a été effectué au Département d'Informatique au sein du laboratoire LIO à l'université d'Oran en coopération avec l'équipe PSI du laboratoire TEMPO de l'Université de Valenciennes et du Hainaut-Cambrésis. Il n'aurait pas pu aboutir sans le soutien des personnes que je remercie ici.

Je voudrais, tout d'abord, exprimer ma sincère gratitude à Mr BELDJILALI Bouziane, professeur à l'université d'Oran, qui m'a offert la possibilité de rejoindre la Post-Graduation Informatique et Automatique et qui a accepté d'être mon encadrant et qui m'a toujours soutenu dans mon travail. Je le remercie pour ses conseils judicieux et pour l'intérêt qu'il a porté à mon travail.

Un grand merci à Mr Damien TRENTESAUX, professeur à l'université de Valenciennes et du Hainaut Cambrésis. J'aurais difficilement pu souhaiter meilleure relation avec mon encadreur, qui a su m'aider à avancer, m'apportant ses points de vue complémentaires et me guidant dans ma méthode de travail. J'ai encore du chemin à faire, et j'espère pouvoir continuer avec la bonne compagnie de mes encadreurs.

Que Messieurs Henri PIERREVAL, professeur à l'Université de Clermont Ferrand, et BENYETTOU Abdelkader, Professeur à l'Université Mohamed Boudiaf (USTO), trouvent ici toute ma reconnaissance pour avoir accepté de rapporter sur ce mémoire.

Je remercie chaleureusement Messieurs Benard ESPINASSE, professeur à l'Université d'Aix-Marseille et BOUFAIDA Mahmoud, Docteur à l'Université Mentouri de Constantine pour m'avoir fait l'honneur de participer à mon jury de soutenance.

Mes remerciements se dirigent naturellement vers toutes les équipes du LIO, mes collègues et professeurs : Mrs Atmani B, Belalem G, Bouamrane K, et Mmes Hamdadou D et Taghezout N et à tous les anciens et nouveaux membres de la PG Informatique et automatique, notamment ceux qui ont contribué activement à l'élaboration de cette thèse : Djamila B, Amine M et mon amie Ait Si Larbi El Yasmine. Sans oublier les Ingénieurs du Département Informatique et du Département Maintenance et Instrumentation à l'IMSI « Institut de Maintenance et de Sécurité Industrielle ».

Les membres de l'équipe PSI de TEMPO, m'ont toujours bien accueillie et m'ont offert un agréable environnement de travail lorsque j'étais parmi eux. Mes remerciements vont naturellement vers: Melle Chaabane S, Mrs, Sallez Y, Berger T, Valli B, Chaari T et l'adorable Nadine Zbib avec laquelle j'ai partagé son bureau et pour tous les agréables moments qu'on a passé ensemble. J'espère que cette thèse est un bon début pour une collaboration constructive à long terme.

Enfin, des remerciements tout particuliers vont à tout le personnel de l'entreprise NAFTEC/Arzew. Je ne veux omettre personne mais je dois citer : Mr Meglati, Chef département technique et contrôle, Mr Remali, Chef de département production P3 et Mr Chakour, Ingénieur production à l'unité 3100, pour les longues séances de travail qu'on a eu ensemble en essayant de comprendre leurs procédés et formules chimiques !! Sans oublier Mr Bouchikhi, Chef service planification au Département Maintenance, avec qui le système GATIOR semblait plus simple.

A tous mille mercis !

SOMMAIRE

Chapitre 3 : Spécification d'un système de pilotage réactif et adaptatif et modélisation multi-agent

Chapitre 4 : Modélisation par processus décisionnel markovien et résolution par apprentissage par renforcement multi-agents et multi-objectifs

Chapitre 5 : Expérimentation et analyse de l'application sur le pilotage de l'unité 3100

Annexe 2 : La plateforme MADKIT

Annexe 3 : Présentation du système de commande DCS

Annexe 4 : Exemple de déroulement du Q-Learning

Annexe 5 : Détail des données expérimentales

Annexe 6 : Les méthodes de construction de politique dans un MDP

Annexe 7 : Prise de décision

Introduction Générale

Les marchés actuels sont caractérisés par une grande compétitivité. Cette compétitivité a mis les entreprises, notamment celles d'envergure internationale, dans une situation de recherche de compromis entre des objectifs et des contraintes de plus en plus forts et contradictoires :

Des contraintes externes qui concernent les cours du marché, le respect des délais et des coûts face aux clients, ce qui nécessite une augmentation de la flexibilité du système de production pour autoriser la présence de plusieurs gammes opératoires et qui pousse les concepteurs à revoir la structure du système de production et de son système de contrôle.

Des contraintes internes qui concernent l'exploitation des ressources, la durée de vie des outils et la disponibilité des ressources humaines et des matières premières. Ces contraintes impliquent une gestion adaptée aux différentes situations, et le système de contrôle doit être réactif face aux événements imprévisibles.

Cet environnement nous conduit à développer un système de pilotage et de contrôle de production qui n'est pas seulement capable de réagir efficacement mais aussi qui soit en évolution permanente pour améliorer ses performances et la qualité des solutions qu'il propose. Ce système doit pouvoir exploiter au mieux les ressources pour produire le maximum dans les plus brefs délais et avec le moindre coût.

Afin d'exploiter au mieux les ressources de production, ces dernières doivent être en état de fonctionner correctement. Ceci est surtout vrai dans les systèmes de production complexes et à risque telle que l'industrie des hydrocarbures. Pour ce faire, ces ressources subissent souvent des entretiens préventifs ou des corrections suite aux pannes, ce qui les rend indisponibles à ce moment. Les systèmes de pilotage de production doivent aussi prendre en considération ces indisponibilités afin de mieux contrôler et commander le système de production. La maintenance constitue une fonction majeure dans un système de production, même si elle est souvent assimilée à une « bête noire » pour les responsables de production, elle est importante pour le bon fonctionnement du système de production. Production et maintenance doivent être gérées conjointement au sein du système de pilotage.

L'une des principales fonctions de ce système de pilotage dans un contexte de production d'hydrocarbures sera alors l'ordonnancement des tâches de maintenance et de production prenant en compte toutes les contraintes dans un cadre générale (réactivité, délai, coût,…) et les contraintes liées à l'industrie pétrolière qui sont essentiellement des contraintes d'ordre opérationnel (capacité des bacs, ordre des opérations et leur succession). La complexité obtenue est ainsi telle que des techniques spécifiques d'ordonnancement doivent être employées.

Dans ce cadre, l'objectif de cette thèse est de proposer un système de pilotage qui soit réactif et capable d'améliorer en permanence ses performances.

Notre système doit tenir compte simultanément des objectifs de production et de maintenance. Le plan de notre mémoire présentant ce système est le suivant :

Le premier chapitre est consacré à la présentation du contexte industriel et de sa relative originalité dans la littérature scientifique. L'industrie des hydrocarbures est un domaine complexe et à risque, il est nécessaire de décrire ses spécificités et les enjeux qui l'entourent. Des définitions plus génériques sont données sur les principaux concepts utilisés dans ce mémoire tels que le pilotage, l'ordonnancement dynamique et ses différents types et les problématiques qui l'entourent notamment les approches de modélisation et de résolution.

Dans le deuxième chapitre une analyse de l'état de l'art est proposée, en faisant le point sur les différentes approches d'ordonnancement et leur influence sur le pilotage de production. Nous avons aussi analysé les travaux qui se sont intéressés à l'ordonnancement de la maintenance en montrant que peu se sont intéressés à l'ordonnancement conjoint de la production et de la maintenance (surtout curative). Ce chapitre se termine en mettant en évidence notre objectif de développement scientifique.

Le chapitre trois est dédié à la spécification de notre système de pilotage en décrivant principalement son architecture de base. Nous présentons les notions de base d'une approche multi-agents support de notre développement, approche qui rend plus facile la gestion réactive face aux événements du système de pilotage.

Nous spécifions également les niveaux d'interaction que les agents entretiennent entre eux et les moyens de communications qu'ils utiliseront tout en définissant le protocole d'interaction et de communication qu'ils respecteront.

Dans le chapitre quatre nous montrons comment le mécanisme d'apprentissage peut être intégré dans notre système de pilotage multi-agent pour permettre l'amélioration continue des performances.

Le dernier chapitre présente les expérimentations qui ont été réalisées par simulation sur un cas industriel réel, l'unité 3100 de la raffinerie d'Arzew NAFTEC/RA1Z. Nous illustrons dans ce chapitre l'applicabilité de l'approche « Multi-agent et apprentissage par renforcement décentralisé» pour un pilotage réactif et adaptatif de production.

Ce mémoire se termine par une conclusion et sur un ensemble de perspectives pour de futurs travaux.

Chapitre 1 : Contexte industriel et problématique

1 Introduction

Les marchés du gaz et du pétrole actuels doivent, comme la plupart des industries caractérisées par une grande compétitivité, faire face à des contraintes de plus en plus fortes et contradictoires, à la fois externes (qui concernent les cours du marché, le respect des délais et des coûts face aux clients), et internes (qui concernent l'exploitation des ressources, la durée de vie des outils et la disponibilité des ressources humaines et des matières premières).

Ce contexte a poussé depuis plusieurs années les chercheurs à développer des systèmes de pilotage et de contrôle de production qui ne sont pas seulement capables de réagir efficacement mais aussi qui soient en évolution permanente pour améliorer leurs performances et la qualité des solutions qu'ils proposent. Ces systèmes doivent pouvoir exploiter au mieux les ressources pour produire le maximum dans les plus brefs délais et à moindre coût.

Afin de mieux les exploiter, les ressources doivent être en parfait état de fonctionner. Ceci est d'autant plus vrai dans les installations à risque tels que les installations pétrolières, où les ressources subissent encore plus souvent des entretiens préventifs (maintenance préventive). Les systèmes de pilotage de production doivent également prendre en considération ces indisponibilités afin de mieux contrôler et commander le système de production.

Notre activité de recherche se place ainsi relativement dans le cadre du pilotage des systèmes. Notre cadre applicatif concerne une raffinerie, celle d'Arzew en Algérie. L'une des principales fonctions de ce système de pilotage dans ce contexte applicatif sera alors l'ordonnancement des tâches de production prenant en compte toutes les contraintes de production dans un cadre général (maintenance préventive, date de livraisons, coût,…) et bien évidemment, les contraintes liées au process lui-même, qui sont, dans le cadre de l'industrie pétrolière relativement variées et difficiles à appréhender simultanément (capacité des bacs, ordre des opérations et leur successivité, types de produits à gérer et leur spécificité..). En outre, la production des hydrocarbures est une production à flux continu, un seul produit de base (pétrole brut, gaz naturel…) suit plusieurs transformations pour donner plusieurs produits finaux (carburants, huiles moteurs…) et des dérivés (paraffines…). Par conséquent, il nous a semblé important de présenter ce cadre applicatif pour identifier les problématiques et les enjeux sous-jacents. Au préalable, nous présentons rapidement le contexte scientifique dans lequel se place notre étude, celui du pilotage des systèmes de production.

2 Définitions et contexte

Dans cette section nous faisons le point sur les différents concepts qui vont être abordés dans ce mémoire. Notamment, les systèmes automatisés de production, leur systèmes de pilotage et les différentes fonctions inhérentes à ces systèmes. Nous commençons par situer les différents types de systèmes de production.

2.1 Typologie des systèmes de production

Un système de production est un ensemble de ressources réalisant une activité de production. La production est la transformation de ressources (machines et matières) conduisant à la création de biens ou de services (Giard, 1988). Les systèmes de production peuvent être classés par leur mode de production (Ghédira, 2006), nous retrouvons alors trois classes principales :

2.1.1 Production en série unitaire

La production de type « série unitaire » est une production mobilisant sur une période assez longue l'essentiel des ressources de l'entreprise pour réaliser un nombre très limité de projets. Elle concerne particulièrement les grands ouvrages telle que l'industrie aéronautique, la construction des navires… la demande est limitée et leur temps de production est très important.

Les grandes préoccupations de tels systèmes concernent l'optimisation du coût qui peut être parfois immaitrisable. Des ressources externes peuvent être engagées (main d'œuvre qualifiées, équipement spécifiques…). L'ordonnancement préventif dans ce cas est essentiel pour coordonner les tâches afin de respecter les délais et maîtriser les coûts.

2.1.2 Production en ligne

On parle de production en ligne lorsqu'un flux régulier de produits passe d'un poste à l'autre. Les usines d'assemblage d'ordinateurs portables peuvent être un bon exemple. Les équipements sont souvent spécialisés dans la réalisation de tâches spécifiques. L'ordonnancement dans ce type de production concerne l'équilibrage des tâches entre les différents postes pour éviter les longues files d'attente, maîtriser les ressources goulets et raccourcir les délais de fabrication.

2.1.3 Production à flux continu

On parle d'industrie de process ou à flux continu lorsque le mode de production est caractérisé par un flux régulier et important de matières premières destinées à être transformées en matières plus élaborées. On trouve par exemple le secteur agro-alimentaire. Dans notre étude, nous nous intéresserons à ce type de production, car la pétrochimie est le secteur où la production à flux continu est prépondérante. Dans ce type de système, le cheminement du produit est unique. Des contraintes très spécifiques sur les capacités des équipements de transformation et de stockage sont à considérer pour l'ordonnancement de production.

2.2 Le pilotage des systèmes de production

Quel que soit le type de production, les systèmes de production doivent être pilotés afin d'atteindre un certain nombre d'objectifs de production. Dans cette partie, nous présentons ce concept de pilotage.

L'Organisation Internationale de Normalisation « International Standard Organization » (ISO) propose la définition suivante pour le pilotage de la production:

"*Factory control is defined as the actuation of a plant to make products, using the present and past observed state of the plant and demand from the market*" (ISO, 1986).

La définition que nous retenons pour ce mémoire est celle donnée par Trentesaux et al (2000):

« Le pilotage consiste à décider dynamiquement des consignes pertinentes à donner à un système soumis à perturbation pour atteindre un objectif donné décrit en termes de maîtrise de performances. La notion de maîtrise intègre non seulement celle de maintien d'un niveau de performance donné, mais également celle de progrès (évolution vers un niveau de performance souhaité ou avec une amélioration continue) ». Cette définition présente en effet l'avantage de mettre en évidence qu'il ne suffit plus de nos jours pour un système de pilotage de réguler autour d'une consigne, mais également de mettre en œuvre des mécanismes d'amélioration.

Bien que notre étude se focalise sur la raffinerie d'Arzew, celle-ci peut être vue comme un élément d'une chaîne logistique globale qu'il faut piloter. Afin de positionner notre travail dans son cadre, nous nous sommes appuyés sur la formalisation des fonctions proposée par Günther et al. (2005) (figure 1.1). Ce modèle prend en considération toutes les fonctions de l'entreprise: approvisionnement, production, distribution et vente.

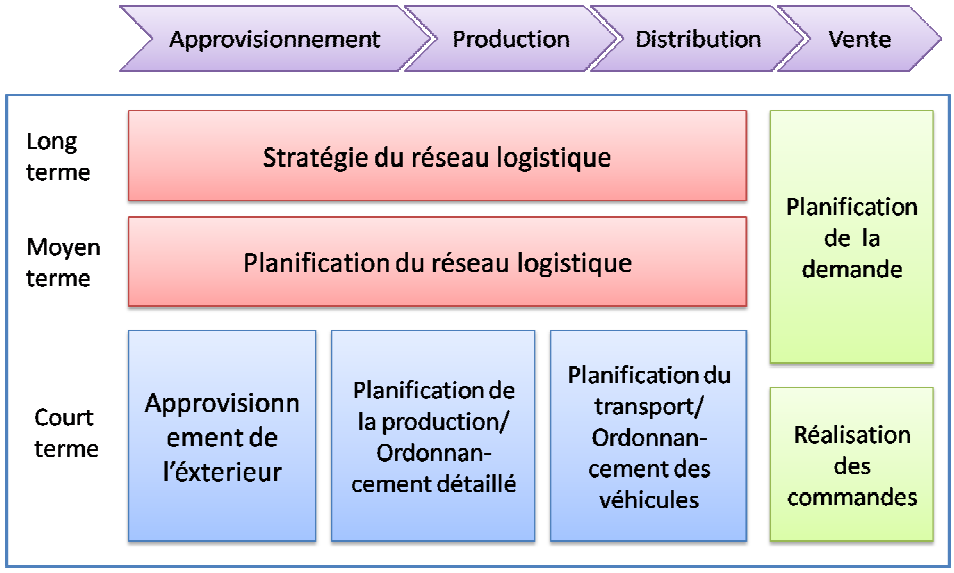


Figure 1.1. Système avancé de la planification (Günther et al., 2005)

Dans notre étude, nous nous intéressons uniquement à un aspect de la logistique interne de la raffinerie, celle relative à la planification de la production et l'ordonnancement, ce qui correspond à la fonction court terme « planification de la production/ordonnancement détaillé » sur la figure 1.1. De manière plus précise, cette fonction peut être affinée comme indiqué dans la figure 1.2, adaptée de (Trentesaux, 2002). Elle met en évidence le lien entre les différentes fonctions de pilotage à court terme.

Dans notre étude, nous nous focaliserons sur les problématiques d'ordonnancement court terme en tenant compte des contraintes de maintenance préventive et de l'état réel du système de production. Ces fonctions sont décrites dans la partie suivante.

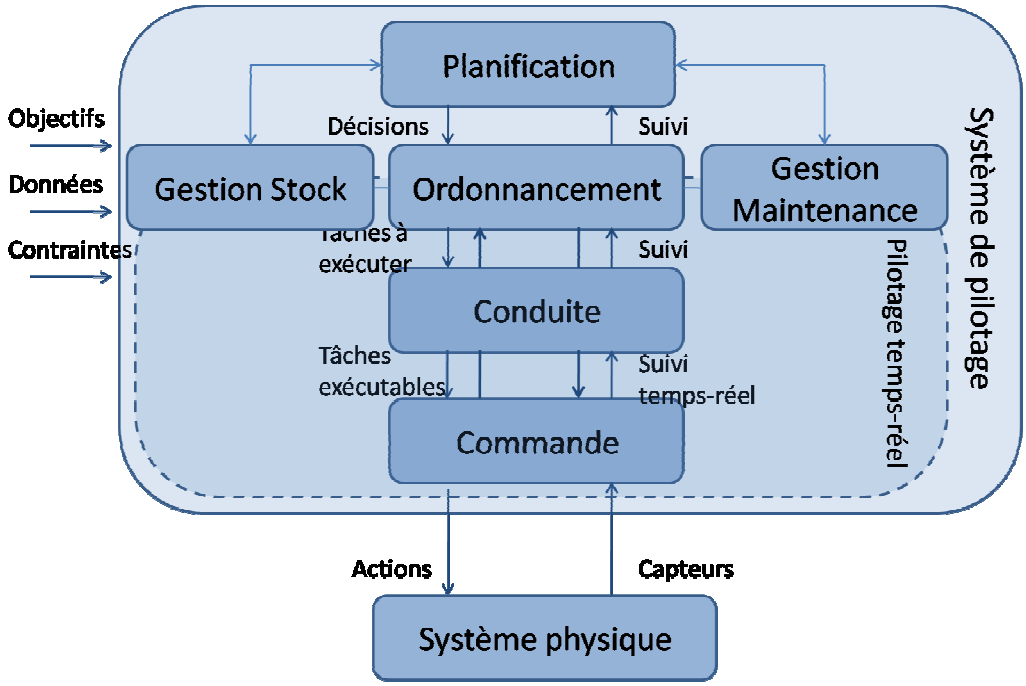


Figure 1.2. Fonctions d'un système de pilotage à court terme

Les principales fonctions d'un système de pilotage sont :

1. **Planification** : Cette fonction correspond au niveau prévisionnel du système de pilotage. Selon des objectifs et des données externes du système des plans de production peuvent être générés en utilisant généralement des systèmes d'informations spécialisés : ERP « Enterprise Resource Planning ». Ces décisions s'inscrivent dans le cadre des décisions stratégiques de l'entreprise.
2. **Ordonnancement** : dans le contexte actuel, l'ordonnancement est l'une des plus importantes fonctions de pilotage de production mais aussi l'une des plus délicates. Selon Pinedo, « *L'ordonnancement est l'allocation des ressources, humaines ou techniques, aux tâches sur une durée définie avec le but d'optimiser un ou plusieurs objectifs* » (Pinedo, 1995).

De notre point de vue, l'ordonnancement est en effet une fonction clé car elle touche les différentes activités de l'entreprise, évidement la production mais aussi la maintenance, l'approvisionnement et la gestion des stocks. En outre, elle présente à la fois une dimension prévisionnelle mais aussi dynamique/temps-réel, ce qui conduit à deux types d'ordonnancement qu'il faut gérer simultanément :

L'ordonnancement prévisionnel : qui se fait en off-ligne, c'est à dire avant même de commencer la production. Il fait suite à la planification.

L'ordonnancement dynamique : Dans un environnement de production réel, la probabilité qu'un ordonnancement prévisionnel soit exécuté exactement comme prévu est faible: des machines en panne, des livraisons de matières premières en retard…Ces perturbations d'exécution entraînent des coûts plus élevés, des dates de livraison repoussées, et plus de temps d'arrêt pour les ressources de production. Pour être plus réalistes un ordonnancement doit tenir compte des incertitudes de l'environnement et s'adapter aux données qu'il perçoit en temps-réel. (Davenport et Beck, 2000)

3. **Conduite** : Cette fonction correspond au niveau décisionnel responsable de la mise en œuvre des tâches ordonnancées pour être lancées par la partie commande. Elle assure la flexibilité quotidienne pour faire face aux fluctuations du système, par exemple, si un plan de production ne peut être appliqué tel quel, une résolution locale peut être entreprise.
4. **Commande** : la fonction de commande décode les tâches à exécuter et lance les commandes pour faire agir le système physique, elle englobe en particulier la commande des équipements et automatismes de production.

2.3 Pilotage et performances

Il y a quelques années, les systèmes de pilotage avaient pour objectif essentiellement des performances décrites en termes d'optimalité et peu en réactivité. Depuis plusieurs années, les dimensions réactive et adaptative (ou globalement, celle relatives à l'agilité) prennent de plus en plus de place. Initialement dans l'industrie automobile (Sauer, 2008 ; Schreiber, 2007), cette évolution se généralise désormais dans tous les types d'industrie.

De nos jours, un système de pilotage doit ainsi pouvoir satisfaire les caractéristiques suivantes :

- *Auto-organisation* : le système de pilotage doit pouvoir trouver la meilleure organisation pour mener le système vers le meilleur de ses performances : c'est un changement décidé de manière autonome pour pallier les changements de l'environnement (Bousbia et Trentesaux, 2002).
- *Adaptation* : Face aux fluctuations de l'environnement industriel et ses changements à moyen et long termes, le système de pilotage doit pouvoir ajuster sa trajectoire pour toujours satisfaire ses objectifs.
- *Optimisation* : Un système de production a pour objectif de réaliser des biens ou des services. L'objectif d'un système de pilotage est de contrôler le système de production pour qu'il réalise ses fonctions en optimisant de plus en plus certains critères afin d'améliorer les performances du système de production, par exemple : minimisation de la durée totale de réalisation des projets Cmax, minimisation des temps d'arrêt des ressources…
- *Réactivité* : Un système réactif est défini comme suit : *Un système réactif agit continuellement et instantanément à des évènements ou stimuli, qu'ils soient externes ou internes* (Zaffalon et Berguet, 1995). Le système de production est aussi soumis à un ensemble d'événements opportuns ou perturbations.

Le système de pilotage doit donc pouvoir réagir dans les meilleurs délais voir en quasi temps-réel pour émettre des décisions qui permettent au système de production de garder son niveau de performances.

Les perturbations auxquelles fait face un système de production sont principalement de deux types :

- *Perturbations Internes* : ce sont des perturbations qui surviennent du système de production lui-même, par exemple : indisponibilité des ressources suite au pannes ou aux arrêts d'entretiens.
- *Perturbations externes* : ce sont des perturbations qui sont externes au système de production mais qui agissent sur ce dernier, par exemple : changement dans la demande des clients, indisponibilité de la matière première…

Dans le cadre de nos travaux relatifs à la raffinerie d'Arzew, nos objectifs de pilotage seront, (de manière assez originale nous le montrerons) de rechercher simultanément non seulement l'optimisation systématique d'un système de production selon des critères usuels dans un cadre d'amélioration continue des performances, mais aussi de réagir aux aléas qui surviennent et de s'adapter par conséquence.

Le contexte scientifique ayant été présenté, la partie suivante décrit le cadre applicatif de nos travaux de recherche et la problématique industrielle qui a justifié notre étude.

3 La raffinerie d'Arzew

La raffinerie NAFTEC/RA1Z d'Arzew est l'une des quatre raffineries de SONATRACH (figure 1.3). La raffinerie est alimentée directement par du pétrole brut de la zone de stockage de Haoud-El-Hamra par un pipeline. La capacité de production de cette raffinerie est de 7,2 millions de tonnes par an. Elle alimente l'ouest du pays avec un surplus exporté vers l'étranger via le terminal d'Arzew (Pour plus de détails consulter annexe 1). Malgré ces chiffres, un premier problème provient du fait que la raffinerie n'est exploitée qu'à 60% de sa capacité (Dossier spécial Algérie, 2002). La raffinerie vise actuellement à augmenter sa cadence de production et améliorer ses performances.

Dans notre projet, nous nous sommes intéressés à une unité prototype l'unité 3100, du département P3 (Production 3). Cette unité est spécialisée dans la fabrication des huiles finies. Ses principaux clients sont : Naftec elle-même, Naftal et le secteur privé.

Figure 1.3. Vue sur la raffinerie d'Arzew

L'unité 3100 est destinée à la fabrication des huiles finies à partir des huiles de base traitées dans les zones HB3 et HB4 (voir annexe 1) et des additifs importés. Cette huile de base est reçue dans des Bacs ou Tank de TK2501 à TK2506, chaque Bac stock un grade (type) d'huile défini : SPO, SAE10-30, BS. Le changement du type d'huile stocké dans un bac nécessite au préalable un rinçage -une opération de longue durée- ce qui est souvent évitée. Cette unité fabrique deux grandes familles d'huiles : huiles moteurs avec un taux de 81% de la production (essence, diesel, huiles pour transmission) et huiles industrielles (hydraulique (tiska), turbines (torba), engrenage (fodda), compresseur (torrada), et huiles diverses). Pour ce faire, deux méthodes sont utilisées : mélange en continu (mélangeuse en ligne) et mélange en discontinu (batch).

Les deux parties suivantes décrivent quelques contraintes externes ou internes que doit gérer la raffinerie.

3.1 Contraintes externes ou « contraintes du marché »

L'industrie des hydrocarbures est d'un niveau international : ses produits sont consommés par le monde entier et en énorme quantité. Cependant, les ressources de même type que la raffinerie d'Arzew sont limitées de par le monde ce qui engendre des contraintes de marché très fortes.

En effet, le marché des hydrocarbures est un marché très vaste où la concurrence est rude et où les prix sont gérés par la loi du marché. La raffinerie doit donc s'adapter aux fluctuations de ce marché en continu. Dans le cadre de sa propre chaîne logistique, la raffinerie a bien évidemment des relations avec des fournisseurs et des sous-traitants, les modalités de ces relations conditionnent l'environnement de production.

Le coût des services des sous-traitants et le coût de la matière première fournie par les fournisseurs influent directement sur le coût de production. Le temps de production est aussi influencé par la date de disponibilité de la matière première et la durée d'exécution des tâches sous-traitées qui conditionnent la durée de réalisation d'un projet de production. Ces conditions doivent alors être prises en considération pour le contrôle et la gestion de production. Le système de contrôle de la raffinerie doit être aussi flexible par rapport à ces contraintes externes qui finalement ne dépendent pas de la raffinerie.

3.2 Contraintes internes

Les contraintes internes sont des contraintes qui dépendent de la raffinerie elle-même. Ces contraintes découlent pour la plupart d'entres-elles du type même de production à flux continu et ses spécificités. Ces contraintes sont bien évidemment extrêmement nombreuses et variées. Dans la suite de ce document, et dans le cadre de nos travaux, nous nous sommes attachés aux contraintes qui vont conditionner l'efficacité du pilotage temps réel. Nous les listons ci-dessous :

Les premières contraintes à considérer sont les *contraintes temporelles*, en particulier, le fait que les durées opératoires des tâches sont très différentes selon les types de tâches. Par exemple, les opérations de transfert de produit se font en quelques heures alors que les opérations de réaction se mesurent en jours. Des contraintes de succession sont aussi présentes dans ce contexte : certaines opérations doivent être exécutées dans un ordre précis, d'autres doivent être strictement successives, c'est-à-dire, sans intervalle de temps entre les opérations. Certaines tâches doivent être cycliques, c'est-à-dire avec une fréquence d'exécution bien précise…

Dans le cadre de l'unité 3100, les durées opératoires des tâches de production dépendent des équipements de transfert et surtout des pompes, car le flux des pompes détermine la durée du transfert qui se fait généralement en quelques heures. Le mélange n'est pas une opération proprement dite, car le mélange se fait en même temps que le transfert, en passant par la mélangeuse en ligne, ou par des opérations de transfert successives sur le manifold. Des contraintes de succession sont aussi présentes dans ce contexte, si le mélange se fait en discontinu, les opérations de transfert doivent être strictement successives, c'est-à-dire, sans intervalle de temps entre les opérations.

Les secondes contraintes sont des *contraintes de capacité* : les produits traités dans ce type de raffinerie sont soit des liquides soit des gaz. Les équipements qui les traitent et les récipients qui les stockent sont d'une capacité limitée, par exemple : dans une opération de transfert il faut savoir si le récipient de destination peut ou non contenir le produit transféré.

Il existe également un ensemble de *contraintes dites opératoires et de sûreté des produits* : les produits considérés sont soit des liquides soit des gaz. Ainsi, en les mettant en contact ils peuvent réagir. De ce fait, les opérations qui se suivent sur un même équipement ou un même récipient doivent respecter ces contraintes chimiques, par exemple : deux grades différents d'une huile ne doivent pas être mélangés dans un même bac.

Enfin, la dernière catégorie de contraintes est relative *à la sûreté des équipements* : la raffinerie est un système de production à risque, soumis à une réglementation sévère qui gère la politique de maintenance de l'entreprise. Cette politique exige un entretien rigoureux et très fréquent des ressources de production, ce qui rend ces ressources indisponibles très régulièrement lors de ces entretiens.

Il existe différentes unités de production au sein de la raffinerie. Notre étude ayant porté sur une de ces unités, l'unité 3100, la partie suivante détaille les spécificités de cette unité.

4 L'unité 3100 de la raffinerie

La présentation de l'unité 3100 est organisée comme suit. Tout d'abord nous présentons les produits réalisés et leur processus de fabrication. Ensuite, nous présenterons les données (flux entrants et sortants) que le système de pilotage doit gérer. Enfin, nous présentons le système de pilotage actuellement implanté et mettons en évidence ses limites.

4.1 Les produits et leur flux

La production dans cette entreprise relève du domaine de la chimie dite « lourde ». L'unité 3100 réalise la fabrication des huiles finies qui sont de deux familles :

- Huiles moteurs avec un taux de 81% de la production : ceux sont les huiles utilisés pour la lubrification des moteurs dans le secteur automobile essence, diesel et huiles pour transmission.
- Huiles industrielles : hydraulique (tiska), turbines (torba), engrenage (fodda), compresseur (torrada).

Ces huiles sont faites à la base des huiles de base avec leurs différents grades : SPO, SAE10, SAE30 et BS, plus des additifs qui sont principalement des minéraux dédiés à chaque type d'huile à fabriquer. Alors une huile est la conséquence de l'équation suivante :

$$X1\%\ Hb1 + X2\%\ Hb2 + X3\%\ Additif1 \ldots\ldots (1)$$

Où : ***Xi*** est un coefficient et ***HBi*** est une huile de base.

Remarque : après avoir étudié les caractéristiques des huiles de base reçues des unités en amont et connaissant les caractéristiques de l'huile finie à fabriquer, les ingénieurs déterminent les types et les quantités des huiles de base à utiliser.

La fabrication d'une huile finie passe par les opérations suivantes (figure 1.4) : La réception des huiles de base, la mise en contact des réactifs dans les conditions opératoires appropriées, le mélange puis le transfert vers les bacs de stockage:

Réception des huiles de base : l'unité reçoit tout d'abord l'huile de base avec ses différents grades des unités en amont (HB3 et HB4), selon un programme décadaire de la réception des huiles de base. Le stockage se fait dans des cuves ou bacs appropriés et l'acheminement à

l'aide de tuyauteries au moyen de pompes. Ces actions sont caractérisées par les contraintes fortes suivantes :

1. Des contraintes sur les capacités des cuves et des bacs qui reçoivent les produits, car ces bacs ont des **capacités limitées** qui différent d'un type de bac à un autre.

2. Des contraintes sur les types de produits contenus dans les bacs, car les bacs sont spécifiques à un produit et le mélange des produits n'est souvent pas possible.

3. Des contraintes sur la capacité de transfert : le transfert se fait généralement par le biais de pompes puisque le produit à traiter est un liquide, ces pompes ont des capacités ou **flux** limités.

Mise en contact des réactifs dans les conditions opératoires appropriées, au sein d'un réacteur (agitateur approprié à la nature et à la consistance du milieu). Ce milieu doit pouvoir assurer un maintien manuel ou automatique de conditions déterminées de température, pression, niveau, concentration, etc. Dans l'unité 3100, des ballons de mélange du D3110 à D3121 sont utilisés pour **mélanger** l'huile de base à des additifs qui sont sous une forme soluble.

Le mélange : La réalisation de l'huile finie en utilisant la formule (1) citée ci-dessus, se fait selon deux manières :

Le mélange en continu : en utilisant une machine appelée mélangeuse, qui joue le rôle d'un mixeur, les différentes huiles utilisées dans la formule sont envoyées simultanément à la mélangeuse pour qu'elle les mixe.

Le mélange par Batch : ou le mélange en discontinu. C'est un ancien mode de fabrication qui est toujours utilisé. Les différentes parcelles d'huiles de base utilisées sont envoyées l'une après l'autre au bac de stockage, et le mélange se fait au niveau de ce bac.

Transfert vers les bacs de stockage : Si c'est un mélange en continu, le produit est alors transféré au fur et à mesure que la mélangeuse est en action. Cependant, si le mélange est en discontinu, alors le transfert se fait parcelle par parcelle. Là aussi on est confronté aux contraintes opérationnelles du type : capacité des bacs, le flux de transfert et le contenu précédent des bacs.

Figure 1.4. Processus de fabrication des huiles finies

La partie suivante détaille les informations qui doivent être nécessaires afin de mettre en place un système de pilotage performant de l'unité 3100.

4.2 Les informations de pilotage et leur flux

4.2.1 Flux entrant

Le système de pilotage de l'unité 3100 doit recevoir un ensemble d'éléments planifiés. Le contrôle de production dans cette unité, doit tenir compte de tous ces plans pour organiser la production avec le meilleur profit, en termes de temps, d'exploitation des ressources, des délais, etc.

Les plans que l'unité reçoit sont:

Le plan décadaire de réception : des unités en amont, l'unité reçoit un plan de réception, où des parcelles d'huile de base sont définies avec leurs types, leurs quantités et la date d'envoi. L'unité 3100 doit se préparer pour les recevoir.

Le plan décadaire de production : Du staff de l'entreprise chargé de la planification, l'unité 3100 reçoit un plan décadaire de production contenant, les différentes huiles finies à fabriquer et leurs quantités. Chaque huile est constituée de différentes quantités des huiles de base et des additifs. Le plan fait apparaitre les tâches d'envoi des huiles de base vers la mélangeuse avec des dates approximatives, ceci afin d'offrir à l'unité une grande marge de manœuvre pour faire face aux contraintes de l'environnement industriel (panne, arrêts d'entretien, demande urgente…). Ces grandes marges introduisent une baisse des performances de l'unité, car la durée d'un projet est en moyenne majorée de 33% alors qu'une ressource n'est occupée qu'à 38% de son temps de marche (Programme décadaire de fabrication huiles finies et graisses U3000). Ceci constitue un premier gisement de progrès potentiel en termes de performances.

Le planning d'entretien annuel : Ce plan est établi par le département maintenance et plus précisément par le service planning. Ce planning fait apparaitre les différentes tâches d'entretien périodiques des ressources et outils de production (équipement concerné, date et durée). L'importance de ces tâches est indiscutable, cependant elles nuisent à la production car elles sont placées indépendamment du plan de production, ce qui constitue un autre gisement de progrès possible.

4.2.2 Flux sortant

Les données sortantes du système de pilotage sont pour l'essentiel :

Un bilan de fabrication : Chaque unité établit un rapport journalier détaillé et un autre mensuel sur l'activité de l'unité. Par rapport à chaque produit fini à fabriqué, une comparaison est faite entre les quantités prévisionnelles à fabriquer et ce qui a été réellement fabriqué pour donner un pourcentage de réalisation. Ces pourcentages permettent d'évaluer les performances de chaque unité et d'avoir une idée précise sur les produits finis disponibles.

Un rapport de la maintenance : Le département maintenance établit aussi un rapport mensuel sur l'activité de la maintenance au niveau de l'entreprise. Ce rapport montre le taux d'intervention sur chaque unité en différenciant maintenance préventive et maintenance curative. Ces rapports permettent de voir si la politique de maintenance préventive est efficace, ceci est défini lorsque le taux de maintenance curative est minime.

Un taux d'occupation et le coût des ressources humaines sont aussi calculés, car ils permettent de faire des prévisions sur les recrutements ou la sous-traitance. D'autres indices de performance sont aussi utilisés et calculés pour évaluer les performances des services de maintenance.

4.3 Le système de pilotage actuel

Le système de pilotage actuellement à la raffinerie est semi-automatique, car les fonctions de planifications et d'ordonnancement sont réalisées par les services planning. Le staff dirigeant de l'entreprise établit la stratégie globale et les plans prévisionnels à long terme, tandis que les plans à court terme sont réalisés par les ingénieurs du site et de chaque unité. La fonction de commande et de suivi est automatisé en utilisant le « distributed control system » (DCS, pour une bref présentation du DCS voir annexe 3). Le DCS permet de faire un suivi et un contrôle des paramètres de production à distance et en temps-réel. Cependant, le DCS n'a aucun effet sur l'organisation ou la planification de la production. La planification et l'ordonnancement constituent le point faible du système de pilotage de raffinerie car ils dépendent exclusivement de l'être humain. Nous pensons alors qu'une automatisation ou semi-automatisation des fonctions de planification et d'ordonnancement et en les liant directement au DCS peut être d'un grand avantage, ce que nous nous proposons de faire dans ce document. Les fonctions de pilotage de la raffinerie sont décrites dans les parties suivantes et sont organisées selon le cadre général présentée figure 1.2.

4.3.1 Ordonnancer la production

L'ordonnancement de la production au niveau de la raffinerie se fait selon un plan prévisionnel établi par le staff planning. Ces plans sont établis en faisant une étude du marché et des commandes prévues en donnant de larges fourchettes de manœuvre pour toutes les tâches de production. Chaque unité a donc un plan de production prévisionnel qu'elle doit honorer, cependant, les responsables de production et les ingénieurs de chaque unité établissent les plans journaliers de production. Lorsque des perturbations proviennent les responsables de production profitent des marges données dans les plans prévisionnels pour récupérer les retards.

4.3.2 Conduite de la production

La fonction conduite est assurée conjointement par l'opérateur humain et le DCS. Le DCS permet de visualiser à distance toute l'installation de production, il fait apparaitre aussi son état en visualisant des paramètres de fonctionnement tel que la pression la température… Le DCS joue aussi le rôle d'un régulateur car, en respectant des consignes pour les paramètres de fonctionnement il agit sur les actionneurs du système de production.

Lorsque les régulations effectuées par le DCS ne parviennent pas à réajuster les paramètres de fonctionnement l'opérateur humain intervient afin de prendre des décisions plus globales.

4.3.3 Gérer les stocks

La gestion des stocks dans la raffinerie est de deux types : la gestion des stocks de production et la gestion des stocks de maintenance.

Certains produits tels que les catalyseurs ou les additifs, sont utilisés dans la transformation de la matière de base, ces produits sont utilisés en petite quantités par rapport à la matière première, alors ils sont stockés en grandes quantités dans des aires de stockage. Tandis que les produits finis, qui sont des carburants, des huiles de base ou des paraffines, sont stockés dans des bacs, leur gestion se fait spontanément par les responsables de production, lorsqu'un produit est en fin de production, les ingénieurs de production vérifient les capacités des bacs de stockage pour y stocker leur produit.

Le deuxième type de gestion des stocks concerne la gestion de stock pour la maintenance. Bien sûr les entretiens et les interventions de maintenance, nécessitent des produits d'entretien et des pièces de rechange. La gestion de ces stocks se fait en utilisant une GMAO (Gestion de maintenance Assistée par Ordinateur) Appelée GATIOR (Manuel utilisateur GATIOR version 1.4).

En ce qui nous concerne, nous nous focaliserons sur la gestion des stocks des produits finis, car c'est une partie intégrante du pilotage de production dans la production à flux continu.

4.3.4 Gérer la maintenance

La gestion de la maintenance au niveau de la raffinerie se fait en utilisant GATIOR. Cette GMAO permet de faire une planification et préparation (en allouant les ressources) des différentes tâches d'entretiens en offrant une bonne visualisation des ressources humaines et physiques.

La gestion de la maintenance se fait selon deux stratégies : la maintenance préventive et la maintenance curative

La maintenance préventive : En fonction du plan de maintenance annuel, le service planning du département maintenance établit les **plans de maintenance mensuels** en déterminant les équipements concernés, les ressources nécessaires : humaines et physiques. Ce plan mensuel organise les tâches d'entretien ou de maintenance préventive sur un horizon de temps qui est d'un mois, ces tâches sont données avec des **marges assez larges** (Date au plus tôt et Date au plus tard), car ces plans sont faits indépendamment de la production. Les responsables de la planification préfèrent donner une grande marge pour pouvoir basculer ces tâches si celles-ci se trouvent en conflits avec les tâches de production.

La maintenance curative : Lorsque des pannes se présentent au niveau des équipements, les tâches de maintenance curatives sont indispensables et ont la plus grande priorité, il suffit alors de leurs affecter les ressources physiques et humaines et les placer immédiatement sur l'horizon temporel.

Enfin, on peut résumer la politique de la raffinerie par: « Entretenir le mieux possible, prévenir le plus possible sans atténuer la production ». Ceci devrait se faire en collaboration avec la production, ce qui n'est pas le cas actuellement et que nous proposons de mettre en œuvre.

Suite à la présentation de la raffinerie et de ses contraintes, nous nous proposons d'expliciter la manière avec laquelle ce système est actuellement piloté afin de mettre en évidence les limites du système de pilotage industriel et les sources d'améliorations possibles. Le prochain chapitre analysera l'état de l'art dans ce cadre et montrera également les limites des modèles de systèmes de pilotage en tant qu'élément de réponse concernant notre problème.

5 Les Objectifs de la raffinerie et limites du système de pilotage actuel

Le système de pilotage semi-automatique actuel présente un certain nombre de limites qui génèrent un certain nombre de sous-performances. Ceci est d'autant plus préjudiciable que dans le contexte actuel que nous avons présenté auparavant, les responsables de la raffinerie ont pour ambition de revoir à la hausse un certain nombre de performances cibles.

5.1 Limites du système de pilotage actuel

Les performances actuelles du système de pilotage de l'entreprise ne sont pas jugées très bonnes et il est facile d'identifier un certain nombre de gisement de progrès de performances.

En effet, les plans de production prévisionnels décadaires établis par l'entreprise montrent que les délais de production ou de la réalisation d'un projet sont très **élargis** pour pouvoir réagir an cas de perturbation**,** par exemple sur un plan de production décadaire 50% du temps global de production n'est pas exploité.

A cet effet, les ressources de production connaissent aussi beaucoup de **temps mort**. Une ressource n'est exploitée en moyenne qu'à 48% de sont temps de disponibilité.

Ces mesures prises par les dirigeants de l'entreprise, montrent que le système de pilotage **n'est pas réactif.** Car ces élargissements de délai et ces temps morts permettent aux responsables de production de décaler autant que possible les tâches de production si des événements inopinés se produisent. Exemple, sur la figure 1.5 la tache p à une durée d'exécution de 1 unité mais sur le plan elle est programmée sur 3 unités, lorsque une perturbation se produit dans la première unité, la tâche p est décalée à l'unité 2.

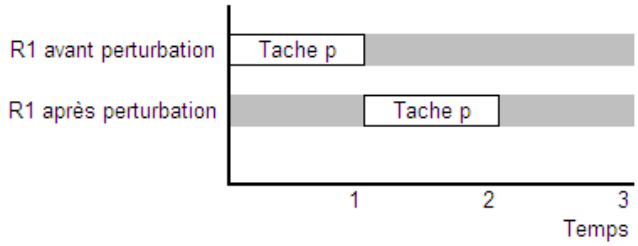


Figure 1.5. Décalages de tâches dans les marges

5.2 Nouveaux objectifs en performance

Au delà des limites de performances imposées par le système de pilotage lui-même, les responsables de la raffinerie ont souhaité améliorer certaines performances. Nous listons ici quelques améliorations recherchées en identifiant celles que nous traiterons par la suite.

5.2.1 Raccourcir et mieux appréhender les délais de production

La raffinerie reçoit des commandes de nombreux clients, afin de les planifier et y répondre dans les délais souhaités, la raffinerie a fait une étude approfondie du marché à moyen terme afin de faire une planification prévisionnelle des commandes sur une année. Cette étude prévisionnelle permet à l'entreprise de s'octroyer de grandes marges temporelles dans la réalisation de ces commandes. Les délais de production d'une commande donnée sont ainsi très larges et très longs par rapport à ce qu'ils peuvent être réellement et par rapport à la concurrence. La réduction de ces délais (recherche d'une meilleure **efficacité**) couplé à l'accroissement de la capacité à suivre les évolutions du marché (recherche d'une meilleure **adaptabilité**) constitue un premier objectif.

Egalement, la raffinerie présente des difficultés à répondre aux commandes urgentes non planifiées au préalable à court terme. Dans l'approche actuelle, les plans de production sont en effet réalisés annuellement puis par décadaire, il est dès lors difficile d'insérer une commande urgente, d'autant plus que les tâches planifiées présentent de longues marges d'erreur. Ceci constitue un second objectif (recherche d'une meilleure **réactivité**).

Nos travaux cherchent à répondre à ces deux objectifs.

5.2.2 Augmenter la capacité de production

La raffinerie, selon ses statistiques, ne tourne qu'à 60% de son régime. Des plans décadaire de production dans l'unité 3100 ont montré que les grandes marges d'erreur données aux tâches introduisaient une baisse des performances de l'unité, car la durée d'un projet est en moyenne majorée de 33% et une ressource n'est occupée qu'à 38% de son temps de marche (Programme décadaire de fabrication huiles finies et graisses U3100).

Afin que l'entreprise soit encore plus compétitive, elle vise à augmenter son régime en adoptant une méthode de planification qui permet de contrôler et de gérer le système opérant de l'entreprise de la manière la plus efficace possible et d'arriver aux plafonds fixés par la réglementation 80% - 85% du régime de l'entreprise (recherche d'une meilleure **efficience**). Nos travaux nous ont poussé à prendre en compte cet aspect et nous montrons que notre approche conduit à une meilleure efficience.

5.2.3 Mieux entretenir

La raffinerie est soumise à une réglementation très stricte d'un niveau nationale et internationale. Cette réglementation exige une bonne gestion des risques en introduisant une politique d'entretien très rigoureuse. L'entretien est une forme de la maintenance préventive, même si cette maintenance est nécessaire ces tâches d'entretien constituent des temps morts pour la production. Les visites de vérification et d'entretien sont très fréquentes.

Ces tâches surviennent à tout moment et peuvent entraver le fonctionnement du système de production. Cependant, on ne peut ni annuler les tâches d'entretien ni arrêter la production, de ce fait, des recherches doivent être menées pour aligner la planification de la maintenance à la planification de la production, c'est-à-dire établir des plans de production en tenant compte de la maintenance et vice-versa (recherche d'une **meilleure intégration des opérations de maintenance dans le planning de production**). Nous tiendrons compte de ces contraintes dans notre proposition.

5.3 Résumé des objectifs de performances

En conclusion, la raffinerie souffre actuellement du manque d'optimisation en planification et en gestion de ses ressources face à la complexité de son processus de fabrication et de ses contraintes. L'entreprise vise alors à améliorer ses performances en raccourcissant les délais de production et en utilisant intelligemment ses ressources de production, ceci afin d'augmenter leur cadence pour produire plus dans de plus court délais. L'entreprise vise aussi à faire plus d'entretien pour être conforme à la réglementation, cependant elle doit être sans incidence sur la production. Lorsque des évènements inopinés se produisent, telles que les pannes ou l'arrivée non prévue de commandes clients, l'entreprise doit pouvoir y répondre et être réactive à tout évènements incertains qui pourrait se produire. Enfin, l'entreprise vise à acquérir un système de contrôle global qui lui permet d'être réactive aux évènements incertains.

Ces constatations ont poussé les responsables de la raffinerie à vouloir disposer d'un système de pilotage qui leur permet d'améliorer conjointement leur capacité à s'adapter à moyen et court terme, ce qui est, nous l'avons montré, très difficile à réaliser avec l'approche et le système de pilotage actuel, mais ce qui, comme nous le montrerons par la suite, est plus facile à réaliser selon notre approche.

6 Conclusion

Dans ce chapitre, nous avons tout d'abord explicité la problématique du pilotage dans les systèmes de production à flux.

Le pilotage des systèmes de production à flux continu fait face à de nombreuses contraintes : d'abord internes qui sont des contraintes liés à la nature des processus de fabrication et leurs complexités, liés aussi aux événements incertains qui peuvent se produire telles que les pannes… Puis des contraintes externes qui découlent de l'environnement du système de production, des marchés dans lesquels évolue ce système, de la nature de ses produits et la catégorie de ses clients. Ces contraintes exigent simultanément une optimalité et une réactivité pour le pilotage.

Nous avons dans ce chapitre présenté notre cas d'étude qui est la raffinerie d'Arzew en Algérie et nous nous sommes focalisés surtout sur l'unité 3100 qui fabrique des huiles finies. Nous avons montré que la performance actuelle de l'unité 3100 est relativement faible en termes d'optimalité et de réactivité, ceci est à la fois dû à l'approche de planification adoptée (sur le long terme, avec d'importantes marges) mais aussi à l'outil de pilotage utilisé.

Un gisement de performance important est ainsi possible, d'autant plus que le contexte actuel pousse les responsables de la raffinerie à accroître leurs exigences face à la concurrence internationale forte.

Pour ce faire, un système de pilotage capable simultanément d'optimiser la production tout en s'adaptant aux fluctuations de l'environnement pourrait contribuer à l'atteinte des objectifs recherchés par les responsables.

Dans le prochain chapitre, nous analysons l'état de l'art scientifique en réponse à cette problématique. Nous identifions les approches possibles applicables dans notre contexte et analysons les résultats obtenus. Nous montrerons les limites des contributions actuelles, ce qui justifiera notre proposition pour un pilotage réactif et adaptatif des systèmes de production à flux continu.

Chapitre 2 : Etat de l'art : pilotage et ordonnancement dynamique

1 Introduction

Dans le chapitre précédent, nous avons étudié le contexte de la production à flux continu. Nous avons montré que dans le contexte actuel, il était nécessaire de définir un système de pilotage capable simultanément d'optimiser la production tout en s'adaptant aux fluctuations de l'environnement. Dans ce chapitre, nous nous intéressons aux approches de modélisation et de résolution pour les systèmes de pilotage réactif et adaptatif. Dans un sens fonctionnel du système de pilotage, l'ordonnancement (qu'il soit dynamique, réactif, etc.) est l'une des plus importantes fonctions et problématiques de pilotage de production, car elle touche toutes les autres fonctions de pilotage : planification, conduite, commande, gestion des stocks et gestion de la maintenance. C'est pourquoi nous avons choisi de nous y intéresser dans le cadre de cette thèse.

Nous commencerons par présenter les notions élémentaires pour la modélisation d'une fonction en d'ordonnancement.

Nous présenterons ensuite une classification des approches d'ordonnancement et leur influence sur les performances des systèmes de pilotage telles que l'optimisation, réactivité et adaptabilité. Les approches d'ordonnancement sont classées selon la considération ou non des événements incertains de l'environnement de production. Plusieurs classifications existent dans la littérature, citons Suresh et Chaudhuri (1993) et Davenport et al. (2000). Pour notre part, nous avons présenté une classification qui identifie les différents types d'incertitudes et de données incertaines et différentes approches d'ordonnancement qui en tiennent compte de différentes manières (Chaari, 2010). Nous nous intéressons plus spécialement à l'approche réactive et ses dérivées qui permettent de donner des solutions d'ordonnancement efficaces tout en restant réactif aux événements perturbateurs de l'environnement avec la possibilité de s'y adapter.

De notre analyse de l'état de l'art, seront déterminées les meilleures pratiques pour la modélisation d'un système de pilotage basé sur un ordonnancement réactif et adaptatif.

Le système de pilotage que nous prévoyons de développer, devrait englober les contraintes issues de la production et la maintenance (prédictive et curative). Afin de mettre le point sur les meilleurs pratiques dans le domaine, une analyse des travaux réalisés pour l'ordonnancement réactif et adaptatif conjoint de la production et de la maintenance sera également développée.

Enfin, une synthèse des idées de base retenues pour la conception de notre système de pilotage sera faite afin d'élaborer notre cahier des charges qui sera décrit dans le chapitre suivant.

2 L'ordonnancement de production et de maintenance dans le système de pilotage

L'ordonnancement se trouve au cœur d'un système de pilotage de production et en liaison directe avec les différentes autres fonctions de ce système.

En effet, l'ordonnancement coordonne et synchronise les activités de production et de maintenance afin d'assurer la disponibilité des outils de production et l'optimalité de la production.

La problématique d'ordonnancement a depuis longtemps attiré l'attention des chercheurs. Dans un système de production complexe tel que les systèmes pétroliers à flux continu ce problème est encore plus difficile compte tenu de la complexité du procédé et les contraintes qui l'entourent. Contrôler ces systèmes c'est aussi prendre en considération son environnement et s'y adapter.

Dans cette première section, nous présenterons des définitions de l'ordonnancement puis un état de l'art sur les différentes approches et leurs influences sur le pilotage de production et ses différentes fonctions.

2.1 Définition de l'ordonnancement

L'ordonnancement en général est un champ d'investigation qui a connu un essor important ces quarante dernières années (Blazewicz, et al., 1996; Brucker, 1998; Lopez et Roubellat, 2001), tant par les nombreux problèmes identifiés que par l'utilisation et le développement de nombreuses techniques de résolution. Nous donnons deux définitions différentes qui vont nous permettre d'identifier les notions importantes sous-jacentes.

Définition 1 de l'ordonnancement : « *L'ordonnancement est l'allocation des ressources, humaines ou techniques, aux tâches sur une durée définie avec le but d'optimiser un ou plusieurs objectifs* » (Pinedo, 1995).

Dans les problèmes d'ordonnancement des systèmes de production quatre notions fondamentales sont utilisées: les **tâches**, les machines ou **ressources** et certains **objectifs** à optimiser en respectant des **contraintes**.

Définition 2 « *Le problème d'ordonnancement consiste à organiser dans le temps la réalisation d'un ensemble de tâches, compte tenu de contraintes temporelles (délais, contraintes d'enchaînement, etc.) et de contraintes portant sur l'utilisation et la disponibilité des ressources requises par les tâches* » (Lopez et Roubellat, 2001).

Dans cette définition nous retrouvons la notion de **séquence** dans « l'organisation dans le temps un ensemble de tâches » ainsi que la notion de **contraintes**. Nous détaillons ces différentes notions dans la suite de cette partie et introduisons également le formalisme dont nous nous servirons dans la suite du document.

.

2.2 Les tâches

« *Une tâche est le processus le plus élémentaire. Elle est constituée d'un ensemble d'actions à accomplir, dans des conditions fixées, pour obtenir un résultat attendu et identifié, en termes de performances, de coûts et de délais* » (norme AFNOR NF X50-310).

Une tâche O_{ij} est localisée dans le temps par:

- p_{ij} : La durée opératoire de la tâche i sur la machine j;
- r_{ij} : La date de début au plus tôt ou date de disponibilité de la tâche i sur la machine j ;
- f_{ij} : La date de fin au plus tard ou «deadline» de la tâche i sur la machine j;
- t*ij* : La date de début d'exécution de la tâche i sur la machine j ;
- c*ij* : La date de fin d'exécution de la tâche i sur la machine j ;

Dans ce mémoire, un Job est constitué de plusieurs tâches.

2.3 Les ressources

« Une ressource R_k est un moyen technique ou humain requis pour la réalisation d'une tâche ». Elle est disponible en quantité limitée, et sa capacité est supposée constante. Plusieurs types de ressources sont à distinguer. Une ressource est **renouvelable** si après avoir été utilisée par une ou plusieurs tâches, elle est à nouveau disponible en même quantité (les hommes, les machines, l'espace, l'équipement en général, etc.). La quantité de ressources utilisable à chaque instant est limitée. Dans le cas contraire, elle est **consommable** (matières premières, budget, etc.). La consommation globale (ou cumul) au cours du temps est limitée. Par ailleurs, on distingue les ressources **disjonctives** (ou non partageables) qui ne sont allouées qu'à une tâche à la fois (machine-outil, robot manipulateur) et les ressources **cumulatives** (ou partageables) qui peuvent être utilisées par plusieurs tâches simultanément (équipe d'ouvriers, poste de travail). La **séquence** sur une machine, détermine sur un horizon temporel la suite des tâches à exécuter sur cette ressource.

Ces notions sont d'une grande importance pour que l'on puisse par la suite déterminer le type du problème d'ordonnancement auquel notre système de pilotage devra faire face.

2.4 Les objectifs

Lors de la résolution d'un problème d'ordonnancement, on peut distinguer deux types de stratégies, visant respectivement à l'*optimalité* des solutions par rapport à un ou plusieurs critères, ou à leur *admissibilité* vis-à-vis des contraintes. L'approche par optimisation suppose que les solutions candidates à un problème puissent être ordonnées de manière rationnelle selon un ou plusieurs critères d'évaluation numérique permettant d'apprécier la qualité des solutions. On cherchera à minimiser ou maximiser de tels critères. Le critère le plus utilisé et le plus connu est le ***Cmax***

Fin de traitement $Cmax=max_{i=1,n}(C_i)$

C_i est la date de fin d'exécution de toutes les tâches du job *i*, *n* nombre de jobs.

2.5 Les contraintes

Selon le domaine d'application, la fonction ordonnancement peut avoir d'autres objectifs que celui de veiller au simple respect des contraintes de temps et de ressource. Il peut s'agir de satisfaire des objectifs économiques, de respecter une législation du travail en vigueur dans une entreprise ou, comme dans notre cas, de gérer au mieux des contraintes sur les capacités des ressources et les différents types de tâches à exécuter.

Une contrainte exprime des restrictions sur les valeurs que peuvent prendre conjointement les variables représentant les relations reliant les tâches et les ressources. On distingue les contraintes temporelles et les contraintes de ressources.

Les contraintes temporelles intègrent:

- les contraintes de temps alloué, issues généralement d'impératifs de gestion et relatives aux dates limites des tâches (délais de livraison, disponibilité des approvisionnements) ou à la durée totale d'un projet ;
- les contraintes d'antériorité et plus généralement les contraintes de cohérence technologique, qui décrivent le positionnement relatif de certaines tâches par rapport à d'autres ;

Les contraintes de ressources traduisent le fait que les ressources sont disponibles en quantité limitée. On distingue deux types de contraintes de ressources, liées à la nature disjonctive ou cumulative des ressources. Une ressource disjonctive ne peut être utilisée que par une tâche à la fois. On trouve aussi des contraintes sur les capacités des ressources qui subissent des opérations de vidange et de remplissage, exemple : bacs de stockage huile.

Dans ce qui suit, nous donnerons une classification des approches d'ordonnancement, tenant en compte ou non des incertitudes de l'environnement de production, et leur influence sur le pilotage de production.

3 Etat de l'art : approches d'ordonnancement et leur influence sur le pilotage

De notre point de vue, la fonction ordonnancement est la fonction principale dans le système de pilotage de production car l'ordonnancement a une influence directe sur les performances du système de pilotage. En outre, Pujo et Kieffer (2002) ont montré que le type d'approche d'ordonnancement influence lui aussi sur la qualité du pilotage. Pour les présenter, dans ce document, les approches de pilotage ont été classées en fonction de l'approche d'ordonnancement utilisée : soit un pilotage qui utilise un ordonnancement prévisionnel et le réajuste, ou un pilotage qui utilise un ordonnancement dynamique qui lui permet d'être réactif, ou en dernier, sans ordonnancement pour les systèmes de production fortement perturbés ce qui donne un pilotage temps-réel.

3.1 Ordonnancement préventif et pilotage à moyen et long terme

Lorsqu'il s'agit d'un pilotage à moyen ou à long terme, on a plutôt besoin d'établir un plan d'actions basé sur des paramètres (temps de process, taux d'arrivée, taux de panne etc.) estimés de manière déterministe ou probabiliste. Le plan d'action permet de prendre des décisions pour assurer le fonctionnement courant. Les paramètres de pilotage sont déterminés avant exécution sur le système réel en établissant des plans de la façon la plus optimale en minimisant un certain coût (durée du projet, coût de main d'œuvre ou temps d'occupation des ressources…). Les approches utilisées sont des approches d'optimisation même si ces dernières nécessitent des temps calculatoires importants.

Les systèmes de pilotage à moyen et long terme s'appuient sur l'ordonnancement prévisionnel, leur modélisation nécessite une certaine certitude et exactitude en utilisant des méthodes optimales énumératives soit exactes soit approchées.

3.1.1 Méthodes exactes

Parmi ces méthodes nous pouvons distinguer les procédures par séparation et évaluation, la programmation linéaire et la programmation dynamique.

Plusieurs travaux ont été réalisés en utilisant les procédures par séparation et évaluation (Lawler et Wood, 1966) des procédures comme *Branch and bound*, énumèrent l'ensemble des solutions possibles pour un problème d'ordonnancement donné et déterminent quelle est la solution la plus optimale selon un certain critère. Cependant, l'analyse des propriétés du problème permet d'éviter l'énumération des classes considérées comme mauvaises solutions. Un bon algorithme par séparation et évaluation, énumère donc seulement les solutions potentiellement intéressantes. Briand et La (2003) ont propose une procédure par séparation et évaluation progressive (PSEP) pour l'ordonnancement robuste de problèmes à une machine en s'appuyant sur le théorème de dominance (Erschler et al. 1983).

La programmation linéaire permet de modéliser les problèmes d'optimisation dans lesquels critères et contraintes sont des fonctions linéaires des variables. Les deux types d'algorithmes les plus importants pour traiter un programme linéaire à variables continues sont la méthode du Simplex et la méthode des points intérieurs. L'inconvénient majeur réside dans le nombre important de variables et de contraintes nécessaires. Les modèles non-linéaires peuvent aussi être utilisés, par exemple, dans une unité de distillation sous vide dans une raffinerie (pour une présentation de l'unité voir annexe 1, Aissani el al. 2009a), Joly et al. (2002) ont développé un modèle mathématique non-linéaire pour l'optimisation du stockage des différents types de pétrole brut reçu et son transfert vers l'unité de distillation pour maximiser le taux de production en respectant les contraintes opérationnelles de l'unité. Face à un problème de distribution des produits raffinés depuis la raffinerie aux différentes destinations, Cafaro et Cerdà (2008) ont proposé un MILP (Mixed-Integer Linear Programming) à temps-continu avec mise-à-jour périodique pour un ordonnancement de distribution multi-produit unidirectionnelle avec plusieurs horizons temporels.

Leurs résultats ont montré que les séquences de pompage proposées par leurs systèmes étaient différentes de celles proposées par des méthodes classiques et elles étaient plus robustes face aux fluctuations de l'environnement, les séquences étaient plus courtes et plus nombreuses. Cependant, ces modèles mathématiques restent toujours pénalisants en termes de temps surtout lorsqu'il s'agit d'un ré-ordonnancement.

La programmation dynamique, inventée par Bellmann (1974), est un processus de décisions séquentiels qui permet d'établir une politique optimale où règles de prise de décisions telle que, pour chaque situation possible (état du système), elle détermine quelle décision (ou action) prendre dans le but d'optimiser une fonction objectif globale. Ceci permet de déduire la solution optimale d'un problème à partir d'une solution optimale d'un sous problème. Citons les travaux de Beasleya et Cao (1998) qui ont utilisé la programmation dynamique pour déterminer l'affectation de K équipes d'opérateurs humain à des tâches avec dates de début et de fin fixes sans que ces opérateurs humains ne dépassent leur temps de travail (8 heures par exemple).

3.1.2 Les méthodes approchées/ Heuristiques

Ces méthodes permettent de fournir des solutions de bonnes qualités en un temps raisonnable. Elles sont généralement utilisées quand les méthodes optimales ne permettent pas de résoudre le problème en un temps acceptable. On trouve, des approches basées sur les algorithmes Gloutons, d'autres sur la recherche locale.

Les algorithmes Gloutons sont utilisés pour des problèmes très difficiles car ils produisent des solutions en un temps réduit, au lieu de faire une exploration totale de toutes les solutions possibles, l'algorithme glouton, pour un sous-problème, sélectionne un choix particulier (selon des règles de priorité par exemple STF (Shortest Time First)) et passe au prochain sous-problème. Cependant, les solutions peuvent facilement ne pas être optimales.

Les méthodes de recherche locale partent d'une solution initiale et explorent un voisinage pour améliorer la solution. Parmi ces méthodes, on peut citer la méthode tabou, le recuit simulé ou encore les algorithmes génétiques.

La méthode de recherche tabou a été introduite par (Glover, 1986). Elle explore itérativement l'espace des solutions d'un problème en se déplaçant d'une solution courante à une nouvelle solution située dans son voisinage.

La méthode du recuit simulé prend ses origines dans la physique statistique, elle exploite le principe selon lequel un système physique qui est chauffé à une très haute température et ensuite graduellement refroidi atteindra en bout de ligne un niveau faible d'énergie correspondant à une structure moléculaire stable et forte. Elle a été introduite dans le domaine de l'optimisation combinatoire par Kirkpatrick (1983).

Les algorithmes génétiques, introduits par Holland (1975), sont des méthodes évolutives inspirées des principes de la sélection naturelle.

Une propriété commune à toutes ces méthodes, c'est qu'elles sont couteuses en terme de temps, de ce fait, elles sont dédiées à l'ordonnancement prévisionnel et à la recherche de la solution la plus efficace possible (Garey et Johnson, 1979), ceci est très utile pour un pilotage à moyen et long terme. Par contre, le pilotage à court terme où en temps réel nécessite une prise de décision rapide et efficace, l'ordonnancement doit être en mesure de produire des solutions dynamiquement, lorsqu'elles sont demandées, en fonction des paramètres, du système et de l'environnement, qui se présentent à cet instant, ce qui conduit au concept d'ordonnancement dynamique.

3.2 Ordonnancement dynamique et pilotage à court terme

Lorsqu'un ordonnancement est construit ou remis en question au fur et à mesure que le système de production évolue, on dit que c'est un ordonnancement dynamique (Elkhyari, 2003). L'ordonnancement dynamique prend en considération les événements imprévus, les données incertaines et l'incomplétude de l'information, sachant que :

Définition : *L'imprécision est un problème qui est dû au formalisme de la connaissance, il s'agit surtout de la connaissance qui ne peut pas être formalisée comme la description de l'état du système en langage naturel ou l'utilisation d'une marge de mesure*. (Chaari, 2010) Deux cas de figures se présentent alors. Le premier : c'est une imprécision sur les mesures, exemple : temps d'exécution d'une tâche est entre 8 et 10 unités. Le deuxième c'est l'utilisation de l'informel dans la description d'un événement, exemple : la distance est courte.

Définition : *L'incertitude est un doute qui concerne l'exactitude d'une donnée ou une hypothèse* (Dunne et Mu, 2008). Exemple : la machine X est en fonctionnement à l'instant T s'il n'y à aucune perturbation et si aucune tâche d'entretien n'est prévue. L'incertitude est souvent due aux contraintes externes, telles que les prévisions des demandes clients, les variations du marché.

Définition : *L'incomplétude existe lorsqu'une partie de l'information est connue*. (Chaari, 2010) Exemple : la tâche I s'exécutera sur la ressource R pour une durée inconnue.

Cependant, ces concepts sont en relation, car l'incomplétude implique l'incertitude et l'imprécision peut être considérée comme une incomplétude. De ce fait, nous considérons dans un cadre général l'incertitude des données dans la résolution des problèmes d'ordonnancement dynamique.

Selon la considération de l'incertitude, plusieurs classifications ont été proposées, citons à titre illustratif et non limitatif la classification de Suresh and Chaudhuri (1993) qui propose de classer toutes les approches réactives selon la nature de l'outil utilisé : approches conventionnelles, approches à base de connaissances et approches distribuées.

D'autres ont classé les approches d'ordonnancement selon leur degré de réactivité, tels que Davenport et al. (2000) où les auteurs distinguent deux approches : réactive et proactive.

L'approche réactive ne prend pas en considération les données incertaines dans la réalisation d'un ordonnancement initial qui peut être remis en cause si un événement inattendu se produit. L'ordonnancement proactif ou robuste prend en considération les événements futurs dans la réalisation d'un ordonnancement initial.

Dans notre équipe, plusieurs chercheurs travaillent dans le domaine de l'ordonnancement dynamique. Pour faciliter le positionnement relatif de ces travaux, nous avons construit une classification qui couvre toutes les approches d'ordonnancement dynamique (Chaari, 2010). En plus des approches proactives et réactives nous recensons aussi des approches hybrides. Nous utilisons cette classification pour positionner dans ce chapitre les travaux dans le domaine de l'ordonnancement dynamique.

3.2.1 Les approches proactives

Définition : *L'objectif des approches proactives est de prendre en compte l'incertitude lors de la réalisation de l'ordonnancement initial. Elle est utilisée pour rendre l'ordonnancement prédictif plus robuste.* (Davenport et al., 2000)

Dans l'approche proactive, les événements futurs sont pris en considération dans la réalisation de l'ordonnancement initial pour que cet ordonnancement soit robuste, c'est-à-dire que quelque soient les événements futurs, cet ordonnancement reste valide, alors souvent un ensemble de scénarios est proposé.

Souvent dans les méthodes proactives des indicateurs de la robustesse sont utilisés. Plusieurs indicateurs ont été étudiés et cités dans la littérature. Kouvelis et Yu (1997) proposent une classification des ces indicateurs : *Robustesse absolue*, lorsque le pire est prévu pour que certaines décisions restent valides dans toutes les conditions. *Ecart de la robustesse*, où pour chaque scénario possible un écart est calculé par rapport à la meilleure solution, et enfin, la *robustesse relative*, qui traduit le calcul du pourcentage de perte par rapport à l'optimalité. Ces indicateurs permettent de classer les différents scenarios afin de rester toujours proche de la solution optimale.

Souvent, des techniques de l'intelligence artificielle et notamment les méthodes à population sont utilisées pour la génération des solutions proactives. Sevaux et Sorensen (2002), utilisent l'algorithme génétique pour l'ordonnancement dans un problème à machine unique, l'algorithme génétique est utilisé pour générer un ensemble de solutions. La fonction objectif est considérée alors comme un critère de robustesse.

L'approche proactive sous-entend, qu'une partie des données du problème d'ordonnancement doit être obligatoirement parfaitement connue, ce qui n'est pas toujours le cas en réalité, de ce fait un autre type d'approches est développé. Ce sont les approches réactives.

3.2.2 Les approches réactives

Définition : *Les approches réactives se déroulent lors de la réalisation de l'ordonnancement.*

Elles se basent sur des informations mises à jour concernant l'état du système et éventuellement sur un ordonnancement prédictif initial, elles décident quand et où les activités doivent être exécutées. (Davenport et al., 2000)

Les approches réactives ne sont basées sur aucun ordonnancement initial, les décisions sont prises localement permettant de construire en temps réel une solution faisable.

Ces méthodes permettent de prendre en compte des variations importantes telles que des variations de durée, des variations sur les dates de début au plus tôt ou au plus tard, l'arrivée de nouvelles tâches,...

De ce fait, les approches réactives sont adaptées aux problèmes évolutifs. En effet, en même temps que le processus d'allocation des ressources évolue certaines informations deviennent disponibles permettant de prendre des décisions en temps-réel (Le quéré et al., 2003a; Pujo et Brun-Picard, 2002; Csaji et Monostori, 2006; Aissani et al., 2008a). Dans ce cas il est indispensable de définir plutôt une politique d'allocation, par exemple l'utilisation des règles de priorité telle que LPT ou SPT (Longuest ou Shortest Processing Time), ..., que l'on peut retrouver en détail dans (Pinedo, 1995).

Cependant, les performances de l'ordonnancement obtenu sont relativement faibles car il n'est connu qu'une fois réalisé.

3.2.3 Les approches hybrides

Nous distinguons deux classes :

Les approches prédictives-réactives

Ces approches sont basées sur un ordonnancement déterministe qui ne tient pas compte de futurs aléas. Il servira à guider les futures décisions grâce à des adaptations en temps réel pour tenir compte des perturbations. Ce type d'approche pallie les inconvénients des approches réactives puisque l'ordonnancement initial peut être adapté. Celui-ci est mis en place une fois réalisé, et en présence d'aléas, il est soit remplacé par un autre, voire modifié et réparé, soit recalculé à partir des nouvelles données, on parle alors de ré-ordonnancement.

Nous citons parmi les travaux les plus récents, ceux de Masuchun et Ferrell (2004), qui ont proposé une approche d'ordonnancement, génétique multi-objectifs, qui a pour objectif d'être efficace et stable aussi, ceci en intégrant ces deux objectifs dans la fonction objectif lors du ré-ordonnancement. Le ré-ordonnancement est périodique et l'efficacité est calculée par rapport à des critères de performance qui sont le Makespan et le retard, la stabilité est une combinaison linéaire de la déviation entre l'ordonnancement original et le nouveau.

Les approches proactives-réactives

C'est une classe identifiée dans les travaux d'Elkhyari (2003). Comme dans l'approche précédente, la résolution ici passe par deux phases : une première pour construire un ordonnancement prévisionnel mais aussi robuste, c'est-à-dire, qui tient compte de l'aspect incertain de certaines données.

Cet ordonnancement est construit par un algorithme dit *hors-ligne*. Cet ordonnancement est ensuite adapté en fonction de l'état du système lors de son exécution grâce à un algorithme dit *dynamique*.

L'idée de base de ces méthodes est de construire un ensemble d'ordonnancements statiques tels qu'il soit facile de passer de l'un à l'autre en cas de présence d'aléas.

L'une des plus anciennes méthodes de cette approche est ORABAID (ORdonnancement d'Atelier Basé sur une AIde à la Décision) (Demmou, 1977). Esswein et al. (2002, 2003), se sont basés sur ORABAID, et ont utilisé la programmation dynamique permettant d'obtenir un ensemble d'ordonnancements dont la date de fin est inferieur ou égale à une borne supérieure donnée.

Les algorithmes génétiques ont été aussi utilisés pour produire l'ensemble des ordonnancements robustes. Ainsi, face à une perturbation, plusieurs alternatives sont possibles (Aloulou et Portmann, 2002).

Plus récemment, Wu et al., (2009) ont développé une approche proactive-réactive, tel que, dans la première phase nous avons un ensemble d'ordonnancements robustes et dans une deuxième phase une ligne de base de cet ensemble est déterminée. Cette ligne représente les écarts entre les différentes solutions afin d'éviter la complexité de recherche dans la phase réactive, car plus les solutions s'approchent de cette ligne, plus le changement est léger.

Les performances de l'approche d'ordonnancement utilisée influent directement sur les performances du système de pilotage. Lorsque l'objectif d'un système de pilotage est d'être réactif et optimal en minimisant le temps et/ou les coûts, ces objectifs doivent être aussi les objectifs de la fonction d'ordonnancement au sein de ce système de pilotage. Comme énoncé dans le chapitre précédent, un système de pilotage doit être adaptatif pour faire face aux fluctuations de l'environnement de production, la fonction d'ordonnancement doit être alors suffisamment flexible pour permettre cette adaptabilité. La question qui se pose est ainsi : Quelle approche d'ordonnancement utiliser dans un système de pilotage pour que ce dernier soit *optimal, réactif et adaptatif ?*

Pour la conception d'un système de pilotage réactif, l'ordonnancement doit pouvoir prendre en charge les événements incertains, d'où le choix d'une approche par ordonnancement dynamique et plus particulièrement *l'ordonnancement réactif*. Le contrôle réactif et adaptatif dans une production à flux continu n'ayant pas été traité auparavant, notre contribution constitue un apport scientifique original. En effet, Wu et al. (2005) ont montré que l'ordonnancement dynamique dans l'industrie pétrolière notamment en raffinerie est mal traité et présente un fossé entre la théorie et la réalité du terrain ce qui oblige les ingénieurs du terrain à continuer à faire leur planification et ordonnancement presque entièrement manuellement, comme c'est le cas à la raffinerie d'Arzew.

Les événements incertains dans un système de production en général englobent aussi les tâches liées à la maintenance. Même si la maintenance préventive est planifiée à l'avance, elle constitue un événement perturbateur pour la production.

Les pannes d'équipements de production, naturellement, sont considérées comme des perturbations ainsi que les interventions de la maintenance corrective. De ce fait, l'ordonnancement réactif devrait pouvoir prendre en compte ce genre de perturbations.

Notre équipe a déjà étudié certains aspects liés à l'ordonnancement dynamique (Trentesaux et al., 1998), (Trentesaux et al., 2000), (Sallez et al., 2004), (Aissani et al., 2008a), (Aissani et al., 2008b) mais dans un contexte purement production. L'ordonnancement des tâches de maintenance a aussi été considéré (Le quéré et al., 2003b) dans un contexte différent : les TGV. Cependant, plusieurs problématiques de l'étude réalisée dans le contexte des TGV peuvent se retrouver dans le contexte de la Raffinerie d'Arzew, en particulier celle qui stipule que les tâches de maintenance préventives doivent être planifiées et gérées en relation avec les tâches de maintenance correctives imprévisibles afin d'assurer une fiabilité et une disponibilité du matériel de production.

Le pilotage de production est constitué de plusieurs fonctions, notamment l'ordonnancement. Mais il doit aussi tenir compte de la maintenance. Le schéma de la figure 1.2, montre une liaison directe entre l'ordonnancement et la gestion de la maintenance. Les tâches de maintenance peuvent être considérées comme des tâches de production pour lesquelles des ressources doivent être allouées pour des durées définies. Elles doivent donc être ordonnancées au même titre que les tâches de production mais avec des objectifs différents qui sont le respect des délais en limitant la perturbation de la production. C'est pourquoi dans la section suivante, nous présenterons la fonction maintenance et son rôle par rapport à la fonction d'ordonnancement et le système de pilotage en globalité.

3.3 L'Ordonnancement et la maintenance dans un système de pilotage

La fonction maintenance influence la fiabilité du système de production et le taux d'utilisation des ressources. Le but de la gestion de la maintenance est de réduire les temps de pannes et d'augmenter le taux de fiabilité avec le moindre coût.

Deux types de maintenance sont identifies dans le cadre de notre projet (voir NF X060-10, NF X 060-11, NF EN 13 306, et Retour et al., 1990) : maintenance préventive et maintenance curative.

3.3.1 Maintenance préventive et ordonnancement

Définition « AFNOR » norme X60- 010 : « *C'est la maintenance effectuée dans l'intention de réduire la probabilité de défaillance d'un bien ou la dégradation d'un service rendu* ». C'est une intervention de maintenance prévue, préparée et programmée avant la date probable d'apparition d'une défaillance.

La problématique d'ordonnancement des tâches de maintenance préventive, est de les placer sur un plan de production de la façon la plus efficace, c'est-à-dire sans atténuer la production tout en respectant la régularité de ces tâches de maintenance. Souvent, des approches d'optimisation sont utilisées, même si parfois elles sont couteuses en temps mais en préventif, c'est plutôt la qualité du résultat qui nous intéresse en dépit du temps.

Par exemple, Cavory et al. (2001) ont vérifié par simulation que l'utilisation des AGs en ordonnancement préventif de maintenance permettent un gain dans le temps de production de 7.5% en rendant les ressources de production encore plus disponibles. Cependant, ces plans restent indépendants de la production.

Certains travaux se sont aussi intéressés à la définition des stocks intermédiaires pour assurer une continuité dans le processus de production en cas de pannes ou de révisions préventives (Van der Duyn Schouen et Vanneste (1995)).

Plus récemment, Chelbi et Ait-Kadi (2003) ont proposé un modèle analytique pour déterminer la taille des stocks intermédiaires et la périodicité des révisions dans une unité de production flexible avec des tâches de maintenance préventive régulières avec des durées aléatoires. Ce type de tâches de maintenance préventive sera considéré dans notre cas où les espaces de stockage (les réservoirs) ne sont pas affectés par ces tâches (ex. Arrêt de pompe, inspection de turbines).

Kenne et Boukas (2003) ont traité l'ordonnancement conjoint des tâches de maintenance et de production dans un atelier flexible. Un modèle de contrôle hiérarchique à deux niveaux a été développé considérant l'âge de l'outil et son taux de panne pour la mise en place d'une politique de maintenance prévisionnelle.

3.3.2 Maintenance curative/corrective et ordonnancement

Selon la même norme **NFX 60-010**, « *La maintenance curative est une opération de maintenance effectuée après détection d'une défaillance* », Elle a pour but de rétablir la ressource à son état fonctionnel après une panne. Ces activités peuvent être des réparations, des modifications ou aménagement ayant pour objet de supprimer les défaillances.

L'arrivée des tâches de maintenance curative est un événement incertain, de ce fait, leur prise en compte lors de la réalisation d'un ordonnancement préventif est une tâche difficile. Des approches spécifiques ont été développées pour l'ordonnancement des tâches de maintenance curative (Le Quéré et al. 2003b). Par exemple pour les approches stochastiques, nous pouvons citer les travaux de Kenne et Gharbib (2004), qui ont développé un modèle représentant la durée et le temps d'arrivée des tâches de maintenance curative en utilisant une approche multi-agents, et dont le module décisionnel est un processus Markovien. Cette utilisation des processus Markoviens permet au système de prendre des décisions à tout moment et qui s'approchent de plus en plus de l'optimalité et nous semble une approche intéressante.

Selon la littérature, très peu de travaux se sont intéressés à l'ordonnancement des tâches de maintenance préventive et corrective couplées à un plan de production. C'est surtout parce que les outils théoriques nous obligent souvent à choisir entre un ordonnancement statique ou dynamique et non pas les deux à la fois, ce qui est notre objectif dans ce projet : développer un système de pilotage réactif qui prend en charge tous les événements incertains en particulier la présence des tâches de maintenance, qu'elles soient préventives ou réactives pour mieux contrôler un système de production à flux continu.

Le système de pilotage que nous nous proposons de développer est alors basé sur une approche hybride prédictive-réactive car le système doit tenir en compte des plans de production établis par le staff de l'entreprise et les plans de maintenance préventive, mais il doit aussi s'adapter aux fluctuations de l'environnement de production : demande non planifiée, pannes de ressources de production, tâches de maintenance curative…

Dans la section suivante, nous montrerons les différentes approches de modélisations qui ont été utilisées pour la modélisation des systèmes de pilotage de production.

4 Etat de l'art : modélisation et approches pour les systèmes de pilotage de production à ordonnancement dynamique

Plusieurs méthodes de modélisations pour les entités des systèmes de pilotage ont été envisagées afin de tenir compte de leurs particularités. En analysant l'histoire des systèmes de pilotage nous pouvons distinguer principalement deux phases :

Une première phase, où le système de pilotage était vu comme un seul bloc décisionnel, c'est-à-dire constitué d'une seule entité, le pilotage était alors *centralisé* dans cette entité.

Puis une deuxième phase, où le pilotage a été scindé sur plusieurs entités organisées de différentes manières, c'est le pilotage *décentralisé*.

Dans ce qui suit, nous analyserons les différentes propriétés de chaque phase et quelles sont les méthodes de modélisation employées pour chaque phase.

4.1 Mode centralisé

Dans ce mode, le système de pilotage est constitué d'une entité décisionnelle qui pilote et contrôle toutes les unités de production, de manière totalement centralisée ou éventuellement hiérarchisée. Cette entité intègre un mécanisme de prise de décision que ce soit par simulation, approches exacte ou par apprentissage.

Comme vu auparavant, dans un ordonnancement réactif, il est plutôt question d'établir une politique d'allocation (Pinedo, 1995). Certaines approches réactives utilisent des règles de priorité. Une règle de priorité est une règle qui indique quelle tâche prendre de la file d'attente d'une ressource lorsque cette dernière se libère (Rajendran et Holthaus, 1999), exp : FIFO *First In First Out*. Ces règles dépendent des objectifs visés par l'ordonnancement, et plusieurs règles peuvent être exploitées dans un même système, dans ce cas le choix des règles à utiliser est alors dynamique dépendant de l'état du système.

Selon Priore et al. (2001), cette pratique permet aux systèmes d'améliorer considérablement leurs performances. La sélection des règles se fait selon deux approches : par simulation ou selon des modèles de connaissances qui impliquent l'apprentissage.

4.1.1 Par simulation

Une simulation de toutes les règles de priorité peut être lancée pour élire à chaque état du système la règle qui donne le meilleur profit.

Kutanoglu et Sabuncuoglu (1999) ont utilisé la simulation pour comparer plus de vingt règles de priorité dans un job-shop dynamique avec un caractère pondéré du retard de l'arrivée des tâches. Dans le même contexte nous retrouvons aussi les travaux de Kim et Kim (1994) et Degres et al. (2004).

4.1.2 Par apprentissage

Le but ici est de construire un modèle de connaissances en hors-ligne, c'est-à-dire dans une phase d'apprentissage qui pourra être utilisé par la suite en enligne dans une phase d'exploitation pour déterminer à tout instant la règle de priorité à utiliser.

Un processus composé de cinq phases a été proposé par Priore et al. (1998) pour la réalisation du modèle de connaissance : définition des paramètres de contrôle, génération des exemples expérimentaux, détermination des conditions d'activation des règles, sélection de la meilleure règle parmi celles activées et enfin évaluation des performances. Plus les performances sont insuffisantes plus le processus est réitéré.

Dans ce même processus, Chen et Yih (1996) ont utilisé l'approche neuronale pour déterminer les paramètres primordiaux pour la sélection des règles à activer.

Certains chercheurs ont utilisé une hybridation Simulation –apprentissage. Pierreval et Ralambondrainy (1990) ont développé un méta-modèle constitué de règle de production déduite d'un apprentissage sur des résultats de simulation. Cette technique a été aussi utilisée par Huyet et Paris (2004) pour le dimensionnement d'ateliers.

4.2 Mode décentralisé

Des approches décentralisées peuvent également être utilisées pour la conception des systèmes de pilotage réactifs, ces modèles sont constitués de plusieurs entités décisionnelles qui interagissent entre-elles sans nécessairement de subordination hiérarchiques.

Dans ce cadre, nous citons les approches suivantes : l'approche multi-agents, l'approche holonique et les approches bioniques (Bousbia et Trentesaux, 2002).

4.2.1 Modélisation multi-agent

Cette dernière décennie et depuis l'invention de ce nouveau paradigme, le concept d'agent a été largement utilisé pour la résolution des problèmes complexes dans un environnement distribué où la coopération et l'interaction sont requises. Un système de pilotage multi-agent est constitué d'un ensemble d'agents autonomes qui coopèrent pour prendre des décisions de pilotage. Un modèle générique pour les systèmes de pilotage de production a été proposé par Querrec et al. (1997), chaque entité du système est représentée par un agent autonome capable de prendre des décisions localement, l'agent est constitué de deux parties : une opérative et l'autre de contrôle (voir figure 2.1). Il perçoit et agit sur son environnement à travers des messages. Les propriétés d'un agent sont donc des attributs qui caractérisent son état interne et des méthodes qui constituent les actions potentielles de l'agent.

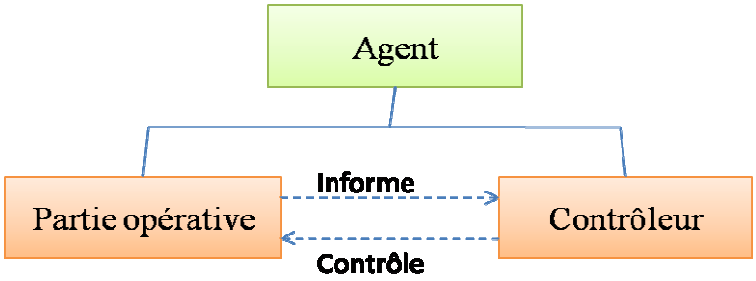


Figure 2.1. Modèle conceptuel d'un élément générique

Plusieurs applications de cette approche ont été proposées, citons par exemple :

Sohier et al. (1996) ont proposé un système multi-agents pour le contrôle d'une cellule flexible de production qui permet une réaction aux événements de production. Ils ont défini deux types d'agents : les agents statiques et les agents dynamiques. Les agents statiques modélisent les équipements de la cellule. Une instance est créée une fois pour toutes lors de la conception du système multi-agents. Les agents dynamiques modélisent les pièces. Des instances des agents dynamiques sont créées lors de la résolution des problèmes de production. Une action est déclenchée lorsqu'un agent doit évoluer (recherche de satisfaction) et lorsque les conditions nécessaires à l'exécution sont vérifiées.

Schneider et al. (1998) ont proposé une modélisation par un Processus Décisionnel Markovien (MDP) pour modéliser la prise de décision chez l'agent dédié à la résolution des problèmes d'ordonnancement avec une modélisation stochastique des incertitudes de l'environnement de production (aléas de production, variation de la demande…). La résolution de ce MDP se fait par un algorithme de la programmation dynamique « Value Function » qui peut générer des solutions d'ordonnancement optimales en enligne. Ces techniques permettent généralement au système d'être réactif, car elles proposent toujours des solutions en ligne.

Chaque agent doit être capable de décider individuellement, sans l'assistance des autres agents. Même si chaque agent est limité à son propre problème d'ordonnancement, il doit être capable de proposer une solution réalisable dans un environnement de processus distribué à plusieurs agents. Mais lorsque plusieurs agents sont ensembles, ils rencontrent des problèmes «sociaux», comme l'adaptation, la résolution des conflits, la compétition... Certains travaux ont ainsi été réalisés en se focalisant sur le processus décisionnel ou mode de raisonnement des agents, d'autres sur le mode de leur interaction ou coopération.

Modélisation orientée mode de raisonnement

Pour un atelier job-shop flexible Lau (1997) a développé un système de contrôle basé sur une approche multi-agents, où chaque agent a une possibilité d'apprentissage supervisé à l'aide d'exemples fournis par simulation. Leur étude a montré que même si chaque agent est limité à son propre problème d'ordonnancement, il est capable de proposer une solution réalisable dans un environnement de processus distribué. Ils ont montré aussi que ces agents peuvent apprendre en groupe pour améliorer leurs performances et s'adapter à leur environnement par un apprentissage renforcé (retour d'expérience).

Cet apprentissage individuel et collectif appliqué aux SMAs peut nous être très utile si on fait une analogie entre le Job-shop flexible et le système de production à flux continu en tenant compte des particularités de chacun.

Vacher (2000) a utilisé une autre technique d'apprentissage : l'approche génétique. Vacher a présenté un système multi-agents qui utilise des algorithmes génétiques multi-objectifs pour l'ordonnancement d'un atelier Job-Shop. Le système est considéré comme un ensemble d'entités actives dont la structure et le rôle sont à préciser. Ces entités sont des agents génétiques qui se reproduisent pour donner la meilleure population (car ils arrivent à se renforcer par croisement avec d'autres agents ayant réalisé de bonnes actions, à s'altérer en modifiant la liste des règles d'actions qui caractérisent l'agent, à retomber en état de sommeil, ou à se détruire aux vus des résultats qu'ils ont engendrés). Le système propose à chaque fois un ordonnancement qui soit amélioré et adapté à de nouvelles situations.

Les travaux de Schneider et al. (1998), ont montré que la modélisation d'un problème d'ordonnancement en MDP permet de le résoudre dynamiquement. Des approches exactes ont été utilisées pour résoudre ces MDPs, telles que la programmation dynamique. Par exemple, Csaji et Monostori (2005) ont proposé un système multi-agent pour l'ordonnancement réactif. Le problème d'ordonnancement est alors modélisé comme un Processus Décisionnel Markovien au sein des agents qui sont inspirés du modèle PROSA (Hadeli et al., 2004) où trois types d'agents sont développés : agent ordre (commandes), agent produit (ensemble des tâches) et l'agent ressource représentant les ressources de production. Ces agents sont dotés d'un module décisionnel basé sur le MDP d'ordonnancement mais indépendamment. Chaque agent a ses propres intentions et ses propres objectifs qu'il essaye d'accomplir alors, il peut entrainer le système vers ses propres objectifs en le déviant de l'objectif global.

De ce fait, chaque agent doit être jugé selon l'efficacité de sa contribution à l'objectif global de production. Utilisant ses retours d'informations, chaque agent peut améliorer sa performance, par l'apprentissage renforcé, comme membre d'une équipe. Ce type de technique, système multi agent et apprentissage par renforcement, a donné des résultats satisfaisants (Aissani et al., 2008a). Cette technique d'apprentissage par renforcement est utilisée pour enrichir la base de connaissance d'un agent et améliorer ses performances au sein d'un groupe. Ceci constitue l'idée de base de notre proposition pour un système de pilotage réactif et adaptatif.

Modélisation orientée mode d'interaction

Certains chercheurs sont intéressés au mode de communication et d'interaction. Dans un système multi-agent qui substitue les éléments du système de production, Taghezout et Zaraté (2008) ont présenté une extension du Contact Net Protocol (Smith, 1980) pour prendre en considération les événements incertains de l'environnement, car même si un contrat est établi et que les données ont changées, des directives ont été ajoutées pour pouvoir adapter et mettre à jour ce contrat. Mentionnons également les travaux de Krothapalli et Deshmukh (1997) et Maione et Naso (2001) tous deux axès sur le mode d'interaction inter-agent pour la résolution des problèmes de contrôle de production.

4.2.2 Approche holonique

C'est une approche plus récente. Le terme 'Holon' est d'origine grecque *holos* qui signifie entier, complet avec tous ses composants et le suffixe « *on* » suggère une particule individuelle. Initialement ce concept a été utilisé pour décrire des entités de base dans des phénomènes sociaux par Koestler (1967). Un holon selon Koestler est un composant-modèle à deux faces: une face regardant en bas qui donne des directives aux entités de bas niveau et l'autre face regarde en haut pour faire partie d'un autre holon. Puis le concept a été adapté pour les systèmes manufacturiers, un holon correspond à un élément d'un système autonome et coopératif dédié à la transformation, au transport, au stockage et/ou à la gestion des objets physiques ou informationnels (Van Brussel et al., 1998). De nombreuses modélisations ont été élaborées à base de ce modèle, nous retrouvons l'architecture PROSA (Product Resource Order Staff Agents) (Van Brussel et al., 1998), qui se base sur 4 types d'holons, trois qui représentent des entités de fabrication : ce sont les ordres et les ressources pour la logistique, les produits pour le process et un agent staff qui coordonne les autres.

L'architecture ADACOR (Leitao et Restivo, 2008) suppose toujours des holons opérationnels et un holon superviseur pour basculer d'une organisation hiérarchique en recherche d'optimisation à organisation décentralisée lorsqu'il faut faire face aux événements incertains.

Il y'a aussi l'apparition de la notion d'augmentation, c'est-à-dire, augmenter le système physique d'une partie virtuelle (Sallez et al.,2009 ; Zbib. N, 2010), cette notion permet d'implémenter les modèle holoniques. Les RFID sont une technologie exemple de cette implémentation. Les RFID ont non seulement permis l'identification des produits mais aussi leur faire transporter leurs informations, ce qui donne aux produits une existence physique, informationnelle et abstraite en même temps. Nous retrouvons par exemple les travaux de Vrba et al., (2008) et Pannequin et al.,(2008).

4.2.3 Modélisation Bionique

Les travaux qui relèvent de cette catégorie essaient de reproduire des phénomènes naturels pour la modélisation d'autres systèmes tels que les systèmes industriels. Plusieurs analogies sont alors possibles :

Colonie de fourmis : En s'inspirant du mode organisationnel d'une colonie de fourmis, une entité décisionnelle peut être une fourmi et les phéromones sont le moyen de communication. Naturellement, ces entités sont organisées selon un agencement hétérarchique. Dans ce contexte nous retrouvons les travaux de Cicirello et Smith (2001), Xiang et Lee (2008).

Certains ont proposé une hybridation de deux classes de modélisation. Berger et al., (2004), Zulch et Stock (2006) ont proposé de considérer les holons produits comme des fourmis pour un contrôle par le produit, les phéromones permettent aux produits de retrouver le meilleur chemin pour exécuter leurs tâches.

5 Conclusion

L'objectif de ce chapitre était de réaliser un état de l'art sur les approches de modélisation des systèmes de pilotage selon le type de sa fonction d'ordonnancement et les méthodes de résolution.

Concernant la structure du système de pilotage, nous avons montré qu'une approche décentralisée permettait au système d'être flexible est réactive grâce à la distribution des centres décisionnels du système. Nous avons montré aussi, qu'un système de pilotage doté d'une fonction d'ordonnancement dynamique permettait aussi système de pilotage de prendre des décisions rapides et efficaces pour un pilotage à court terme ou en temps réel. Parmi les approches de modélisation décentralisées, nous avons opté pour une approche multi-agent car ce sont les approches les plus abouti dans la résolution des problèmes décentralisés et complexe. En outre, les techniques d'apprentissages ont été largement utilisées pour la résolution des problèmes d'ordonnancement notamment pour l'amélioration de la qualité des solutions proposées.

Cependant, il reste à choisir la technique d'apprentissage et l'architecture du système multi-agent qui permet au système de rester dynamique et de s'adapter à son environnement. Nous avons constaté que la modélisation d'un problème d'ordonnancement en MDP, permet de le résoudre dynamiquement. Un MDP peut être résolu en utilisant l'apprentissage par renforcement qui permet aux entités décisionnelles d'un système d'améliorer leurs performances tout en restant réactives et autonomes. Ceci a été déjà montré dans (Aissani et al., 2008a) et va être exploité pour élaborer notre modèle de système de pilotage adaptatif pour la production a flux continu et qui va être spécifié dans le prochain chapitre.

Chapitre 3 : Spécification d'un système de pilotage réactif et adaptatif et modélisation multi-agent

1 Introduction

Dans le chapitre précédent nous avons montré que dans un environnement de production complexe et dynamique le pilotage de production doit être réactif pour faire face aux fluctuations de l'environnement. En outre, ce système doit être aussi adaptatif pour pouvoir améliorer au fur et à mesure la qualité de ses décisions.

Nous avons également montré qu'une modélisation distribuée de type multi-agents offre de grandes potentialités en terme de résolution complexe et de réactivité, et lorsqu'elle est couplée à un apprentissage réactif tel que l'apprentissage par renforcement elle permet au système d'être adaptatif à son environnement.

A partir des résultats de notre analyse présentée dans le chapitre précédent, l'objectif de ce chapitre est de spécifier l'architecture générale de notre système de pilotage et de proposer les modèles agents. Le chapitre suivant détaillera les mécanismes d'apprentissage.

La conception d'un système multi-agent passe par différentes étapes, d'abord l'identification des différentes entités ou différents centres de décision qui peuvent être substitués par des agents, puis l'identification de leurs attributs, spécifications, architectures et leurs rôles. Enfin, le mode d'interaction entre ces différents agents, scénarios de communication et protocole de communication doivent être définis. Le plan de ce chapitre respecte cette séquence. Au préalable, nous préciserons quelques concepts importants qui concerneront notre modélisation.

2 L'intelligence artificielle distribuée

Contrairement à l'approche centralisée de l'IA, l'intelligence artificielle distribuée (IAD) vise la distribution de l'expertise sur un ensemble de composants appelés agents qui communiquent pour atteindre un objectif global. Les composants doivent être capables de raisonner sur les connaissances et les capacités des autres dans le but d'une coopération effective. Pour ce faire, ils doivent être dotés de capacités de perception et d'action sur leur environnement et doivent posséder une certaine autonomie de comportement.

Nous commencerons par définir les principaux concepts qui relèvent de l'IAD : agent, groupe, rôle, mode de communication et contrôle.

2.1 La notion d'agent

En dépit de l'absence d'une définition consensuelle de ce qu'est un agent, on retient la définition classique suivante proposée par Ferber (1995) :

« *Un agent peut être défini comme une entité (physique ou abstraite) capable d'agir sur elle même et sur son environnement, disposant d'une représentation partielle de cet environnement, pouvant communiquer avec d'autres agents et dont le comportement est la conséquence de ses observations, de sa connaissance et des interactions avec les autres agents*».

L'agent est caractérisé par son degré d'observation, sa capacité d'agir, son facteur décisionnel et ses dispositifs de communication.

2.2 Système multi-agents et organisation

Lorsque plusieurs agents évoluent dans un même environnement, c'est-à-dire observent et agissent sur cet environnement, ils constituent un système multi-agents.

Définition : « *Une organisation peut être définie comme un agencement de relations entre composants ou individus qui produisent une unité, ou système, dotée de qualités inconnues au niveau des composants ou individus.*

L'organisation lie de façon interrelationnelle des éléments ou événements ou individus divers qui dès lors deviennent les composants d'un tout. Elle assure solidarité et solidité relative, donc assure au système une certaine possibilité de durée en dépit de perturbations aléatoires » (Ferber 1995).

2.3 Mode de communication

Chaque agent a une vue partielle de son environnement, de ce fait, il a une vue incomplète qui peut être enrichie par un échange d'informations avec les autres agents de son environnement. La communication se fait principalement selon trois modes :

- *L'acheminement par voie direct* est le mode le plus simple: lorsqu'un agent désire envoyer un message, celui-ci est pris par le canal de communication et l'apporte directement à son destinataire (ou à tous les destinataires potentiels dans le cas d'un message diffus). C'est le type d'acheminement qui est le plus souvent pratiqué dans les systèmes multi-agents cognitifs.
- *Le mode d'acheminement par affichage* est moins utilisé dans les systèmes multi-agents. C'est typiquement le mécanisme de communication des "petites annonces". Un agent, s'il désire communiquer, place son message dans un espace commun, appelé tableau d'affichage ou tableau noir, visible par tous les agents (ou tous ceux d'une classe particulière).
- *L'acheminement par voie indirecte*, en s'inspirant des deux modes précédents, dans un environnement multi-agents il existe plusieurs ressources de communication, lorsqu'un agent désire communiquer il place son message dans l'une des ressources correspondantes et selon le type de la ressource, les agents consultent son contenu. La communication par phéromone relève de ce mode d'acheminement.

2.4 Mode d'interaction

En utilisant ces modes de communication, un regroupement d'agents aura un ensemble de comportements pour aboutir à leurs buts ou aboutir à l'objectif global. Cet ensemble de comportements est l'interaction.

Définition : « *Une interaction est une mise en relation dynamique de deux ou plusieurs agents par le biais d'actions réciproques* » (Ferber 1995)

Selon le mode d'interaction voulu (coordination, coopération…), différents protocoles de communication existent pour contrôler le système multi-agents.

2.5 Le contrôle

Le contrôle se focalise sur l'utilisation des capacités d'introspection et de protocoles d'interaction multi-agents pour la composition de fonctionnalités. Le contrôle a pour rôle de préciser le comportement de chaque agent par rapport à sa propre capacité décisionnelle et par rapport à sa communauté. L'évolution de toute la communauté d'agent doit être aussi maitrisée pour que le système n'évolue pas anarchiquement.

Le contrôle est défini alors selon trois dimensions : le niveau de coopération, l'organisation du groupe d'agents et la dynamique de l'évolution.

Le niveau de coopération : la coopération est la forme générale de l'interaction, elle peut être de 4 niveaux selon la compatibilité des buts des agents, la disponibilité des ressources et la compétence des agents (Ferber 1995) (Tableau 3.1).

	Collaboration Simple	**Encombrement**	**Collaboration coordonnée**	**Compétition**
Buts	*Compatibles*	*Compatibles*	*Compatibles*	*Incompatibles*
Ressources	*Suffisantes*	*Insuffisantes*	*Insuffisantes*	*Suffisantes*
Compétences	*Insuffisantes*	*Suffisantes*	*Insuffisantes*	*Insuffisantes*

Tableau 3.1. Les niveaux de coopération

La dynamique de l'évolution : Il s'agit ici de déterminer comment les agents évoluent, s'ils acquièrent de nouvelles compétences, les conditions de leurs créations ou destruction. Il faut aussi déterminer l'évolution de leur organisation.

L'organisation du groupe d'agents : Il s'agit de la façon dont les agents sont organisés et liées entre eux, est ce que c'est en groupes ou en individus indépendants, voire aussi les règles à respecter pour communiquer entre eux : un agent peut communiquer avec tous les agents ou juste un groupe défini par une condition. Mentionnons également l'aspect relatif aux priorités, un agent peut être prioritaire par rapport à un autre dans l'accès à une ressource. Plusieurs modèles organisationnels ont ainsi été développés par la communauté multi-agent.

2.6 Les modèles organisationnels

Un modèle organisationnel, en tenant compte des aspects agents, interactions et environnement des systèmes multi-agents, traite de l'aspect social.

Un modèle organisationnel est une modélisation des fonctionnements récurrents d'un ensemble d'agents en interaction afin de spécifier les comportements globaux que l'on aimerait voir émerger des interactions entre les agents.

Plusieurs modèles organisationnels ont été développés, des modèles basés sur la notion de groupe (Hirsh et al., 2003), la théorie des activités (qui est une théorie de la psychologie sociale sur le développement, la transformation et la dynamique de l'activité de travail collectif de l'homme) (Omicini et al., 2003) ou les rôles (Cabri et al., 2003).

Le rôle est l'abstraction d'un comportement ou d'un modèle de conduite, rattaché à un statut, et pouvant interagir avec d'autre rôles (Fowler, 1997). Une organisation peut être alors vue comme un ensemble de rôles et de relations entre ces rôles qui réalise une fonction globale. L'avantage de l'utilisation des rôles dans les modèles organisationnels est double :

Premièrement, il permet la séparation entre les problèmes algorithmiques et les questions d'interaction dans le développement d'applications multi-agents (Cabri et al., 2002).

Deuxièmement, il permet la réutilisation des solutions et expériences. En fait, les rôles sont liés à un scénario d'application, et les concepteurs peuvent exploiter les rôles précédemment définis pour des applications similaires.

En se basant sur ces notions, nous analyserons quelques modèles des plus connus pour en choisir un qui sera à la base de nos spécifications.

2.6.1 Le modèle BRAIN

Le modèle BRAIN ((Behavioral Roles for Agent INteractions) Cabri et al., 2003) présenté dans la figure 3.1 présente les propriétés suivantes :

1. Un modèle d'interaction multi-niveaux simple et général à base de rôles
2. Une formalisation basée sur XML pour la description des rôles
3. L'infrastructure de l'interaction est basée sur cette description des rôles ce qui permet de contrôler l'émergence du système. Elle est représentée par des paires (action-événement).
4. L'implémentation de ce modèle a été faite sur la plateforme JADE (Java Agent DEvelopment Framework) donnant lieu à un « Rolesystem »

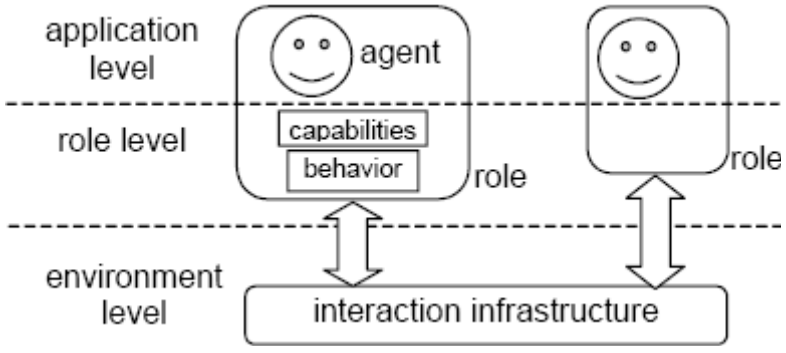


Figure 3.1. La structure du système

Même si ce modèle est considéré parmi les modèles les plus complets, il présente un inconvénient majeur lié à la difficulté de son implémentation et de l'utilisation des API (Application Programming Interface : ensemble de classes, fonction, etc., mises à disposition dans des bibliothèques Java pour les utiliser en programmation) de RoleSystem.

2.6.2 Le modèle RoleEP

Le modèle RoleEP (Role Based Evolutionary Programming (Ubayashi et Tamai, 2000)) est basé sur la notion des agents mobiles et leur domaine où ils collaborent. Ce domaine est considéré comme un environnement et la fonction que prend un agent dans cet environnement est son rôle. Un agent peut avoir un ou plusieurs rôles.

Ce modèle offre un dynamisme important dans l'incarnation ou l'annulation d'un rôle pendant l'exécution ce qui offre un potentiel d'adaptation. Toutefois, ce modèle est plutôt spécifique puisqu'il s'applique à des agents mobiles et exige que chaque agent doit avoir un rôle adéquat pour communiquer alors que ceci n'est pas toujours exigé dans certaines situations.

2.6.3 Le modèle AALAADIN

Le modèle Aalaadin a été développé par (Ferber et Gutknecht, 1998) pour la conception des systèmes multi-agents. Le modèle est basé sur trois concepts : l'agent, le groupe et le rôle. La figure 3.2 présente un diagramme de ce modèle.

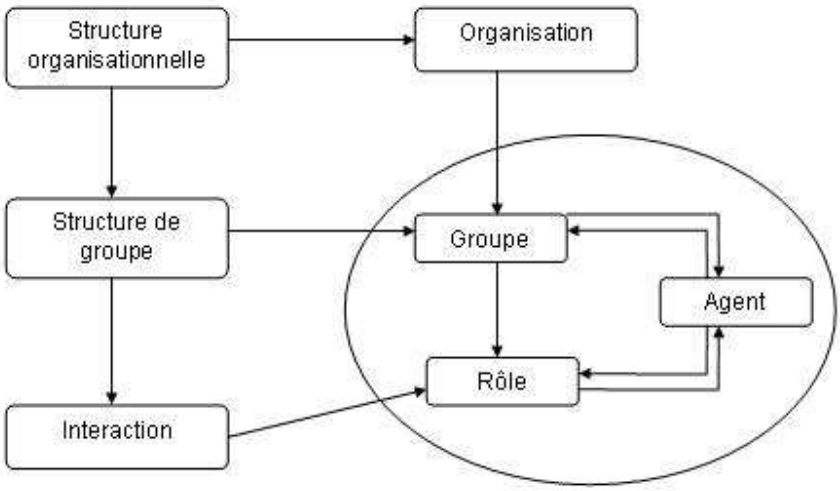


Figure 3.2. Le modèle Aalaadin

L'agent

Dans ce modèle, aucune contrainte ou pré-requis sur l'architecture interne de l'agent n'est posée, ni sur le formalisme ni sur un modèle particulier à utiliser pour décrire son comportement. Un agent est simplement défini comme une entité autonome communicante qui joue des rôles au sein de différents groupes.

On pourrait reprocher à cette définition son flou, mais elle est intentionnellement générale pour permettre à chaque concepteur d'agents de choisir parmi les modèles classiques le plus adapté à son application, et de ce fait permettre une liberté de création et de conception pour les futurs utilisateurs de ce modèle.

Le groupe

Le groupe est défini dans Aalaadin, comme la notion primitive de regroupement d'agents. Chaque agent peut être membre d'un ou de plusieurs groupes. Dans sa forme la plus simple, un groupe peut être vu comme un moyen d'identifier, par regroupement, un ensemble d'agents, d'une manière plus évoluée, le groupe peut être vu comme un SMA usuel. Un groupe peut être fondé par n'importe quel agent.

Le rôle

Le rôle est une représentation abstraite d'une fonction, d'un service ou d'une identification d'un agent au sein d'un groupe particulier. Chaque agent peut avoir plusieurs rôles, un même rôle peut être tenu par plusieurs agents, et les rôles sont locaux aux groupes.

La maîtrise de l'hétérogénéité des situations d'interaction est rendue possible par le fait qu'un agent peut avoir plusieurs rôles distincts au sein de plusieurs groupes, et que les interactions sont toujours locales à un groupe.

Les modèles basés sur Aalaadin sont facilement implémentés grâce à l'existence d'une plateforme de développement appelée MadKIT (http:\\www.madkit.org;Voir les détails de présentation dans Annexe 2).

Une gamme plus large de modèles est présentée dans les travaux de Cabri et al. (2004) qui présentent Aalaadin comme l'un des modèles les plus ouverts et les plus outillés (figure 3.3). Nous reprenons alors un graphe illustratif, qui positionne Aalaadin au centre d'un triangle dont les sommets sont : la modélisation, l'analyse et l'implémentation, ce qui peut constituer un modèle de référence très complet. Nos motivations pour le choix de ce modèle proviennent pour l'essentiel de cette étude.

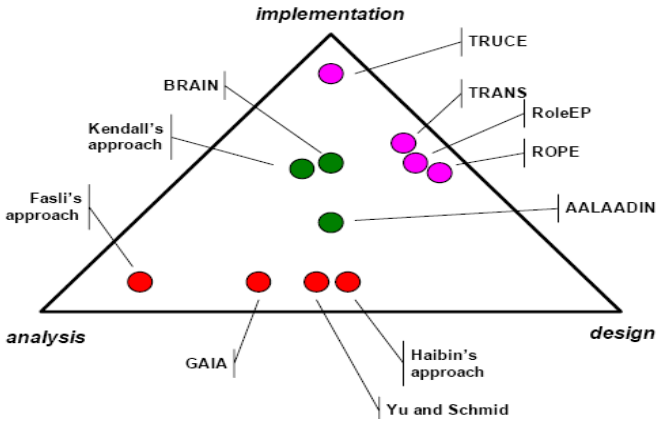


Figure 3.3. Une représentation graphique de la comparaison des approches par rapport à certains critères

Dans Aalaadin, une organisation est vue comme un cadre (un « framework ») pour les activités et les interactions grâce à la définition de groupes, de rôles et de leurs relations. La notion d'organisation correspond donc ici à une relation structurelle entre les rôles définis au sein des groupes.

3 Structure globale du système de pilotage

Dans un système de pilotage nous pouvons distinguer deux types d'entités décisionnelles différentes qui correspondent, dans notre cadre applicatif, aux composants physiques du système à piloter :

- Les entités qui offrent les services telles que les ressources de production (les bacs de stockage, les pompes de transfert…)
- Les entités qui demandent les services, principalement les produits à fabriquer (les carburants, les huiles finies…)

Dans ce mémoire, nous partons du principe que ces entités décisionnelles peuvent être substituées en agents, ce qui constitue une approche de modélisation multi-agent assez classique. Par conséquent, nous distinguons les *agents ressource* et les *agents produit*.

Selon le modèle Aalaadin, nous regroupons les agents en groupes. Un point commun regroupe les ressources de production : c'est leur spécialité ou le type de l'huile de base à utiliser. Présentés dans la figure 3.4, les agents ressources sont assemblés par groupe où chaque groupe représente la spécialité de ces ressources (par exemple : le groupe dont l'huile de base est SAE30). Dans ce groupe, tous les agents sont de même niveau.

En dehors de ces groupes nous retrouvons les agents produits qui représentent les produits à fabriquer (huile finie) et qui disposent d'un ensemble de tâches à faire exécuter par les agents ressources. Ces agents sont aussi de même niveau que les agents ressources. Tous ces agents sont liés par un lien de communication à double sens entre tous les agents.

De manière classique et similaire à ce que nous avons étudié (PROSA, ADACOR, etc.), nous proposons l'adjonction d'un agent superviseur qui aura pour rôle de s'assurer de la convergence de ce système vers son objectif global. Cet agent collecte des informations de suivi des agents du système afin de les analyser et d'évaluer l'évolution des performances du système. Cet agent constitue un sommet dans la structure du système et ses liaisons avec le reste des entités du système constituent les liaisons verticales. Il permet en outre de mettre en œuvre l'interopérabilité avec les niveaux plus élevés tels que le niveau ERP.

En analysant la structure globale du système ainsi présenté nous pouvons dire qu'il est structuré selon un agencement hétérarchique au sens large tel que défini dans (Trentesaux, 2002, Trentesaux, 2007) (figure 3.4): On dit qu'une organisation est *Hétérarchique* lorsqu'un système décisionnel distribuéfavorise une interaction à double sens entre toutes les entités de même niveau ou de niveaux différents.

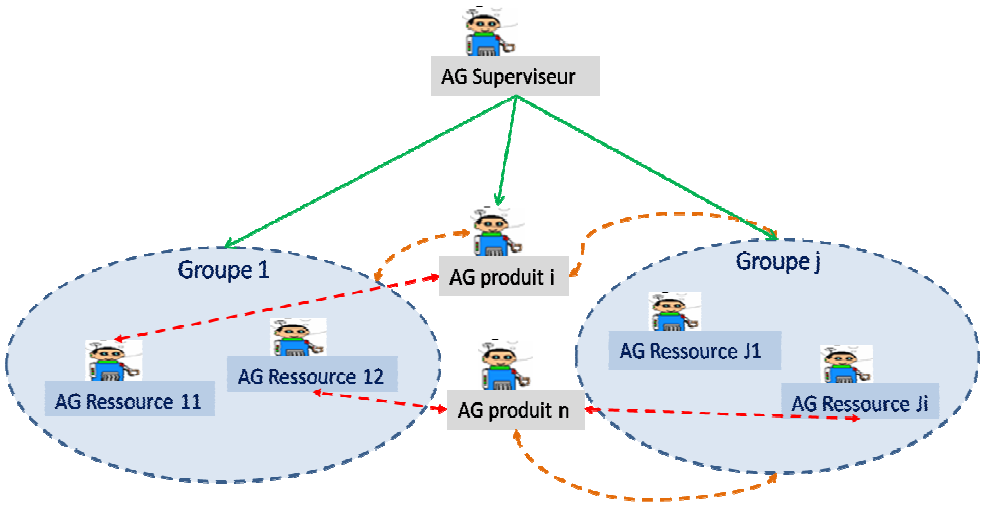


Figure 3.4. Structure générale du système

Cette architecture a été largement expérimentée dans des systèmes fortement complexes, citons les travaux de (Prabhu, 2003; Haruno and Kawato, 2006). Cette organisation garantit la rapidité de la réponse grâce à la connectivité très forte qui existe entre les différentes entités donc la réactivité du système. Une autre caractéristique très forte est l'autonomie des entités, cette autonomie permet au système d'être auto-organisé et adaptatif (Bousbia et Trentesaux, 2004). Nous partons ainsi du postulat qu'une organisation hétérarchique de notre système de pilotage peut garantir la réactivité et l'adaptabilité de ce dernier.

Dans ce qui suit, nous analyserons plus en détails l'architecture et le comportement de chaque agent, c'est-à-dire, l'aspect statique et dynamique des agents.

4 Architecture générique d'un agent du système de pilotage

Afin de développer chaque type d'agent nous commençons d'abord par élaborer une architecture de base que nous appellerons l'architecture des agents de substitution. Leurs rôles et les groupes seront succinctement décrits.

4.1 Les agents de substitution

Nous l'avons appelé ainsi, car chacun de ces agents de substitution est une reproduction abstraite des objets physiques du système de production. Comme nous sommes dans un contexte de production à flux, ces objets sont typiquement les bacs de stockage, les pompes, les produits finis, etc.

Ces agents coopèrent et interagissent pour contrôler le système de production à flux, leur principal but est de déterminer les séquences d'opérations de production et de maintenance en optimisant les temps et le coût d'emploi des ressources.

Ces agents sont modélisés selon le modèle de base de Querrec et al. (1997) (présenté dans le chapitre II) et l'architecture de base d'un agent Aaladin (Ferber et Gutknecht, 1998). L'architecture de base est ainsi composée des modules suivants (figure 3.5).

Module perception : c'est le module qui permet à l'agent de percevoir son environnement, ceci se fait par la lecture de certains paramètres indicateurs de l'état du système.

Module Action : Ce module permet à l'agent d'agir sur son environnement, il peut changer des paramètres de l'environnement, un agent par exemple peut changer l'état de la ressource dont il est responsable.

Module communication : C'est le module de liaison inter agents, il est responsable de la communication, de l'envoi et de la réception des messages, mais le traitement se fait par le module décisionnel.

Module décisionnel : C'est le cœur d'un agent, car ce module produit les intentions de l'agent. Suite à la réception d'un message ou l'observation d'un ou de plusieurs paramètres de l'environnement, ce module fait le traitement de ces informations et produit des décisions qui se traduisent par un ensemble d'actions.

Les agents de substitution sont des agents cognitifs (par opposition aux agents purement réactifs à comportement reflexe). En effet, les agents de substitution sont des agents intentionnels qui ont des buts fixés qu'ils tentent d'accomplir, ces agents sont dotés d'une connaissance et d'une capacité décisionnelle, exécutant le cycle : perception – délibération/décision – action.

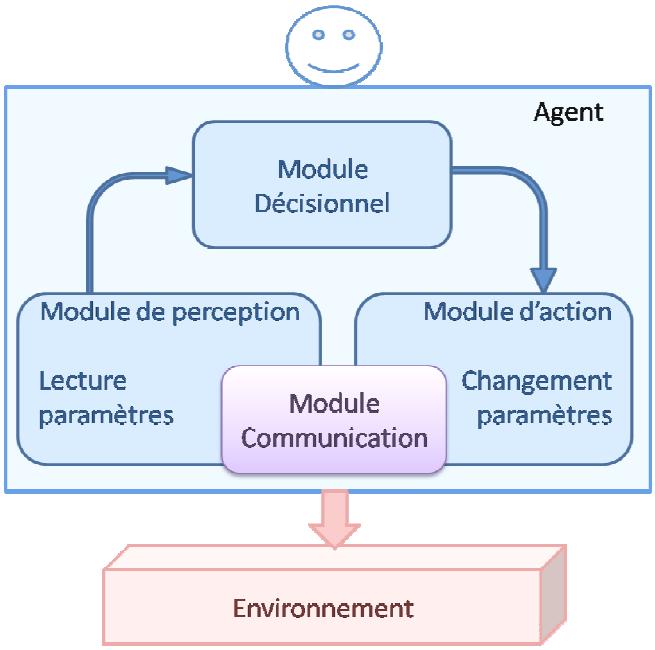


Figure 3.5. Architecture générique d'un agent de substitution

4.2 Le rôle des agents

Selon le modèle *Aalaadin*, le rôle est un attribut de l'agent, il représente son rôle et son identification au sein du système multi-agent ou plus précisément dans un groupe. Ces agents prennent pour identité les ressources ou produits qu'ils représentent. Par conséquent, le rôle sera les fonctionnalités des ressources ou produits qu'ils représentent.

Nous retrouvons : des agents machine et des agents produit dans un modèle général et des agents Bac et des agents Produit ou Huile pour le problème de la production à flux continu.

4.3 Le groupe

Le groupe est une sous-communauté d'agents si tout le système est la communauté. Le groupe permet de mieux gérer l'interaction inter agents, car lorsqu'il s'agit d'un « broadcast » (envoi généralisé d'une requête par exemple) on peut spécifier l'envoi à un groupe au lieu de l'envoyer à tous les agents du système, ce qui nous permet dans notre cas d'étude de définir des groupes par gamme opératoire. Par exemple, concernant le problème de la production à flux continu, nous pouvons définir un groupe par grade d'huile : SPO, SAE10, SAE30, BS.

5 Spécification des agents du système de pilotage

L'agent de substitution est l'agent de base sur lequel se base la conception de nos agents système. Dans la première section de ce chapitre nous avons défini deux types d'agents :

Les agents ressource et les agents produit, nous avons également défini et justifié l'agent superviseur. Dans ce qui suit, nous détaillons chacun de ces agents.

5.1 Les agents produit

5.1.1 Le module perception

Les agents produits représentent les produits finis que doit réaliser le système de production. Leur modèle doit pouvoir distinguer chaque tâche de production et permettre de déterminer pour chacune leur temps de début au plus tard, leur durée et la quantité demandée. En production à flux continu, une tâche de production est typiquement une tâche de vidange. La tâche est alors caractérisée par sa date de début, sa durée et la quantité d'huile à remplir ou à vider.

Il faut également déterminer les contraintes de précédence. Par exemple dans le cas de la production des huiles finies, l'ordre des tâches n'est pas important mais elles doivent être consécutives, sans temps mort entre les tâches.

L'agent produit perçoit l'ensemble des tâches qui constituent sa gamme opératoire et leur état ainsi que les messages des autres agents de son environnement.

Tâches de production (vidange)

Une tâche de production $\boldsymbol{Op_{ij}}$ est l'*ième* opération pour fabriquer un produit fini *j*, cette tâche correspond au transfert d'une huile de base HB d'un bac $\boldsymbol{R_k}$ vers le « manifold » avec une quantité $\boldsymbol{p_{ij}}$ sous réserve que le bac R_k dispose d'une quantité disponible suffisante.

La durée de chaque opération de vidange $\boldsymbol{a_{ijk}}$ est connue puisque le débit des pompes est connu .Sa date de début $\boldsymbol{t_{ij}}$ peu être toutefois quelconque.

Cependant, lorsque la fabrication $\boldsymbol{Op_{ij}}$ d'un produit a commencé, il ne faut en aucun cas qu'elle soit interrompue (tâches non préemptives). Toutefois, toutes les tâches $\boldsymbol{Op_{ij}}$, tel que $i=0..n$, peuvent être réalisées dans n'importe quel ordre.

5.1.2 Le module décisionnel

Ce type d'agent dispose donc d'une liste de tâches {$\boldsymbol{Op_{ij}}$ } à exécuter. Pour chaque tâche, il doit lancer une demande d'exécution au groupe d'agents susceptibles de la lui exécuter. Suite à cela, il reçoit une réponse des agents ressource, des réponses de refus et des propositions d'exécution. Le module décisionnel doit pouvoir analyser ces propositions, voir si elles sont valides par rapport à ses contraintes telles que les contraintes de précédences, faire une comparaison et élire la meilleure proposition pour conclure. Il est ainsi nécessaire d'établir un critère pour identifier la meilleure proposition. Ce critère devra tenir compte de la date de début proposée par la ressource et la valeur du contrat.

5.2 Les agents ressource

5.2.1 Le module perception

Les agents ressources reçoivent un plan d'entretien annuel avec un ensemble de tâches qui doivent être impérativement exécutées avec plus ou moins une flexibilité sur les dates d'exécution.

De ce plan, l'agent ressource doit dégager des informations qui concernent l'indisponibilité des outils de production. Ce type d'informations conditionne la capacité de l'agent à participer ou non à la négociation d'un contrat d'exécution de tâches. Le modèle de l'agent ressource devra alors pouvoir déterminer :

- La disponibilité de la ressource en termes de temps, après la réception d'une demande d'exécution,
- La capacité de stockage, sachant que les agents ressource sont des ressources de production qui sont de capacité limitée (exemple des bacs de stockage).

En plus de contraintes liées à l'état de la ressource, des contraintes d'utilisation doivent être prises en compte, par exemple : le taux cible d'utilisation d'une ressource, une pompe liée à un bac qui ne peut être utilisée à plus de 75% de sa capacité, etc.

En fonction de ces données et de ces contraintes, et face à une demande, l'agent ressource doit pouvoir prendre des décisions de participation aux négociations et aussi de déterminer ses offres.

5.2.2 Les tâches à considérer

Trois types de tâches sont donc à considérer : les tâches de production (émises par l'agent produit. Elles ont été présentées dans le paragraphe 5.1.1), les tâches de remplissage et enfin, les tâches de maintenance.

Tâche de remplissage (alimentation)

L'unité 3100 reçoit de l'unité en amont, unité de fabrication des huiles de base, un programme décadaire contenant les dates d'envoi des huiles de base avec leurs types et leurs quantités. L'unité 3100 doit alors se préparer pour les recevoir dans les bacs adéquats. Ce plan décadaire est alors constitué d'un ensemble de tâches de remplissage pour les bacs de réception.

Chaque ordre d'envoi correspond à une tâche d'alimentation ou de remplissage $\boldsymbol{Or_{lm}}$ qui est l'*lième* opération pour recevoir l'huile de base HB_m avec une quantité $\boldsymbol{p_{lm}}$, sur une durée connue $\boldsymbol{a_{lm}}$, les mêmes pompes utilisées pour la vidange sont utilisées pour le remplissage. La date de début $\boldsymbol{t_{lm}}$ est connue aussi, car elle est déterminée par l'unité en amont. L'unité 3100 doit alors déterminer quel serait le bac $\boldsymbol{R_o}$ qui pourrait recevoir cette quantité notée $\boldsymbol{a_{lmo}}$.

Tâche de maintenance

Les équipements de l'unité subissent un entretien régulier. Ces tâches de maintenance occupent les ressources de production et les rendent indisponibles sur une période donnée. On doit distinguer :

- Les tâches de maintenance préventive
  $\boldsymbol{Omp_{ij}}$ Tâche de maintenance préventive concernant la ressource $\boldsymbol{R_j}$ sur une durée $\boldsymbol{amp_{ij}}$ avec une date de début au plus tôt $\boldsymbol{r_i}$, et une date de début au plus tard $\boldsymbol{fm_{ij}}$.
- Les tâches de maintenance corrective
  $\boldsymbol{Omc_{ij}}$ tâche de maintenance corrective qui surviennent aléatoirement et qui doivent être traitées en urgence sur la ressource $\boldsymbol{R_j}$ pendant une durée $\boldsymbol{amc_{ij}}$ et une date de début au plus tôt $\boldsymbol{r_{ij}}$.

5.2.3 Le module décisionnel

A chaque fois qu'un agent ressource reçoit une requête d'exécution, il doit analyser son état puis faire le choix, c'est-à-dire prendre une décision, de proposer d'exécuter soit une tâche de maintenance soit de production (si elles se présentent en même temps), puis, s'il choisit de faire une proposition d'exécution il doit calculer ses paramètres d'exécution pour formuler complètement sa réponse à l'appel d'offre (figure 3.6).

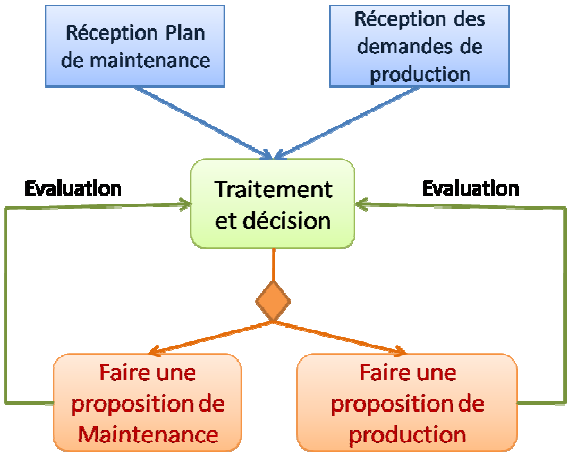


Figure 3.6. Diagramme de base de l'agent ressource

Ce processus décisionnel doit pouvoir s'adapter aux changements de l'environnement, que ce soit après modification par les autres agents ou après réception de requêtes imprévues.

Le processus décisionnel se doit aussi d'être continuellement en amélioration. Ceci peut se faire en utilisant des techniques d'apprentissage.

Suite à l'analyse réalisée dans le chapitre précédent, nous avons opté pour l'apprentissage par renforcement (Watkins, 1989).C'est en effet une technique réactive qui permet d'exploiter ses résultats dans toutes ses phases tout en permettant d'améliorer continuellement les résultats. Elle sera décrite en détail dans le prochain chapitre.

5.3 Spécification du rôle de l'agent superviseur/observateur

Cet agent a une vue globale sur le système, il récolte des informations des différents agents pour évaluer leurs performances.

Des agents produit, il reçoit le temps global de production afin d'analyser le temps global d'achèvement du projet *Cmax*. Les agents ressource l'informent de leur taux d'occupation et des arrêts de maintenance ou d'attente.

Les indicateurs de performances générés par cet agent permettent au système de pilotage de suivre en temps réel l'évolution des performances.

6 Spécification du protocole d'interaction entre agents

L'interaction au sein d'un système multi-agent doit respecter un certain formalisme pour qu'elle puisse être contrôlée et permettre l'émergence de solutions au problème abordé. Ce formalisme est le protocole d'interaction.

6.1 Le protocole d'interaction

Un protocole d'interaction permet aux agents d'échanger et de comprendre les messages de manière structurée.

Le protocole qui devra être développé pour notre modèle est inspiré du protocole Contract Net Protocole (Smith, 1980) de la norme FIPA. Des primitives de base ont été utilisées :

Requête : Il s'agit de la demande qu'établit un agent produit lorsqu'il veut s'offrir un service qu'un autre agent peut lui rendre (ex : se faire exécuter une tâche).

Acceptation : Lorsqu'un agent qui a reçu une requête se voit en mesure de rendre ce service, il envoie à l'agent qui a émis la requête un message d'acceptation.

Refus : Lorsqu'un agent qui a reçu une requête ne se voit pas en mesure de rendre ce service, il envoie à l'agent qui a émis la requête un message de refus.

Confirmation : Lorsque l'agent demandeur reçoit un ou plusieurs messages d'acceptation, il envoie à l'un de ces correspondants, qu'il choisit selon un certain critère, un message de confirmation pour lui signaler qu'il accepte qu'il lui rende ce service.

Libération : et envoie aux autres correspondant un message pour les libérer de cette requête de service.

Fin : message envoyé par un agent qui a exécuté un service à l'agent demandeur pour l'informer de la fin d'exécution.

Annulation : suite à un problème au niveau de la ressource un contrat peut être annulé.

Bien sûr ces primitives sont génériques, en fonction du problème elles sont spécifiées et elles sont envoyées augmentées d'information sur le contexte du contrat.

Dans la figure suivante (3.7), une présentation d'un cas général d'utilisation des ces primitives est donnée :

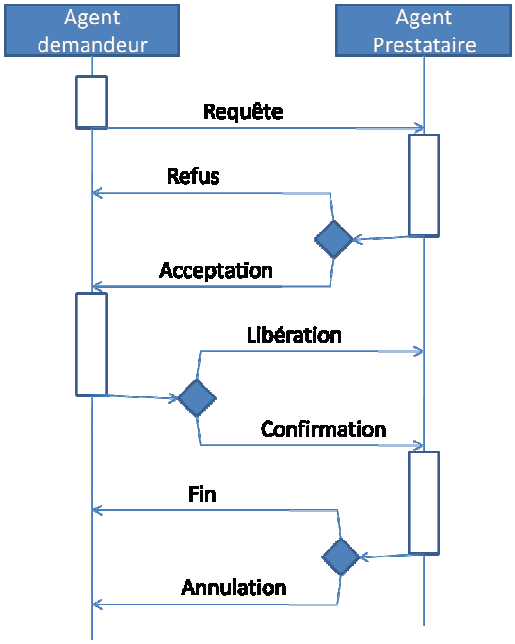


Figure 3.7. Séquence de communication en utilisant les primitives de base

Dans notre étude, le protocole est ainsi le suivant : lorsqu'un agent produit veut se faire exécuter une tâche, il lance une requête au groupe d'agents ressources susceptibles de pouvoir exécuter cette tâche. Le message est de type **Requête** avec les propriétés de la tâche (durée, quantité…).

Lorsqu'un agent ressource reçoit un message requête, il analyse les données de la requête et consulte son propre état pour décider de refuser la requête ou faire une proposition d'exécution. S'il refuse, alors un message de **Refus** est envoyé à l'agent produit, sinon, un message d'**Acceptation** avec les modalités du contrat (coût d'exécution, durée d'attente…) est envoyé.

L'agent produit, émetteur de la requête, peut ne recevoir que des refus, dans ce cas là il remet sa requête. S'il reçoit des messages d'acceptation, il lance alors une analyse des propositions par rapport aux modalités du contrat : coûts d'exécution proposés, durée d'attente requise…

Une comparaison est ensuite faite et un message de **Confirmation** est envoyé à l'agent ressource dont la proposition a été la plus intéressante et un message de **Libération** est envoyé aux autres.

Dès que le contrat est établi, l'exécution de la tâche est alors lancée et à la fin de cette dernière, l'agent ressource envoie à l'agent produit un message de **Fin** pour l'informer de la fin de la tâche et pour libérer les deux contractuels.

La mise en œuvre de ce protocole nécessite l'utilisation de langages de communication, plusieurs types de protocoles et langages de communication multi-agents ont été développés. Nous présenterons deux protocoles et langages de communication des plus connus et des plus utilisés.

6.2 Le protocole et langage de communication

Un protocole de communication permet aux agents d'un système multi-agent d'échanger des messages entre eux et surtout de les comprendre. Il faudra alors disposer d'un langage commun pour pouvoir coopérer pour la résolution d'un problème et d'un langage primitives connues par chaque agent.

Dans le modèle Aalaadin la communication inter agents est directe. Un agent peut choisir de communiquer avec tel agent qui dispose de propriétés (par exemple, de tel rôle dans tel groupe), ou alors d'utiliser directement une adresse pour envoyer un message.

Deux langages de communication sont définis dans le modèle Aalaadin : ACLMessages et KQMLMessages. Nous présentons rapidement ces deux langages.

6.2.1 ACL Messages

Le concept ACL (ACL pour Agent Communication Language) a été créé par la Foundation for Intelligent Physical Agents (FIPA). Ce dernier standard repose en particulier sur la théorie des actes de langage (Theory of Speech Acts) (Austin, 1962, Searle, 1969).

Les agents dotés du mécanisme ACL peuvent communiquer des propositions, des règles et des actions et pas uniquement des objets qui n'ont pas de sémantique associée. La communication peut être structurée dans des conversations en utilisant des identifiants de conversation et des descriptions de réponses (en réponse à, réponse avec…). Il est alors facile d'utiliser ce langage dans un protocole d'interaction. La figure suivante (3.8) montre la structure d'une trame d'un message ACL.

Type message	*Envoyeur*	*Récepteur*	*Contenu*	*Num réponse*	*Id conversation*
Inform	Sender	Receiver	Content (Prix, prod1, 150)	In reply to	conversation-id

Figure 3.8. Structure d'un Message ACL

6.2.2 KQML Messages

Knowledge Query and Manipulation Language est plus ancien que le ACL. KQML est un langage de communication de haut niveau orienté messages et un protocole pour l'échange des informations qui est indépendant de la syntaxe du contenu (KIF, SQL, Prolog,…) et de l'ontologie de l'application (ontologie : une spécification des objets, concepts et relations dans un domaine particulier afin de le décrire). KQML sépare entre la sémantique du protocole de communication (indépendante de domaine) et la sémantique du message (dépendante de domaine). La figure 3.9 montre une trame de langage KQML qui définit clairement le langage (de programmation cette fois) utilisé et le type de connaissance qu'il transporte.

Type message	***Envoyeur***	***Récepteur***	***type réponse***	***Langage***	***Connaissance***	***Contenu***
Ask-one	Sender	Receiver	Reply with	Language	Ontology	Content (Prix, prod1, 150)

Figure 3.9. Structure d'un Message KQML

Nous avons choisi d'utiliser l'approche de type ACL Messages car ces messages sont plus faciles à manipuler, intégrant sémantique et contenu. En outre, cette approche répond aux normes FIPA (FIPA, SC00061) et est régulièrement utilisé pour une communication en Contract Net Protocole.

7 Conclusion

Dans ce chapitre, nous avons spécifié notre système de pilotage à partir d'une approche multi-agent basée sur le formalisme Aalaadin.

Plus précisément, notre système de pilotage se composera de trois types d'agents : des agents ressource, qui représentent les ressources de production, leur états et leurs capacités. Les agents produits, qui représentent les produits finis qui doivent être produits par ce système de production, les différentes tâches qu'ils doivent subir et leurs contraintes. Un troisième type d'agent a été introduit dans notre modèle, c'est l'agent superviseur/observateur, cet agent a une vue globale sur le système, de ce fait, il intègre des critères de performances pour pouvoir suivre l'évolution des performances du système de pilotage.

Nous avons également spécifié le rôle et le comportement de chaque agent au sein du groupe auquel il appartient et au sein de tout le système, nous avons défini les types de données que les agents perçoivent et les traitements qu'ils sont amenés à faire. Nous avons aussi spécifié les niveaux d'interaction que les agents entretiennent entre eux et les moyens de communications qu'ils utiliseront tout en définissant le protocole d'interaction et de communication qu'ils respecteront.

Nous avons identifié aussi pour chaque type d'agents un processus décisionnel. Nous avons dans ce chapitre supposé que ce processus contenait des mécanismes d'apprentissage sans les avoir pour l'instant décrit. Ceci constitue l'objet du prochain chapitre.

Chapitre 4 : Modélisation Par processus décisionnel markovien et résolution par apprentissage par renforcement multi-agents et multi-objectifs

1 Introduction

Dans ce chapitre nous présenterons le modèle de connaissance des agents tel que spécifié dans le chapitre précédent. Ce modèle est basé sur un processus décisionnel markovien (Markovien Decisional Process MDP). Nous commencerons par présenter les concepts de base d'une modélisation MDP, puis MDP-décentralisé, et enfin l'élaboration d'un MDP local pour la prise de décision au niveau des agents du système. Nous présenterons alors l'apprentissage par renforcement qui permet d'apprendre aux agents la politique d'action la plus efficace pour le modèle MDP défini. Nous ferons référence à certains outils mathématiques utiles comme les jeux de Markov pour décrire des modèles de décision et d'apprentissage dans le cas où l'apprentissage d'un agent est influencé par l'observation du comportement des autres agents. Nous présenterons par la suite des outils d'apprentissage multi-objectifs pour un agent qui présente deux objectifs à atteindre (tel que dans notre cas : optimisation de la production et de la politique de maintenance). Nous commencerons tout d'abord par placer l'apprentissage par renforcement parmi l'ensemble des techniques d'apprentissage automatique.

2 Apprentissage par renforcement une technique d'apprentissage automatique pour une entité abstraite ou physique

L'apprentissage automatique pour des entités abstraites ou physiques consiste en l'introduction d'algorithmes capables d'apprendre et d'améliorer leurs connaissances par rapport aux expériences déjà faites.

L'apprentissage par renforcement est une variante des techniques d'apprentissage artificielles, ces techniques d'apprentissage sont classées selon trois modes :

2.1 Le mode d'apprentissage supervisé

Le but dans un apprentissage supervisé est de trouver une description générale et caractéristique décrivant une classe sans avoir à énumérer tous les exemples de cette classe. Il faut découvrir ce que les exemples ont en commun et ce qui peut donc être induit comme étant la description de la classe.

L'idéal d'un algorithme d'apprentissage est d'arriver à une description très réduite, simple avec un minimum de critères pour identifier exactement un concept et de pouvoir affirmer qu'un exemple répond ou ne répond pas à ce concept avec le maximum d'exactitude.

Il existe une grande variété de méthodes, utilisant ou non des heuristiques. Comme principales approches, nous citons l'Analyse Discriminante, le modèle de Régression, l'Espace de Versions (Rabased et Loudcher, 1996) les algorithmes génétiques, les réseaux de neurones, les graphes d'induction : SIPINA (Zighed et al. 1992), ...

Une entité qui a appris en apprentissage supervisé est capable de faire exactement ce qu'il faut, car elle a une base de connaissance complète qui lui permet de prendre exactement la bonne décision, correspondant à un modèle d'inférence (SI *fait1* ET *fait2* ALORS *fait3*).

2.2 Le mode non-supervisé (ou auto-organisationnel)

Dans un apprentissage non-supervisé, au début de l'apprentissage, il n'y a pas de classes prédéfinies et le but est d'effectuer les meilleurs regroupements possibles entre les objets et d'en trouver une description explicative. L'apprentissage non supervisé, appelé aussi apprentissage par observation ou par découverte, consiste en l'élaboration de nouveaux concepts ou de nouvelles théories caractérisant les objets donnés (Mjolsness et Decoste, 2001).

Dans ce cas, l'apprentissage est basé sur des probabilités. On essaye d'établir des catégories, en attribuant et en optimisant une valeur de qualité, aux catégories reconnues.

Ici l'entité n'a aucun guide, elle fait des propositions et les vérifie selon certains critères.

Il existe un troisième type, entre les deux, où l'entité fait des tests et reçoit des évaluations de ses tests, c'est l'apprentissage par renforcement ou l'apprentissage semi-supervisé ou par l'application.

2.3 Le renforcement (Apprentissage par l'application)

Ce type d'apprentissage renvoie à la première partie de la définition de H. Simon (Simon, 1983) où l'apprentissage permet à un système de mieux accomplir une tâche lorsqu'elle se répète une deuxième fois.

Nous supposons disposer d'un système déjà capable de résoudre un problème. Ce système possède des heuristiques concernant l'utilisation de règles ou d'opérateurs de résolution du problème et propose une solution dite initiale et supposée optimale. C'est à partir de cette solution initiale que l'apprentissage commence en faisant ré-exécuter au système la tâche qu'il vient de réussir.

Puisque notre choix s'est porté sur ce type d'apprentissage, nous présentons en détail dans la partie suivante les fondements mathématiques de l'apprentissage par renforcement.

3 Modélisation Processus Décisionnel Markovien décentralisé sur les agents

Dans le chapitre précédent (3), nous avons vu qu'un agent du système est constitué de modules : module de perception, module d'action, module de communication et le module décisionnel. Ce module décisionnel est le cœur d'un agent, il doit pouvoir lui permettre de prendre des décisions efficaces dans les plus courts délais avec la possibilité de les améliorer. Pour ce faire, nous avons recours à des Processus Décisionnels de Markov (MDP). Chaque agent construit son propre MDP en ne considérant que les tâches qu'il doit réaliser.

Le système multi-agent que nous développons est un système décentralisé à base de MDP, nous le désignerons par MDP-DEC (MDP Decentralized). La figure 4.1 représente de façon schématique le fonctionnement du système. Dans cette section, nous présenterons les MDPs ainsi que les MDP-DEC. Nous décrirons ensuite plus en détail notre modèle décisionnel et comment intégrer des mécanismes d'apprentissage par renforcement dans ce modèle.

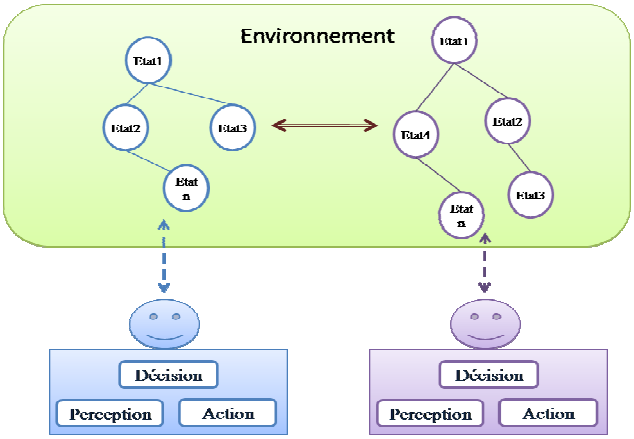


Figure 4.1. Représentation schématique du fonctionnement d'un MDP décentralisé

3.1 Processus décisionnel Markovien

Les Processus Décisionnels de Markov (MDPs) sont utilisés pour résoudre des problèmes de décision séquentiels, c'est-à-dire pour lesquels il est nécessaire, à chaque étape du problème, de décider quelle action réaliser. Les MDPs permettent, à partir d'un état initial, d'effets sur les actions et d'objectifs, d'obtenir une politique d'action dont l'exécution parviendra au mieux à réaliser les objectifs de l'agent. Les objectifs sont définis à l'aide de récompenses. Une politique d'action fait correspondre à chaque état une action. Lorsque l'agent est dans un état *s*, dire qu'il suit une politique π signifie qu'il choisit d'effectuer l'action associée à *s* dans π.

Définition formelle

Un MDP est défini par un quadruplet < S,A,T,R > *:*
– S *un ensemble fini d'états*
– A *un ensemble fini d'actions stochastiques possibles*
– T = S x A x S – [0 ;1] *un modèle de transition correspondant à une distribution de probabilité associant à chaque triplet* (s, a, s') *une probabilité : la probabilité de passer de* s *à* s' *en réalisant l'action* a.
– R= S x A x S – $\mathbb{R}$ *une fonction de récompense permettant de calculer l'utilité espérée*

(Putterman, 1994)

3.1.1 L'utilité d'un état / Fonction 'valeur d'état'

Elle représente l'utilité qu'apportera le fait d'être sur l'état *s* dans une politique π (équation 1). Autrement dit, c'est la somme des récompenses futures à partir de l'état *s* en appliquant la politique π à l'instant *t* (Bellman, 1957).

$$V_{t+1}^{\pi}(s) = R(s) + \gamma \sum_{s'} P_{\pi(s)}(s,s') V_t^{\pi}(s') \qquad (1)$$

Sachant que $\gamma \in [0,1]$ est un coefficient de compensation qui est proche de 1 lorsqu'on a fait suffisamment d'expériences afin de donner plus de poids aux récompenses, on l'appelle aussi un taux de diffusion vers les expériences passées.

Une autre fonction valeur d'état a été élaborée telle qu'une situation n'est pas définie par l'état mais par la paire (état *s*, action *a*), on n'évalue pas seulement l'état mais le fait de choisir l'action *a* dans l'état *s* (Watkins, 1989). Elle est définie par :

$$Q_{t+1}^{\pi}(s,a) = R(s) + \gamma \sum_{s'} P_a(s,s') V_t^{\pi}(s') \qquad (2)$$

3.1.2 La notion de politique

Dans le cadre des MDP, la politique π: S → A est une suite qui indique pour chaque état quelle est l'action à effectuer (voir exemple figure 4.2). Il s'agit là d'une politique déterministe, dans laquelle il n'y a pas d'ambiguïté sur l'action à effectuer. On note *Π* l'ensemble des politiques déterministes.

Le but d'un MDP est d'évaluer les politiques. La politique optimale est celle dont l'utilité espérée est maximum. Elle fournit aux agents un comportement optimal, c'est-à-dire que dans chaque état, l'action choisie est celle susceptible d'augmenter le plus possible le gain des agents.

$$\pi(s) = \arg\max_{a} \sum_{s'} P_a(s,s') V(s')$$

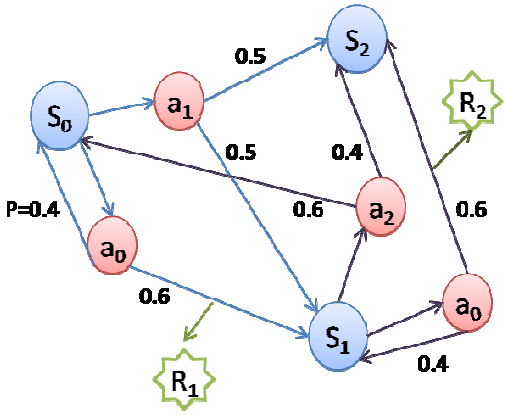


Figure 4.2. Exemple d'un MDP à 3 états et 3 actions

3.2 Problématique des MDPs et résolution par l'apprentissage par renforcement

Dans les MDP, l'évolution du système est supposée correspondre à un processus markovien. Autrement dit, le système suit une succession d'états distincts dans le temps et ceci en fonction des *probabilités de transitions*.

L'hypothèse de Markov consiste à dire que les probabilités de transitions ne dépendent que de l'état précédent, ce qui permet de ne considérer que l'état courant et l'état suivant.

Le modèle MDP ainsi présenté est supposé stable dans le temps, c'est-à-dire que les composants du quadruplet sont supposés *invariants*. Il n'est donc pas applicable tel quel pour un système qui évolue, par exemple pour modéliser un système qui s'exécute face à un autre système évolutif. Lorsqu'un modèle n'est pas déterministe, c'est-à-dire dont on ne peut pas prédire l'état suivant, ou un système fortement perturbé il n'est pas possible de donner un modèle de transition, comme c'est le cas pour le système de production.

La question qui se pose est alors: comment faire pour identifier des stratégies ou politiques ? et comment déterminer la politique optimale ?

L'apprentissage par renforcement est une technique qui a été élaborée justement pour répondre à ces questions, c'est-à-dire identifier une politique optimale sur n'importe quel MDP (avec ou sans modèle de transition)

4 Apprentissage par renforcement

4.1 Apprentissage par renforcement pour un agent

Définition *L'apprentissage par renforcement désigne toute méthode adaptative permettant de résoudre un problème de décision séquentielle* (Sutton et Barto, 1998).

Le terme "adaptatif" signifie qu'on part d'une solution inefficace, qui est améliorée progressivement en fonction de l'expérience de l'agent (ou des agents).

L'apprentissage par renforcement repose sur le principe énoncé par Bellman (1957) et repris pas Sutton (1998) qui dit que : une suite optimale de commandes (décisions) telle que, quelle que soit l'étape, les commandes suivantes doivent constituer une suite optimale de décisions pour la suite du problème.

Il existe trois familles de méthodes qui construisent cette politique optimale :

- Méthodes de Programmation dynamique (DP)
- Méthode Monte-Carlo (MC)
- Méthodes des différences temporelles (TD)

Les méthodes Programmation dynamique et Monte-Carlo sont présentées dans l'annexe 6.

La méthode TD a donné lieu à plusieurs algorithme tels que : SARSA, Q-Learning, Critique-acteur et TD(λ).

Ces algorithmes se caractérisent par le type de fonction valeur estimée, et par les techniques d'évaluation et d'amélioration des stratégies. Deux familles se distinguent (Zennir, 2004):

Les méthodes « On-policy » : elles évaluent et améliorent la stratégie utilisée pour la prise de décision, exemple l'algorithme Sarsa.

Les méthodes « off-policy » : la politique de prise de décisions est indépendante de la politique à améliorer. Ceci permet de faire une meilleure exploration car la politique à optimiser n'influence pas la prise de décision pendant le processus d'apprentissage, exemple, l'algorithme Q-learning.

Nous présenterons un algorithme de chaque famille, ces deux algorithmes constitueront la base de notre étude.

4.1.1 L'algorithme SARSA

SARSA est un algorithme d'apprentissage par renforcement « on-policy » qui évalue et améliore la stratégie utilisée pour la prise de décision. Une expérience consiste non seulement à choisir l'action *a* pour l'état *s* mais aussi le choix de l'action *a'* qui sera choisie pour *s'* suivant la stratégie d'exploration choisie. La fonction d'état-action (3) est une modification de l'expression (2) en considérant $Q^{\pi}(s,a)$:

$$Q_{t+1}(s,a) = Q_t(S,a) + \alpha\left[r_{t+1} + \gamma.Q_t(s',a') - Q_t(s,a)\right] \qquad (3)$$

L'information nécessaire pour réaliser une telle mise à jour lorsque l'agent réalise une transition est le quintuplé (s_n; a_n; r_n; s_{n+1}; a_{n+1}), d'où découle le nom de l'algorithme.

Il existe alors une dépendance entre la politique à construire et la politique de prise de décision pendant l'exploration (le terme exploration va être défini par la suite), cette dépendance a compliqué la mise au point de preuves de convergences pour ces algorithmes, ce qui explique que de telles preuves de convergence soient apparues beaucoup plus tard après leur invention (Singh, 2000) ce qui a été différent pour les algorithmes *off-policy* tel que Q-Learning.

SARSA converge vers la politique optimale (la fonction Q_n converge presque sûrement vers Q*) si :

- Toutes les paires (état, action) sont visitées une infinité de fois
- La stratégie de choix de l'action tend vers une stratégie gloutonne.

  Stratégie ε-gloutonne : elle consiste à choisir l'action gloutonne avec une probabilité ε et à choisir une action au hasard avec une probabilité 1 – ε. Donc il suffit de prendre ε=1 /t.

4.1.2 L'algorithme Q-Learning

Le Q-Learning est considéré comme une simplification du SARSA car il n'est plus nécessaire de déterminer, un pas de temps à l'avance, quelle sera l'action réalisée au pas de temps suivant. Son équation de mise à jour est la suivante :

$$Q_{t+1}(s,a) = Q_t(S,a) + \alpha\left[r_{t+1} + \gamma.\max_{a'} Q_t(s',a') - Q_t(s,a)\right] \qquad (4)$$

La différence entre SARSA et Q-Learning se trouve au niveau des termes $Q_t(s',a')$ et $\max_{a'} Q_t(s',a')$ respectivement. Cette différence réside dans le fait que l'algorithme SARSA effectue les mises à jour en fonction des actions choisies effectivement alors que l'algorithme Q-Learning effectue les mises à jour en fonction des actions optimales mêmes si ce ne sont pas ces actions optimales que l'agent réalise en fin de compte, ce qui est plus simple. Cette simplicité a fait la réputation du Q-Learning, Il s'agit sans doute de l'algorithme d'apprentissage par renforcement le plus connu et le plus utilisé en raison des preuves formelles de convergence qui ont accompagné sa publication dans Watkins et Dayan (1992) :

La convergence de cet algorithme est établie (la fonction Q_n converge presque sûrement vers Q*) sous les hypothèses suivantes :

- finitude de S et A,
- chaque couple (s, a) est visité un nombre infini de fois,
- $\sum_n \alpha_n(s,a) = \infty$ et $\sum_n \alpha_n^2(s,a) < \infty$,
- $\gamma < 1$ ou si $\gamma = 1$
- pour toute politique il existe un état absorbant de récompense nulle.

Rappelons que cette convergence presque sûre signifie que $\forall s,a$ la suite $Q_n(s,a)$ converge vers $Q^*(s,a)$ avec une probabilité égale à 1. En pratique, la suite $\alpha_n(s,a)$ est souvent définie comme $\alpha_n(s,a) = \frac{1}{n_{sa}}$ tel que α est proportionnel au nombre de visites de la paire (s,a).

L'algorithme du Q-Learning et SARSA (avec la différence sur la fonction de mise-à-jour de la valeur d'état) est (figure 4.3):

```
Initialisation arbitraire Q(s,a)
Répéter pour (chaque épisode):
Choisir un point de départ (état de départ)
      Répéter pour chaque transition :
            a ← action donnée pour la quelle Q(s,a) est
            maximale
            Exécuter a, observer la récompense r et le nouvel
            état s'
            Misa-à-jour Q_{t+1}(s,a) (expression 3/4)
            S← s'
      Jusqu'à l'état s terminal
```

Figure 4.3. Algorithme du Q-learning/SARSA (Watkins 1989;Sutton 1998)

En principe, il faut expérimenter ou explorer aléatoirement l'environnement suffisamment de fois pendant un grand nombre d'itérations pour que l'algorithme d'apprentissage par renforcement puisse converger vers la fonction Q optimal et seulement ensuite il est possible d'utiliser la politique optimal déduite π^*. Ce dilemme exploration – exploitation est présenté dans l'annexe 6.

4.2 Apprentissage par renforcement et système multi-agents

Dans un apprentissage par renforcement décentralisé, plusieurs questions se posent :

- Comment décomposer un objectif global en plusieurs objectifs locaux ?
- Comment coordonner l'ensemble des agents pour atteindre l'objectif global ?
- Comment décentraliser la fonction récompense ?
- Comment partager la connaissance entre plusieurs agents ?

Pour répondre à ces questions, nous commencerons par présenter l'architecture du système multi-agents, nous présenterons ensuite le Q-multiagent

4.2.1 Architecture du système et d'apprentissage et le niveau de coopération

Dans le chapitre précédent (section 3) nous avons pu déterminer l'architecture globale de notre système de pilotage et le modèle organisationnel des agents. Nous avons pu montrer que l'organisation *Hétérarchique* permet aux agents d'être en étroite relation avec des liens de communication et de contrôle très présents (simplifiée dans Figure 4.4). Sur cette architecture, l'apprentissage peut être soit centralisé soit décentralisé.

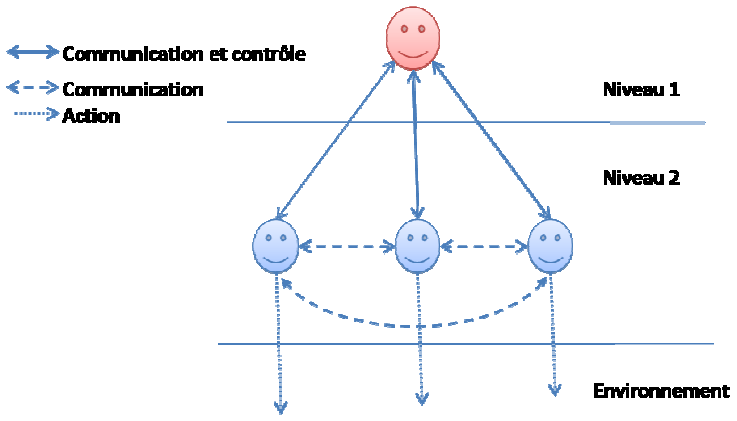


Figure 4.4. Organisation hétérarchique au sens large des agents

Apprentissage centralisé

Dans ce mode, la mémoire d'apprentissage est centralisée, la fonction de sélection et de mise-à-jour est unique et implémentée au niveau d'un agent de haut niveau. Cet agent perçoit l'environnement (les états), le signal de renforcement et dicte aux agents les actions à effectuer (voir figure 4.5).

Cette architecture d'apprentissage a l'avantage de libérer les mécanismes d'apprentissage des agents apprenant.

Les agents peuvent alors développer des stratégies d'exploration/ exploitation différentes d'une même mémoire d'apprentissage. Ce choix peut être exigé par des contraintes d'application, par exemple, cette architecture a été utilisée par Johannet et Sarda (1995) pour la commande de la marche d'un robot hexapode.

On peut reprocher à cette architecture comme à toute architecture centralisée son intolérance aux fautes car si l'agent superviseur tombe en panne c'est tout le système qui s'arrête.

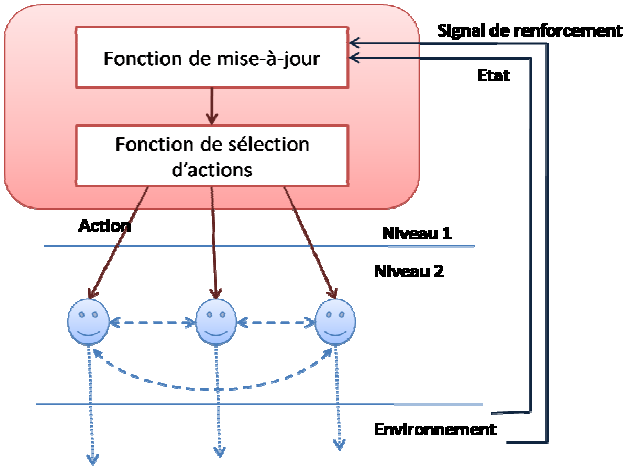


Figure 4.5. Apprentissage par renforcement centralisé

Apprentissage décentralisé

Au début des années 90s, des chercheurs ont commencé à penser à une architecture décentralisée de l'apprentissage par renforcement (voire figure 4.6), comme par exemple Beer et al. (1991) qui ont proposé d'utiliser des contrôleurs sur les agents qui prennent des décisions pour réaliser des sous-tâches où le but est de satisfaire un objectif global.

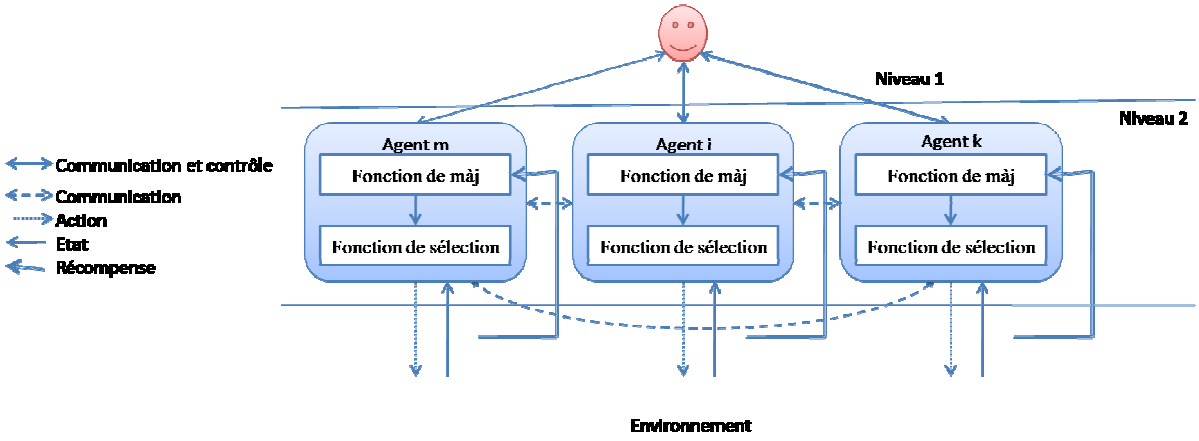


Figure 4.6. Apprentissage par renforcement décentralisé

Cette architecture permet aux agents d'être indépendants dans leur apprentissage mais lorsqu'ils participent ensemble à la réalisation d'une tâche, des problématiques de coopération peuvent apparaitre.

4.2.2 Avantages et inconvénients de l'apprentissage par renforcement décentralisé

Les avantages d'une architecture décentralisée d'apprentissage par renforcement sont :

- Les avantages d'une architecture décentralisée
- La mémoire d'apprentissage est optimale puisqu'elle est distribuée entre les différents agents
- Exploitation des avantages de l'interaction directe et indirecte (par modifications sur l'environnement) entre les agents
- Parallélisassions possibles des processus locaux d'apprentissage

Néanmoins, des limites sont possibles dont il faut prendre considération :

- La difficulté de déterminer le niveau de coopération et les moyens de communication.
- La difficulté de mettre en place des critiques locales pour satisfaire l'objectif global.
- Le système est non stationnaire du point de vue d'un agent, car l'agent subit parfois, la conséquence des actions des autres agents.

Dans ce qui suit nous présenterons quelques moyens pour palier ces inconvénients.

4.2.3 Coopération et apprentissage décentralisé

La distribution d'un problème et sa résolution donne la possibilité d'avoir des mécanismes de résolution simples à un niveau local. Chaque agent contribue à la résolution du problème commun en prenant en compte son environnement et l'influence qu'il peut avoir dessus.

Selon les facultés de représentation de l'environnement, du raisonnement et d'interaction entre les agents, plusieurs niveaux de coopération pour la résolution et l'apprentissage peuvent être distingués (Zennir, 2004):

- Niveau 1 : L'agent apprend à partir de ses observations de l'environnement sans interaction avec les autres agents, si ce n'est une interaction indirecte en modifiant leur environnement.
- Niveau 2 : L'apprentissage est influencé par l'observation du comportement des autres agents, ce mode nécessite une modélisation du comportement des autres agents
- Niveau 3 : Les agents peuvent échanger des requêtes, demander des informations pour assurer la coordination
- Niveau 4 : Les agents peuvent se répartir des tâches par délégation ou division du travail. C'est un niveau très élevé de la coopération qui nécessite une maturité des agents.

Dans notre système, et comme vu dans le chapitre précédent, nous voulons que nos agents acquièrent de la connaissance en apprennent à mieux agir chacun à son niveau, mais cela ne les empêche pas de communiquer et d'échanger des informations. De ce fait nous estimons que le niveau 3 est le niveau qui correspondant le mieux au modèle que nous mettons en œuvre.

Dans ce qui suit nous représenterons les mécanismes de contrôle du Q-Learning/SARSA associé à ce niveau.

4.3 L'apprentissage par renforcement multi-objectifs

Un agent apprenant peut avoir plusieurs objectifs à atteindre, l'agent devra alors déterminer une politique qui lui permette d'atteindre l'ensemble de ses objectifs, plus ou moins équitablement.

A la fin des années 90, Humphrys (1996) et Karlsson (1997) ont développé des méthodes d'apprentissage par renforcement multi-objectifs.

Lorsque l'environnement d'apprentissage est grand le temps d'exploration peut être long. Humphrys et Karlsson ont pensé pouvoir minimiser ce temps en décomposant l'objectif global en sous-objectifs et décomposer les tâches en sous-tâches indépendantes. Par exemple, un « robot gibier » qui apprend à vivre dans une jungle a besoin de survivre et manger, ce robot peut décomposer son objectif d'apprentissage en : apprendre à s'échapper des prédateurs et à chercher sa nourriture.

4.3.1 Définition des modules

En partant de ce principe, une architecture modulaire de l'apprentissage a été développée par Humphrys (1996). Des Sous-modules sont définis pour chaque sous-objectif. Chaque module perçoit les données qui le concernent et émet des actions pour atteindre son objectif.

Pour chaque module *i*, trois fonctions associées aux sous-objectifs avec les données qui les concernent, doivent être définies :

- m_i la fonction de sélection extrait de la description d'état les données qui correspondent à ce module. Autrement dit, définit un nouvel espace d'état extrait de l'espace d'état original.
- R_i la fonction récompense ne donne d'évaluation que sur l'accomplissement de ce sous-objectif indépendamment des autres objectifs
- Q_i la fonction d'utilité d'état évalue les paires (sous-état, action)

La figure 4.7 montre comment ces fonctions agissent, chaque module évalue sa fonction-Q. Ces évaluations sont utilisées par l'arbitre qui détermine quelle action exécutera l'agent.

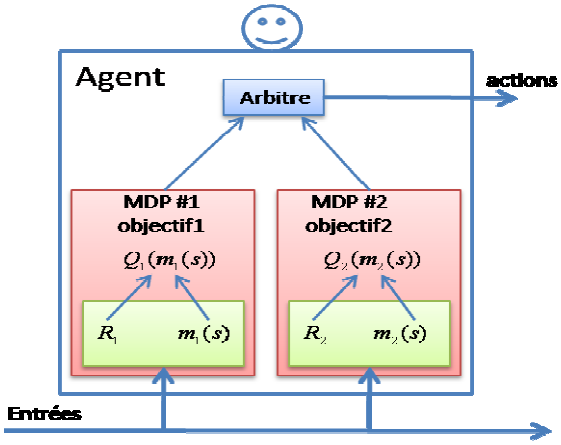


Figure 4.7. Architecture modulaire pour l'apprentissage multi-objectif

4.3.2 La fonction de sélection

L'ensemble des états que l'agent perçoit sont S= {s1,s2,…,sn} mais chaque module ne reçoit que les données qui le concernent. Reprenant l'exemple du « robot gibier » il perçoit les données suivantes : {nourriture à 10m Nord, prédateur 20m Est}. Le robot doit alors pouvoir deviner que « nourriture à 10m nord » est une donnée qui concerne l'objectif « se nourrir ». Ce filtrage est assuré par la fonction de sélection m_i. Cette fonction est considérée comme une projection de l'espace d'état global à un espace d'état local :

$$S_i = \{s \mid s \in S, m_i(s) = s_i\}$$

Si ainsi définie est une abstraction, en apprentissage automatique *Si* est vu comme un ensemble d'états cachés. Néanmoins, cacher des détails inutiles à un agent ou à un processus d'apprentissage peut s'avérer plus avantageux en lui épargnant des traitements inutiles. Agre (1988) définit cette abstraction comme *passive*.

4.3.3 La fonction de récompense

Dans un module d'apprentissage, la fonction de récompense attribut une valeur nulle à la plupart des cas et certaines récompenses positives pour les états buts. Dans un apprentissage multi-modules, la fonction récompense est une composition de la récompense de tous les modules. La récompense globale est en générale la somme de toutes les récompenses :

$$R(s) = \sum_{i=1}^{n} R_i(m_i(s))$$

Où *n* est le nombre de modules. Bien sur, il est possible de définir des coefficients pour donner des priorités à certains modules par rapport aux autres ou des agrégations plus complexes que la seule somme pondérée.
Reprenant toujours l'exemple du « Robot gibier », les récompenses seront :

$$R_{Se-nourrir}(s') = \begin{cases} 1 \ si \ attraper \ nourriture \\ 0 \ Sinon \end{cases} \quad R_{Survivre}(s') = \begin{cases} -5 \ si \ s'est \ fait \ manger \\ -1 \ si \ à \ 1m \ du \ danger \\ 0 \ Sinon \end{cases}$$

Aucune règle n'est établie sur la définition de ces fonctions de récompenses et la fonction de sélection, elles restent à l'appréciation du concepteur. De ce fait, cet aspect l'apprentissage par renforcement est parfois vu comme un « art » car délicat à définir si ce n'est au travers de l'expérience du concepteur (PDMIA, 2004).

.

4.3.4 La fonction de mise-à-jour de la valeur d'état

Pour chaque module, une fonction de mise-à-jour de la valeur d'état est définie, c'est une extension de l'expression (4) en intégrant la fonction de sélection :

$$Q_i(m_i(s), a) \leftarrow (1-\alpha) Q_i(m_i(s), a) + \alpha \left[R_i(m_i(s)) + \gamma \max_{a \in A} Q_i(m_i(s'), a') \right] \qquad (5)$$

Lorsque l'agent exécute une action et perçoit les récompenses, tous les modules calculent leur nouvelle Q-valeur. Même si l'agent n'a pas d'information sur l'état global, une estimation de la valeur-Q globale doit être faite pour choisir l'action à exécuter. Valeur-Q globale peut être calculée en utilisant les valeurs-Q locales et de différentes manières :

- En lui attribuant la Q-valeur locale maximale « Nearest neighbor » $Q_{nn}(s,a) = \max_{1 \le i \le n} Q_i(m_i(s), a)$ Humphrys (1996). Cette stratégie est appelé le voisin le plus proche parce qu'elle permet à l'agent d'atteindre ses objectifs les plus proches.
- Et la plus connue « Greatest mass », une simple somme des valeurs-Q locales $Q_{gm}(s,a) = \sum_{i=1}^{n} Q_i(m_i(s), a)$ Karlsson (1997). Cette stratégie semble plus raisonnable, car même si les tâches sont indépendantes dans certaines situations elles peuvent s'influencer. Par exemple pour le robot gibier, il vaut mieux s'éloigner de la nourriture si le prédateur est très proche.

Ainsi, l'algorithme de l'apprentissage aura la forme suivante (figure 4.8)

Initialisation à 0 $Q_i(s_i,a)$

Répéter pour chaque tentative de prise de décision:

a ← action donnée pour la quelle $Q_{nn}(s,a)$ est maximale (en passant par l'étape d'exploration)

Exécuter a, observer la récompense r et le nouvel état s'

Misa-à-jour $Q_{i,t+1}(m_i(s),a)$ (expression 5)

S← s'

Jusqu'à l'état s terminal

Figure 4.8. Algorithme du Q-learning/SARSA multi-objectifs (Sprague et Ballard, 2003)

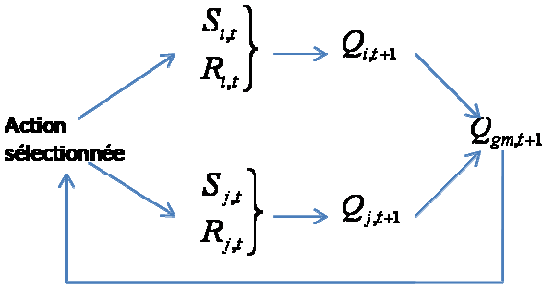


Figure 4.9. Déroulement de l'algorithme

La figure 4.9 montre comment les valeurs des deux modules sont agrégées pour donner une décision qui va être effectuée par l'agent.

5 Modélisation générique d'un système multi-agent réactif et adaptatif basé sur l'apprentissage par renforcement multi-objectif

Dans cette section, nous présentons la modélisation détaillée d'un agent apprenant par renforcement et satisfaisant deux objectifs, puis le modèle de négociation du système multi-agent pour la prise de décision dans le cadre de pilotage de production a flux continu en général.

5.1 Modélisation de l'agent apprenant par apprentissage par renforcement

Dans un problème de pilotage de production, notamment pour le pilotage de la raffinerie, les agents substituent les ressources de production (Bac de stockage avec les pompes qui lui sont attribués ou les équipements de production telle qu'une colonne de distillation…) ainsi que les produits à fabriquer (les carburants, lubrifiants ou paraffines…).

Le but de notre système de pilotage est d'assurer les deux fonctions principales de pilotage : production et maintenance de la façon la plus efficace en se basant sur une approche d'apprentissage par renforcement dans un contexte multi-agent. De ce fait, un agent apprenant aura l'architecture suivante Figure 4.10:

Ce qui est présenté décrit une architecture générique d'un agent, avec un module perception un autre d'action et le module décisionnel. Ce module décisionnel est composé de deux modules MDP appliquant l'apprentissage par renforcement, chaque module est dédié à un objectif : un module pour le pilotage de production et un autre pour le pilotage de la maintenance. Ces deux modules proposent des actions à exécuter pour chaque état du système. Un module arbitre est mis en place dans ce module décisionnel pour choisir l'une des deux actions à exécuter.

Chaque module est modélisé selon le contexte industriel, où l'ensemble des états et actions sont à définir sans oublier la fonction récompense qui permet au module de construire et d'apprendre sa politique optimale.

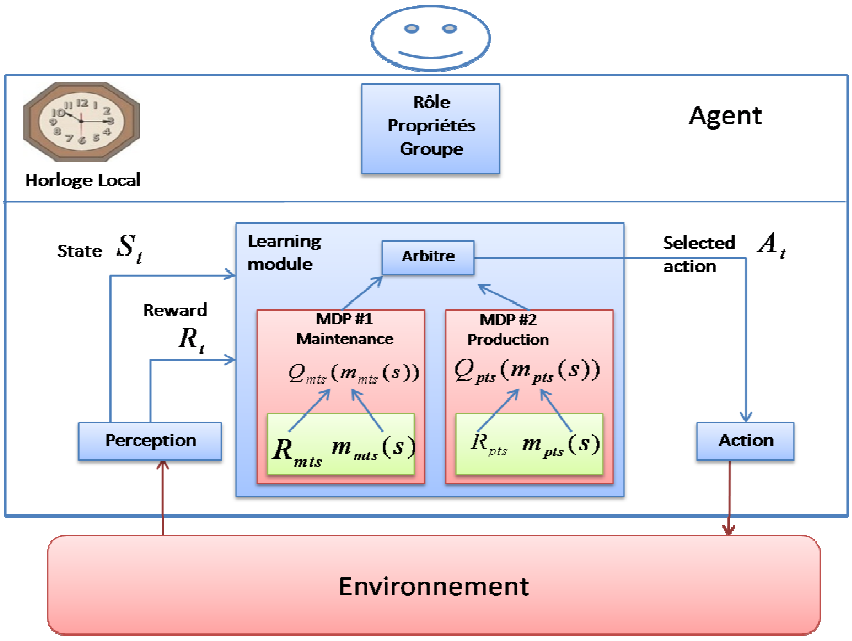


Figure 4.10. Architecture générale d'un agent apprenant

5.2 La négociation pour la prise de décision de pilotage

Un agent produit possède une gamme opératoire, c'est-à-dire, un ensemble de tâches à se faire exécuter par un ensemble de ressources selon un ensemble de contraintes, ces contraintes sont souvent de caractère opérationnel (exemple : la précédence de tâches, la type et la quantité du produit à traiter…). Pour chaque tâche, l'agent produit lance une requête à l'ensemble des ressources susceptibles de lui exécuter ces tâches. Puis les agents ressource lancent leur module décisionnel, pour qu'ils décident soit d'exécuter une tâche de maintenance (qui est présente dans leur plan d'entretien) ou de faire une proposition d'exécution.

Les agents ressource, qui décident de faire une proposition, envoient leurs offres avec le cout de cette décision qui est représenté par la valeur Q(s,a) de l'action à entreprendre pour cet état. Selon cette valeur et selon les critères de l'action, l'agent produit repond à une ressource affirmativement et négativement aux autres. De ce fait, un problème de décision d'allocation est alors résolu. Ce processus est présenté dans la figure 4.11.

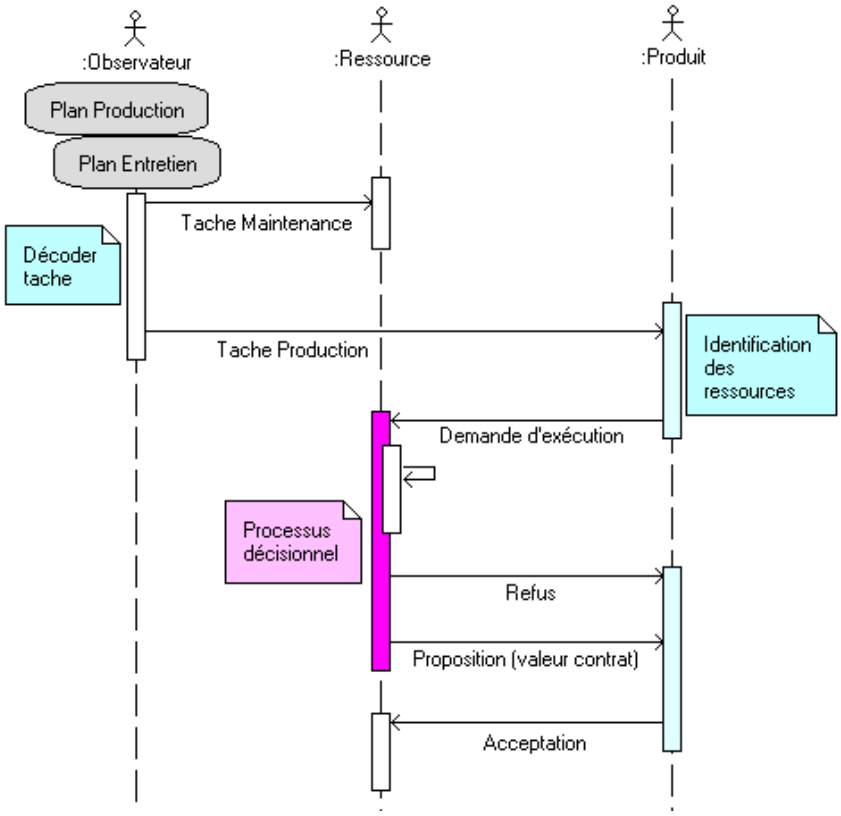


Figure 4.11. Scénario de négociation pour la prise de décision

6 Conclusion

Dans ce chapitre, nous avons présenté les fondements de l'apprentissage par renforcement. Ensuite, nous avons montré comment plusieurs agents peuvent apprendre en même temps, quels sont les différents niveaux de coopération et les différentes architectures d'apprentissage.

Nous avons ensuite, détaillé la modélisation d'un agent apprenant dans un cadre multi-objectif (de production et de maintenance) et la négociation inter-agents pour la prise de décision pour le pilotage d'un système de production.

Dans le prochain chapitre nous développons un modèle applicatif correspondant à l'ensemble des spécifications présentées dans ce chapitre dans l'objectif de piloter un système de production à flux continu. Ce chapitre décrit également un ensemble d'expérimentations qui ont été réalisées sur le cas d'étude de l'unité 3100.

Chapitre 5 : Expérimentation et analyse de l'application sur le pilotage de l'unité 3100

1 Introduction

Nous appliquons dans ce chapitre les spécifications présentées dans le chapitre précédent dans le cadre du pilotage de l'unité 3100. Nous expliquons les différents paramètres pris en considération et les critères mis en place pour évaluer l'évolution des performances de notre système de pilotage.

La première partie de ce chapitre décrit plus en détail l'unité 3100 qui a été introduite dans le chapitre 1. Cette description détaillée est nécessaire pour bien appréhender la phase « d'agentification » du problème. La partie suivante détaille l'application de notre modèle pour ce cadre applicatif. Le simulateur et les résultats expérimentaux sont enfin introduits.

2 Description pratique de l'Unité 3100

Le contexte industriel auquel nous nous intéressons est l'unité 3100 de la raffinerie d'Arzew (chapitre 1), c'est un prototype de la production à flux continu: le produit de base (l'huile de base) est reçu dans les bacs intermédiaires, puis passe par le manifold (cerveau de connexion) pour être acheminé vers les bacs de stockage d'huile finie (voire figure 5.1). Sachant qu'une huile finie est constituée, d'au moins, de deux types d'huiles de base, le transfert de ces deux types d'huile avec les quantités nécessaires sont deux tâches qui doivent être successives. Par exemple, pour fabriquer 200 tonnes de NAFTILIA 20W50 (huile finie) il faut mélanger 25,26 tonnes de SAE10 et 155,24 tonnes de SAE30.

Si le transfert vers le manifold commence par SAE10 tout de suite après devra se faire le transfert de SAE30 et vice-versa.

Nous décrivons dans la suite les éléments clés de l'unité.

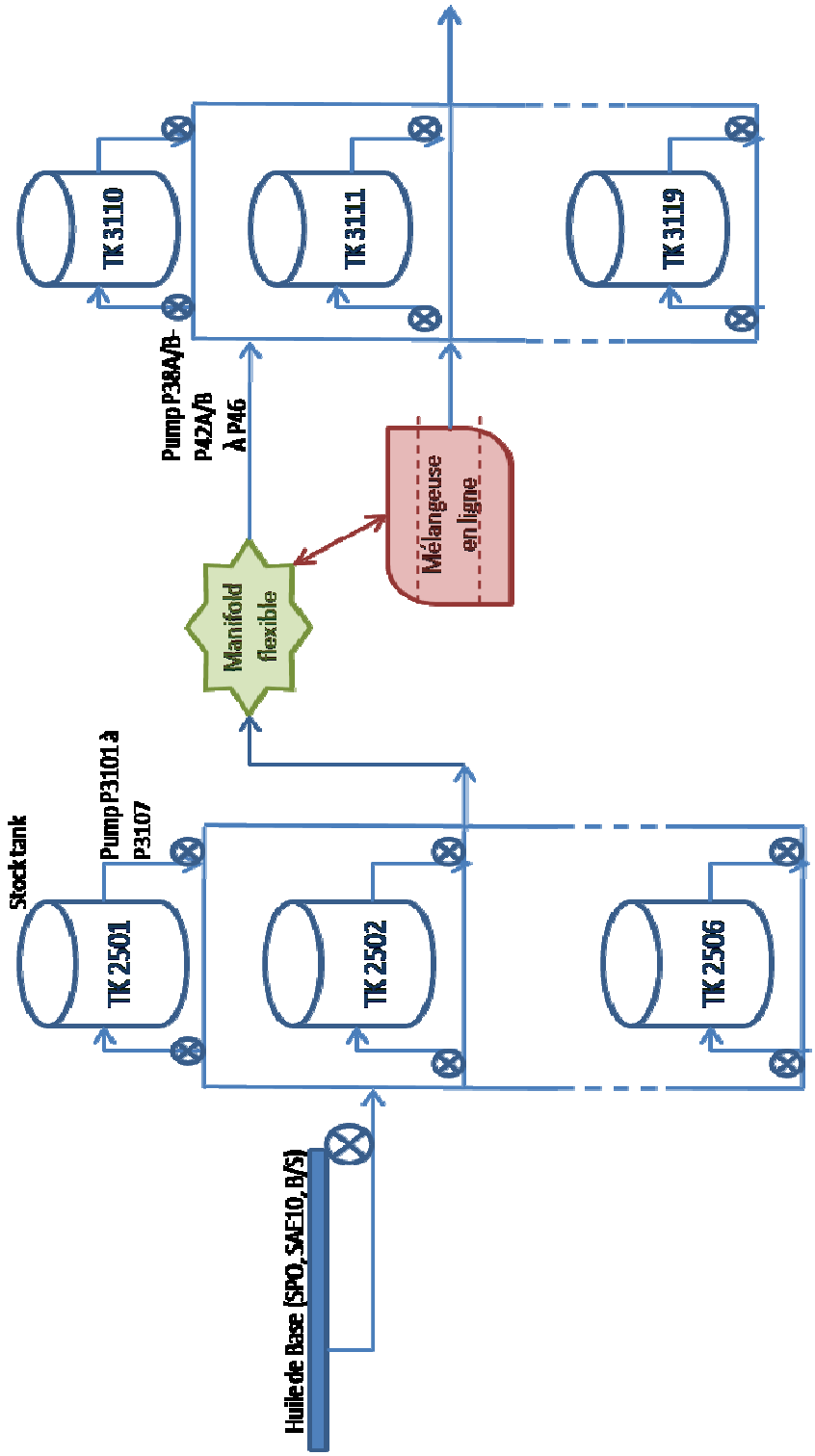


Figure 5.1. Schéma synoptique de l'unité 3100

2.1 Les bacs de stockage de l'huile de base

Les bacs de stockage sont des bacs à toit flottant (les toits flottants permettent d'évacuer la vapeur et l'air) avec une capacité limitée de l'ordre de 2300 tonnes, et une capacité minimale Ca_{min} et une capacité maximale Ca_{max} à ne pas dépasser. Chaque bac stocke un type particulier d'huile de base qui est de 4 grades différents (voir table 5.1). Il est possible de changer l'huile de base stockée dans un bac mais ceci nécessite le rinçage du bac, une opération fastidieuse et de longue durée, ce qui est souvent évité.

Bac	Pompe principale	Débit M3/h	Huile de base	TONNAGE tonne
TK2501	P3101	45	SPO	2343
TK2502	P3102	45	SAE10	2371
TK2503	P3104	45	SAE30	2399
TK2506	P3105	45	SAE30	1924
TK2504	P3106A	45	B/S	1968
TK2505	P3106B	45	B/S	1968

Table 5.1. Caractéristiques des bacs de stockage de l'huile de base

Les huiles de base sont classées de la moins dense à la plus dense (SPO →B/S).Chaque bac est associé à deux ou trois pompes avec un débit de 45m^3/h. Ces pompes sont utilisées pour le remplissage ou la vidange des bacs.

2.2 Le manifold et la mélangeuse

Le manifold est un centre de connexion de tuyaux qui relie les différents bacs entre eux avec la mélangeuse. Le manifold est une ressource unique, ce qui signifie qu'un seul produit peut être réalisé à la fois.

La mélangeuse est une machine unique (figure 5.2), elle est utilisée pour le mélange en continu des huiles de base pour la fabrication des huiles finies. Cette machine reçoit les différents huiles de bases qui vont constituer l'huile finie à fabriquer, en même temps (opérations en parallèle).

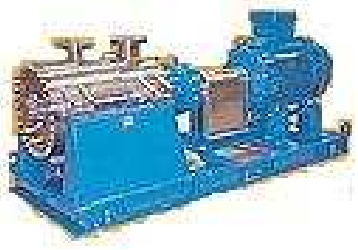


Figure 5.2. Mélangeuse centrifugeuse (mélangeuse en ligne)

2.3 Les bacs de stockage de l'huile finie

Les bacs de stockage sont aussi des bacs à toit flottant avec une capacité limitée (voir table 5.2). Les bacs peuvent contenir différents types d'huile finie car ils sont déjà constitués de grades différents d'huile de base (sauf si la constitution de base est très différente, mais cet aspect relatif à la chimie des composants n'est pas considéré dans cette thèse).

BAC	Tonnage tonne
TK3109	**2673**
TK3110	**2673**
TK3111	**2900**
TK3112	**2898**
TK3113	**2373**
TK3114	**2373**
TK3115	**2673**
TK3116	**2373**
TK3117	**2373**
TK3118	**2373**

Table 5.2. Les bacs de stockage de l'huile finie

L'unité fabrique plus de 40 types d'huile finie différents (voir l'exemple de la table 5.3), chaque huile respecte la formule suivante :

```
X1% Hb1 + X2% Hb2+ X3% Additif1 …… (1)
```

Où : ***Xi*** est un coefficient et ***HBi*** est une huile de base.

Huile finie		Huile de base				
Grades	**Quantité à fabriquer** tonne	**SPO**	**SAE10**	**SAE30**	**BS**	**Total**
NAFTILIA 20W50	200		25,26	155,24		**180,5**
TISKA 32	200		198,4			**198,4**
CHELIA VPS 20W40	400		95	211,4	60	**366,4**
TASSILIA EP 90	120		24	12	78	**114**
Totale	**920**	**0**	**342,66**	**378,64**	**138**	**859,3**

Table 5.3. Quatre huiles finies à fabriquer et leur composition

2.4 La typologie des tâches à traiter

2.4.1 Tâche de fabrication

La formule (1) détermine la gamme opératoire du produit à produire. Chaque quantité d'huile de base à utiliser pour la fabrication de l'huile finie représente une tâche de production dont la durée est donnée par la quantité à transférer puisque le débit des pompes est connu.

Exemple :

Pour fabriquer 200 tonnes de NAFTILIA 20W50 (table 5.3), il faut :

- Tâche 1 : 25.26 tonnes de SAE10 → 29,37 m3 → 40mn
- Tâche 2 : 155,24 tonnes de SAE30 → 178,43 m3 → 4 heures

La durée approximative pour fabriquer 200 tonnes de NAFTILIA 20W50 est donc de 5h.

2.4.2 Tâche de remplissage

L'unité 3100 reçoit de l'unité en amont, unité de fabrication des huiles de base, un programme décadaire contenant les dates d'envoie des huiles de base avec leurs types et leurs quantités, exemple table 5.4. Ces quantités déterminent les caractéristiques des tâches de remplissage : la date d'arrivée, la durée de la tâche et le type de l'huile qui déterminent les bacs susceptibles de les recevoir.

Exemple :

Tâche de remplissage 1 : transfert de 540 tonnes de SAE10 → 628 m3→ durée du transfert 14h. La date de début de la tâche : 12h.

Huile de base	Quantité Tonne	date de réception H
SAE10	540	15
SAE10	277	54
SAE30	817	0
B/S	1376	0
B/S	636,6	33

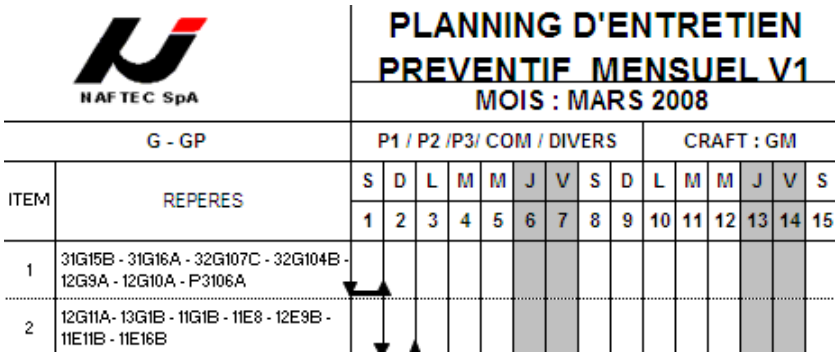
Table 5.4. Exemple de plan de réception décadaire

2.4.3 Tâche de maintenance préventive

Les tâches de maintenance préventive sont également dictées par le staff de l'entreprise. Des plans annuels sont donnés à l'unité décrivant ces tâches de maintenance en donnant : l'activité à effectuer, les moyens à mettre en œuvre, la ressource concernée, la durée de la tâche et la date présumée.

Exemple :

Dans la partie du plan d'entretien préventif mensuel présentée dans la figure 5.3, nous pouvons remarquer dans le champ 'repères' le code : P3106A le code de la pompe du bac TK2504. Cette pompe subira un entretien la journée du 1er Mars (l'heure fixe n'est pas donnée), cet entretien rend la pompe indisponible à ce moment. Les équipes de maintenance déterminent eux-mêmes la durée de la tâche.



NAFTEC SpA

PLANNING D'ENTRETIEN PREVENTIF MENSUEL V1

MOIS : MARS 2008

G - GP		P1 / P2 /P3/ COM / DIVERS									CRAFT : GM					
ITEM	REPERES	S	D	L	M	M	J	V	S	D	L	M	M	J	V	S
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	31G15B - 31G16A - 32G107C - 32G104B - 12G9A - 12G10A - P3106A															
2	12G11A- 13G1B - 11G1B - 11E8 - 12E9B - 11E11B - 11E16B															

Figure 5.3. Parcelle d'un plan d'entretien préventif mensuel

Dans le cadre de nos travaux, les informations utiles concernent la ressource en maintenance, la durée de la tâche et la date provisoire d'intervention.

Pour les installations pétrolières, des priorités sont données aux tâches de maintenance, de la plus urgente à la moins urgente. Ces priorités sont :

- Priorité 1 : danger immédiat
- Priorité 2 : danger imminent
- Priorité 3 : impératif ou nécessaire
- Priorité 4 : nécessitant une demande d'arrêt mais pas très urgent
- Priorité 5 : code donné pour une tâche que les services de maintenance ont décidé de reporter

2.4.4 Tâche de maintenance curative

Les priorités 1 et 2 caractérisent logiquement les tâches de maintenance curative.

Même si l'unité subit un entretien de rigueur, des pannes peuvent se produire. La réparation de ces pannes a la priorité la plus élevée. En analysant la nature de la panne, les équipes de maintenance déterminent la durée de réparation.

3 Objectifs opérationnels pour le système de pilotage proposé pour l'unité 3100

Le but de notre étude est de permettre au système de pilotage d'acquérir une politique de placement des différentes tâches sur l'horizon de temps en leur allouant les ressources nécessaires de la meilleure façon possible, si possible optimale et que le système puisse s'adapter aux fluctuations de l'environnement et d'y être réactif avec le soucis d'améliorer constamment les résultats. Nous pouvons alors distinguer les objectifs suivants :

L'allocation des ressources aux tâches : Le système de pilotage proposé doit pouvoir prendre des décisions pour allouer les ressources aux tâches de production, c'est-à-dire dire quel serait le bac concerné par l'opération de remplissage ou de vidange pour fabriquer une huile finie.

Chercher l'optimalité : Permettre au système de faire le placement des tâches de production et de maintenance dans l'horizon temporel d'une façon performante, en minimisant le temps d'achèvement du projet de fabrication (réaliser le plan décadaire proposé par le staff de l'entreprise).

Assurer la réactivité : La réactivité est l'une des performances du système de pilotage que nous voulons assurer. Ceci sous-entend que le système, face à une imprévision, indisponibilité de ressources suite aux pannes ou l'arrivée de demande de production urgente…, puisse prendre les décisions nécessaires pour placer les nouvelles tâches, de production ou de maintenance, sur l'horizon temporel tout en maintenant l'efficacité du système.

Être adaptatif : C'est une autre performance recherchée par le système, ceci implique que le système puisse améliorer la qualité de ses décisions, en termes d'optimalité, au fur et à mesure qu'il rencontre des changements dans son environnement.

La partie suivante décrit une application de notre approche pour atteindre ces différents objectifs.

4 Application de notre approche pour le pilotage de l'unité 3100

Le point de départ de notre démarche applicative se base sur le mécanisme d'agentification selon le modèle Aalaadin qui a été introduit dans le chapitre précédent (figure 5.4).

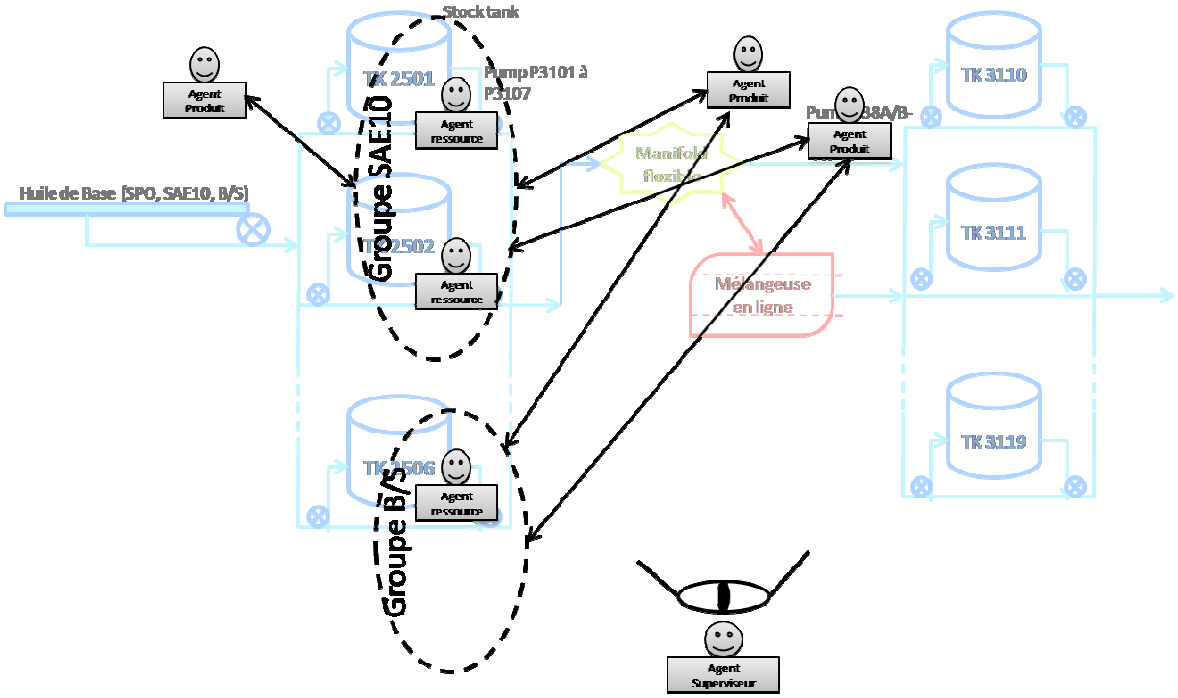


Figure 5.4. Agentification du problème

Les différentes étapes de cette agentification sont décrites dans la partie suivante.

4.1 Agentification du système de pilotage

Conformément à ce qui a été proposé dans le chapitre précédent, nous devons identifier les agents produits, ressources et l'agent superviseur/observateur.

Un **Agent Produit** représente chaque huile finie à produire : cet agent stocke les données concernant ce produit : sa gamme opératoire, l'ensemble des tâches de fabrication en donnant le type de l'huile de base, la quantité requise qui détermine la durée du transfert. Cet agent lance la procédure de négociation avec les Agents Ressource pour se faire exécuter ses tâches. Les agents produit représentent aussi les huiles de base à stocker que l'unité reçoit de l'unité en amont : ceci pour permettre aux agents ressource de négocier les tâches de remplissage. Ces tâches sont définies par : le nombre de parcelles à stocker, la date d'arrivée de la tâche et le type de l'huile de base à stocker.

Un **Agent Ressource** représente ici un bac de stockage avec les propriétés de ce bac : sa capacité, le débit du produit entrant et sortant, le type du produit (qui détermine son groupe) et la quantité du produit présente à chaque instant *t*.

Il reçoit le plan de maintenance préventive, ceux sont des tâches qui doivent être absolument exécutées mais qui leur emplacement dans le temps est négociable, au niveau de cet agent, avec les tâches de production. Ceci grâce à l'apprentissage multi-objectif.

Enfin, l'agent reçoit les requêtes d'exécution des tâches de production des Agents produit et en consultant son état, propose une date d'exécution pour la tâche ou refuse si son état ne lui permet pas de faire une proposition.

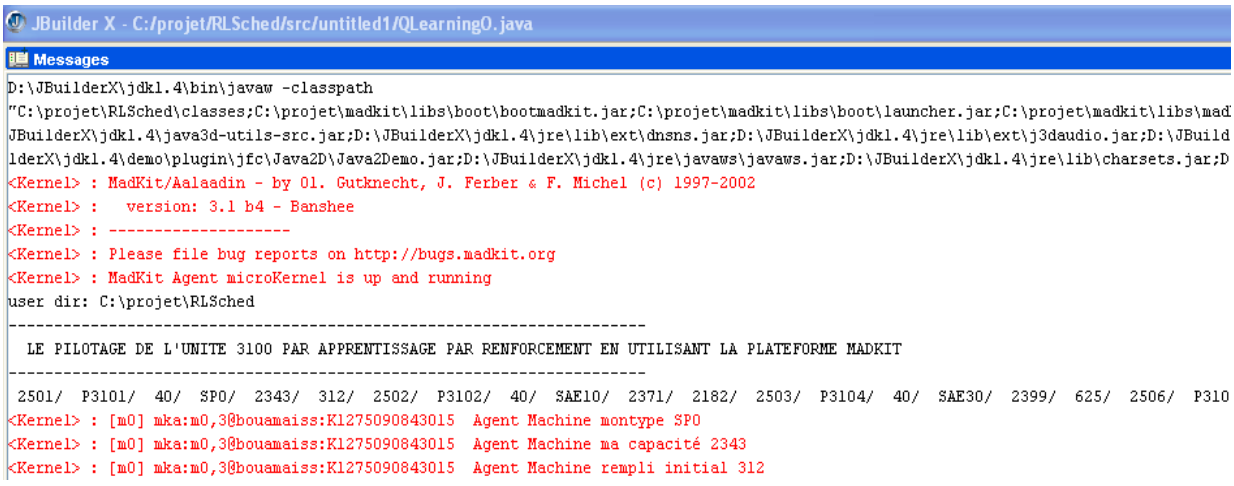


Figure 5.5. Initialisation de la plateforme et lancement des agents (ressource m_0 visible représentant le bac TK2501)

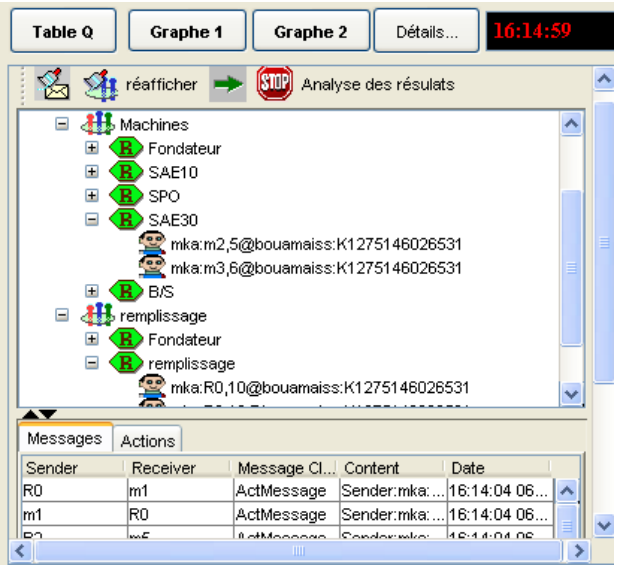


Figure 5.6. Vue de l'agent observateur montrant les différents groupes et différents agents et les premiers messages échangés

Les figures 5.5 et 5.6 montre les agents lancés dans notre système, rangés par groupes et rôles.

L'agent système ou l'agent observateur est un agent unique dans le système, son rôle est d'observer tout le système et évaluer l'évolution des performances du système. La figure 5.7 montre le cycle de fonctionnement de cet agent observateur: Lorsqu'un agent produit finalise une tâche envoi un message à l'agent observateur en lui donnant les caractéristiques de cette exécution: la durée de la tâche, la ressource qui l'a exécuté et le temps d'attente enregistré. Et à la fin de la réalisation de tout le plan de production, les agents ressources envoient également des informations: leur durée opératoire, leur temps d'arrêt le temps de soumission à l'entretiens… ces informations servent à l'agent observateur pour évaluer les critères de performances du système. Cet agent intègre les critères d'évaluation, des critères tels que la durée totale d'un projet, le taux d'occupation des ressources et le débit du produit entrant et sortant. Ces critères et leur calcul vont être détaillés dans la section 5.4 de ce chapitre.

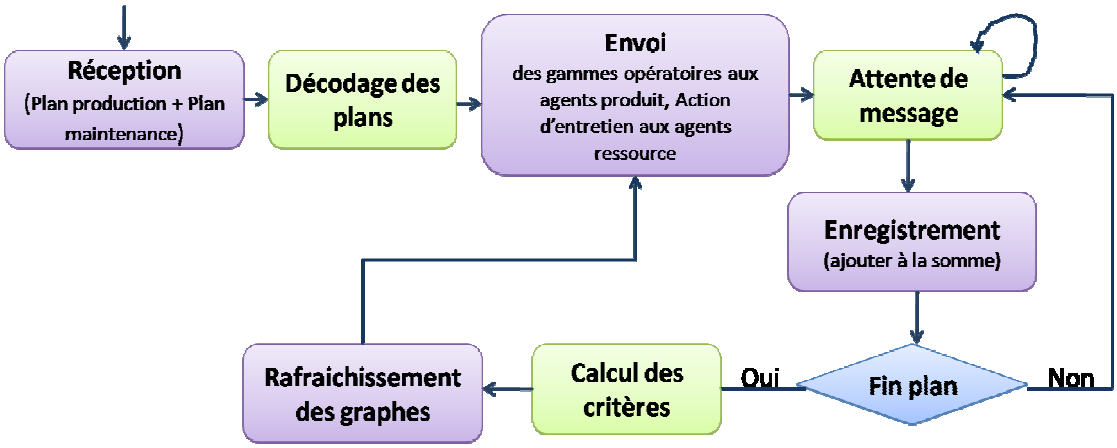


Figure 5.7. Cycle de fonctionnement de l'agent observateur

4.2 Négociation pour l'allocation des ressources aux tâches de production

La figure 5.8 décrit le scénario de négociation entrepris par les agents pour exécuter une tâche de production demandé par un agent produit. Ce scenario se base conformément aux spécifications présentées dans le chapitre 3 sur une approche de type « contract-net ».

Après avoir reçu l'ensemble des tâches de production qui le concerne, l'agent produit prépare ses requêtes pour les envoyer à l'ensemble des agents ressources qui peuvent exécuter cette tâche.

Par exemple, pour la tâche 1 du produit NAFTILIA 20W50: Tâche 1 : 25.26 tonnes de SAE10 → 29,37 m3 → 40mn

La requête se formule par conséquent de la façon suivante : « Demande d'exécution, durée 40mn, date de début 0 ». Cette requête est envoyée à tous les agents ressource du groupe SAE10.

Les Agents Ressource reçoivent cette requête. En fonction de leur états, leur modules décisionnels peuvent « refuser » la requête ou faire une proposition qui est constituée de « La valeur du contrat (Qpts) et la date de début proposée ».

L'Agent Produit peut recevoir une à plusieurs propositions. En fonction de la valeur du contrat et la date de début proposée l'agent produit choisit d'accorder le contrat, l'exécution de la tâche, à un agent ressource et envoie un message de refus aux autres.

Les agents et leurs interactions décrits, la partie suivante détaille de manière approfondie les mécanismes d'apprentissage qui ont été développés et intégrés au sein des agents.

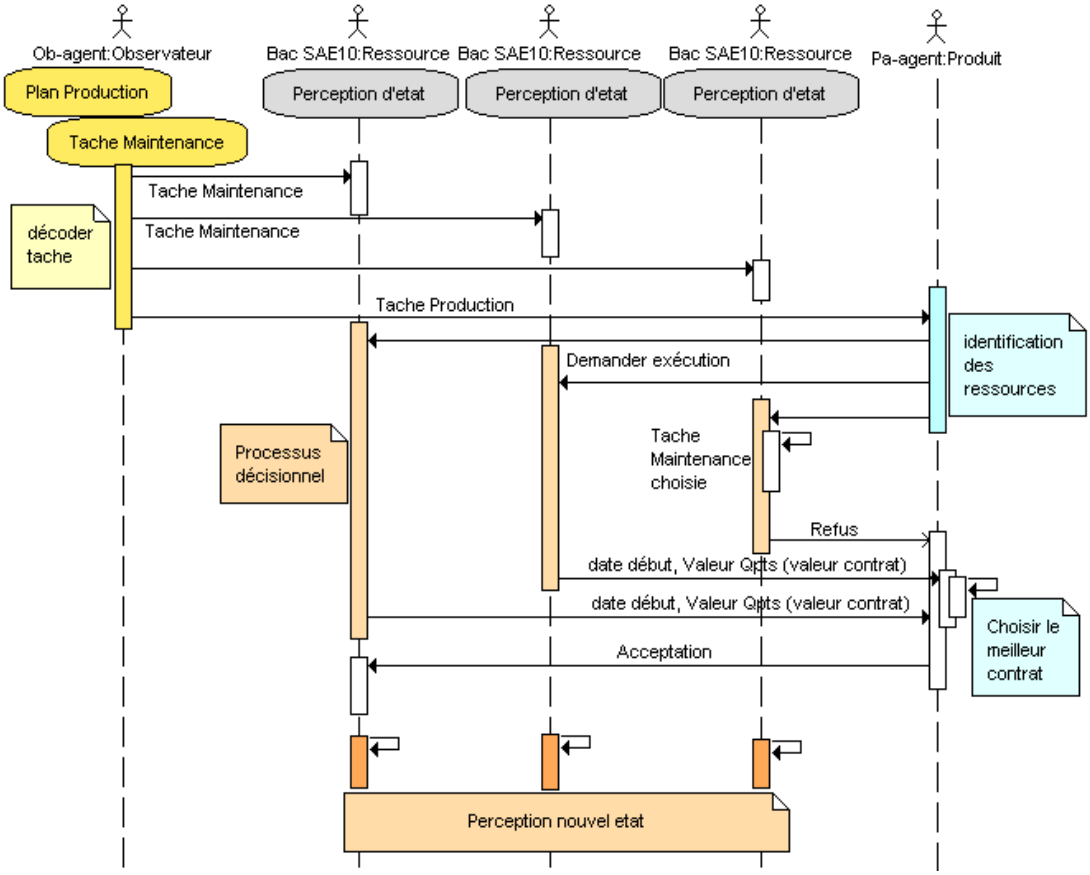


Figure 5.8. Scénario de négociation pour l'exécution d'une tâche de production

4.3 L'apprentissage par renforcement décentralisé

Comme indiqué dans le chapitre précédent, notre approche est basée sur l'apprentissage par renforcement décentralisé. Chaque agent ressource est autonome pour prendre des décisions, par exemple, d'exécuter ou non telle ou telle tâche, à partir de la connaissance sur son état, déterminé par la disponibilité du produit et la disponibilité des pompes de transfert. L'apprentissage est alors local pour chaque agent. Les travaux de Csaji et Monostori (2006), Aissani et al. (2008a) et Aissani et al. (2009a) (chapitre 2) ont montré qu'il était possible d'utiliser l'apprentissage par renforcement et l'apprentissage par renforcement décentralisé pour l'ordonnancement dynamique.

Parmi l'ensemble des algorithmes présentés auparavant, nous avons choisi d'utiliser l'algorithme SARSA pour les raisons suivantes :

- Il permet d'estimer l'utilité associée au choix d'une action à partir d'un état, car il ne suffit pas seulement d'évaluer l'état, mais plutôt d'évaluer la qualité de la prise de décision (Annexe 4). Cette pratique permet de choisir à tout moment l'action qui permet de transiter de n'importe quel état vers celui qui présente l'utilité la plus grande.
- Il permet de faire une évaluation de l'évolution du système apprenant à tout moment, c'est un apprentissage qui se fait au fur et à mesure que le système s'exécute. Il est bien

sûr possible d'utiliser des critères d'évaluation, lesquels sont à évaluer tout au long du processus d'apprentissage.

- Enfin et surtout, cet algorithme peut être étendu à notre contexte, c'est-à-dire à l'apprentissage dans un système décentralisé et multi-objectif.

4.3.1 Paramétrage de l'apprentissage décentralisé

Afin d'appliquer cet apprentissage décentralisé il faut déterminer :

- L'ensemble des états actions pour chaque agent ressource
- La définition des objectifs locaux qui sont compatibles avec l'objectif global
- Les critères de performance du système de pilotage pour surveiller leur évolution

a. Déterminer l'ensemble des états et actions des agents ressource

L'état de l'agent ressource est déterminé par deux paramètres :

Etat de marche : la ressource est soit « en marche » soit « à l'arrêt ». Ceci peu être déterminé grâce à la séquence de la ressource (figure 5.9). En effet, pour chaque agent ressource une liste des tâches lui est attribuée. Cette liste montre les intervalles de temps où la ressource est occupée et quand elle est libre (annexe 7). Cette séquence constitue la représentation de l'état de la ressource.

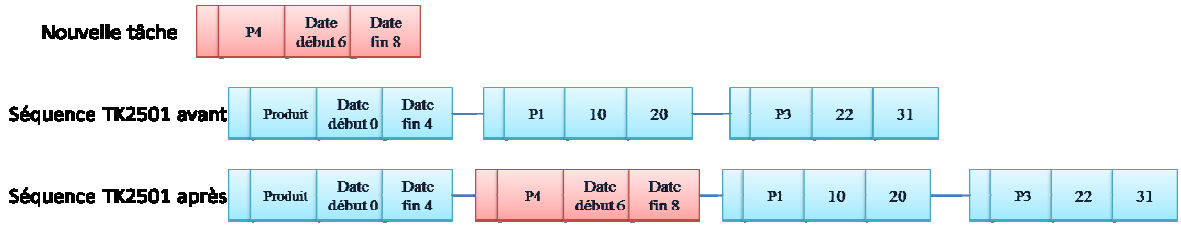


Figure 5.9. Séquence représentant l'état de la ressource

Nous sommes dans un contexte de production à flux continu où la capacité de la ressource et la quantité du produit disponible déterminent la capacité de cette ressource à exécuter ou non une tâche. Donc deux états possibles, ou « quantité disponible » ou « quantité indisponible ».

Les actions d'un agent ressource doivent également être déterminées. Après avoir reçu une requête d'exécution d'une tâche, l'agent peut soit « faire une proposition avec les paramètres de la requête» ou « faire une proposition avec des données différentes» (mise en attente) ou « ne pas faire de proposition ».

b. Les fonctions de récompense ou les fonctions critiques

Dans un apprentissage décentralisé, chaque agent cherche à maximiser ses récompenses, ceci implique que chaque agent veut être occupé le maximum de temps à produire respectant ses contraintes de capacité, répondre affirmativement aux requêtes de production et avoir le minimum de temps mort.

L'agent est alors pénalisé s'il fait des propositions et qu'il n'a pas été retenu, autrement dit, il était capable d'exécuter la tâche mais il n'a pas été retenu parce que son offre n'était pas suffisamment intéressante pour l'agent produit. Ça induit aussi un temps mort pour la ressource si elle n'est pas retenue pour une autre requête. Selon le mode de coopération entre les agents, on peut définir le type de l'apprentissage et sa fonction récompense, on distingue deux types: apprentissage concurrent ou apprentissage coopératif.

4.3.2 L'apprentissage décentralisé concurrent

Selon cette approche, les agents ressource se trouvent dans un apprentissage concurrent lorsque plusieurs agents ressources font des offres pour une requête, ils sont concurrents pour cette requête, ce qui est le cas ici (cf. contract net). L'agent produit ne choisira qu'une seule ressource pour sa requête, donc l'agent de cette ressource sera récompensé et les autres pénalisés.

La fonction récompense d'un agent ressource aura la forme suivante :

$$R(s) = \begin{cases} +1 \ si \ contrat \ établi \\ -1 \ si \ proposition \ refusée \\ 0 \ \ si \ pas \ de \ proposition \end{cases} \quad \ldots\ldots\ldots(2)$$

Selon cette fonction, on accorde une récompense positive à l'agent ressource lorsqu'il a fait une proposition et qu'elle a été acceptée par l'agent produit, ceci signifie que l'agent ressource a fait une proposition intéressante selon les critères de l'agent produit. A cet effet, l'agent ressource est pénalisé s'il a fait une proposition qui a été refusé, ceci insinue qu'il a fait une proposition de mauvaise qualité. Et bien sûr aucune récompense n'est alloué à l'agent ressource s'il ne participe pas à la négociation du contrat de l'agent produit.

4.3.3 L'apprentissage décentralisé coopératif

Les agents apprenant ont aussi un objectif global à atteindre, c'est l'objectif de tout le système ou de toute l'unité.

L'unité a pour objectif de produire tout en respectant ses contraintes opérationnelles pour rester en sécurité et disponibilité, ceci est déjà assuré grâce à la veille des agents ressource au respect de leur contraintes de capacité.

Une production à flux continu implique que le produit circule en permanence dans l'unité avec un flux stable (Aissani et al., 2009a).

La fonction de récompense des agents ressource peut prendre en charge les objectifs de réduction de délais pour inciter les agents à les atteindre (Aissani et al., 2009b) comme suit :

$$R(s) = \begin{cases} +1/C\max \ \ si \ contrat \ établi \\ -1 \ si \ proposition \ refusée \\ 0 \ \ si \ pas \ de \ proposition \end{cases} \quad \ldots\ldots\ldots\ldots(3)$$

Au lieu d'avoir une valeur arbitraire pour la récompense d'un agent qui a fait une proposition acceptable, on lui donne une récompense inversement proportionnelle à l'objectif global qui est le *Cmax* à cet instant (*Cmax* partiel). Sachant que le (Fin de traitement) *Cmax* est défini par la formule suivante: $Cmax=max_{i=1,n}(C_i)$, tel que C_i est la date de fin d'exécution de toutes les tâches du produit *i* c'est-à-dire fin de fabrication.

4.4 L'apprentissage multi-objectif (produire et maintenir)

L'agent ressource lui-même à deux objectifs à atteindre :

D'abord produire le maximum de temps et minimiser ses temps morts en participant à l'objectif global qui est « produire dans les plus courts délais ».

Ensuite, comme ceci sous-entend que la ressource est disponible et fiable pour produire, la ressource doit « respecter le plus possible sont plan d'entretien ».

Par conséquent, le module décisionnel de cet agent se scindera en deux modules, *mts* pour maintenance et *pts* pour production, synchronisés par un arbitre, tel que montré dans le chapitre précédent (Karlsson, 1997 ; voir figure 4.7).

Les paramètres d'apprentissage du premier module (module de production) ont été déjà définis dans la section précédente. Ceux pour le module maintenance sont définis dans la partie suivante.

4.4.1 Ensemble des états et actions pour le module maintenance

L'agent ressource reçoit un plan de maintenance décadaire (comme pour le remplissage), le module maintenance a pour objectif de respecter ce plan. Le module maintenance surveillera le module production pour que ses objectifs ne négligent pas l'exécution de la maintenance. Les états du module maintenance se composeront des paramètres suivants :

- La présence ou non d'une tâche de maintenance et sa priorité, à partir du plan de maintenance (table 5.5), l'agent ressource établit un tableau des tâches d'entretien ordonnées chronologiquement en précisant la date d'arrivée probable, la durée de la tâche et sa priorité exemple :

Tâche d'entretien	Durée	date début	Priorité
Changer courroie	3	2	1
rinçage bac	12	10	3
vérification étanchéité	1	31	2

Table 5.5. Tâches d'entretien

Une ressource a deux états possibles: « en marche » ou « à l'arrêt », en considérant la "présence" ou "non" d'une tâche de maintenance.

L'ensemble des actions est alors constitué de deux actions : « faire une proposition de maintenance avec sa date prévue » ou « faire une proposition sans date prévue (pour qu'elle soit déterminée par négociation)».

4.4.2 La fonction de récompense pour le module Maintenance

Le module responsable de la maintenance, a pour objectif de faire le plus d'entretien possible. Il est alors récompensé lorsque son action de maintenance est choisie et pénalisé lorsque sa tâche de maintenance est retardée ou annulée (équation 4).

$$R_{mts}(s) = \begin{cases} +1 \ si \ maintenance \ effectuée \\ -1 \ si \ maintenance \ refusée \\ 0 \ \ si \ pas \ de \ proposition \end{cases} \quad \ldots\ldots\ldots(4)$$

4.4.3 Le rôle de l'arbitre et la prise de décision

Les deux modules, production et maintenance, sont des modules concurrents, car si une tâche de maintenance a été choisie pour être exécutée par l'agent, la requête de production ne va pas être traitée et si l'agent décide de faire une proposition pour une requête de production la tâche de maintenance ne va pas être exécutée.

L'arbitre a pour rôle de choisir laquelle parmi ces deux actions va être exécutée par l'agent. En effet, lorsqu'une tâche de maintenance se présente en même temps qu'une tâche de production, le module maintenance, choisit l'action à exécuter selon sa politique (table Q_{mts}) et envoie à l'arbitre la valeur $Q_{mts(t)}(s_{mts},a_{mts})$ correspondante. Le module production fait de même, choisit l'action à exécuter et envoie à l'arbitre la valeur $Q_{pts(t)}(s_{pts},a_{pts})$ correspondante. L'arbitre choisit alors celle qui maximise les gains de l'agent ressource en choisissant l'action dont la valeur-Q est la plus importante (Humphrys, 1996).

Remarque : le module maintenance n'a pas été traité pour l'apprentissage décentralisé car les actions entreprises par ce module n'affectent que l'agent lui-même. Si l'agent ressource décide de faire de l'entretien il ne participera pas à la négociation d'une requête d'exécution d'une tâche de production.

4.4.4 L'algorithme d'apprentissage dans un module

Lorsqu'un module est incité à prendre une décision, il y a perception d'un nouvel état par l'agent, cet état est utilisé pour choisir l'action à exécuter. L'action choisie *a* est celle qui maximise $Q_{mts/pts}$(état, action) avec une probabilité de Boltzmann (équation 5). La probabilité de Boltzmann permet d'équilibrer les phases exploration et exploitation (voir annexe 6).

$$P(a|s) = e^{Q_{mts/pts}(s_{mts/pts},a)/T} \Big/ \sum_{a_i \in A} e^{Q_{mts/pts}(s_{mts/pts},a_i)} \qquad (5)$$

La valeur $Q_{mts/pts}(s, a)$ correspondante à l'action choisie est envoyée à l'arbitre, suite à cela une réponse est reçue : soit l'action a été exécutée ou non.

Suite à cette exécution, une récompense est perçue ainsi que le nouvel état (si une autre tâche de maintenance se présente).

L'agent ressource met-à-jour sa valeur $Q_{mts/pts}(s, a)$ et passe à la prochaine prise de décision, ceci est résumé dans la figure 5.10.

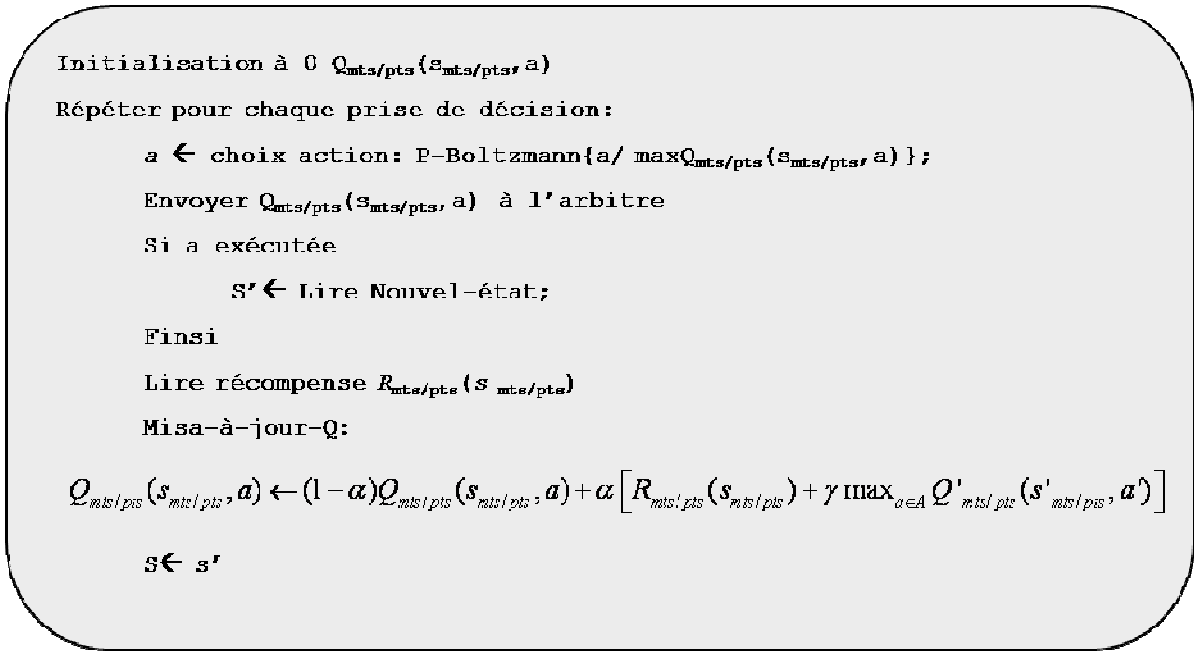


Figure 5.10. Algorithme d'apprentissage pour un module

5 Le simulateur

Nous présentons dans cette partie les informations liées au simulateur développé : l'outil de développement, les données en entrée, le simulateur (son interface), les paramétrages à réaliser, les critères de performances (sortie du simulateur). La partie qui suivra décrira les résultats obtenus.

5.1 Outils de développement

Pour réaliser les simulations, nous avons utilisé un micro-ordinateur avec les caractéristiques suivantes :

- Capacité processeur: Pentium Dual-Core 2.0Ghz x 2.0Ghz
- Capacité mémoire: 3 Go

Le langage de programmation que nous avons utilisé est le JAVA sur un environnement de développement le *JbuilderX* qui est un logiciel BORLAND pour la conception et la réalisation des applications en JAVA.

Pour la réalisation de nos agents, et comme énoncé dans le chapitre précédent, nous avons utilisé la plateforme *MADKIT*.

Madkit est une plate-forme Multi-agents pour la conception et la réalisation des systèmes multi-agents, elle implémente le modèle Aalaadin. Madkit est développé en JAVA, ce qui a facilité son intégration à notre projet.

Pour le graphisme de notre application, réalisation des graphes, nous avons utilisé la bibliothèque open-source *Jfreechart*.

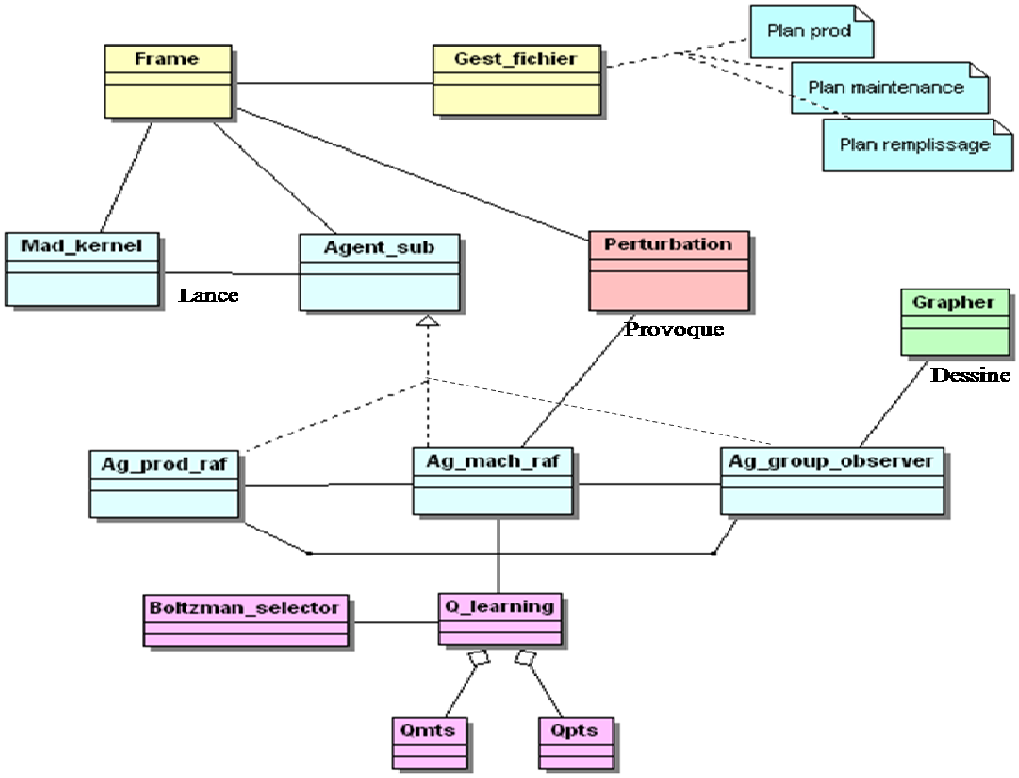


Figure 5.11. Diagramme de classe de l'outil développé

Le diagramme 5.11 est le diagramme de classe de l'outil développé: les classes en jaune sont les classes d'interface, les classes en bleu sont les classes du système multi-agent, les classes en mauve sont les classes de l'apprentissage par renforcement, la classe en vert est la classe d'évaluation des performances et la classe en rouge est la classe perturbation utilisée pour tester la réactivité du système. Toutes les relations non-indiquées sont des relations de type "utilise".

5.2 Données en entrée

Les entrées de notre simulateur sont stockées dans le Benchmark de la raffinerie, qui est constitué des fichiers suivants:

1. Un fichier qui contient ***la configuration de l'unité***. Le format du fichier est le suivant (figure 5.12):

```
numtank    pompe      flux   typehuile  tonnage  tonnageinitiale
```

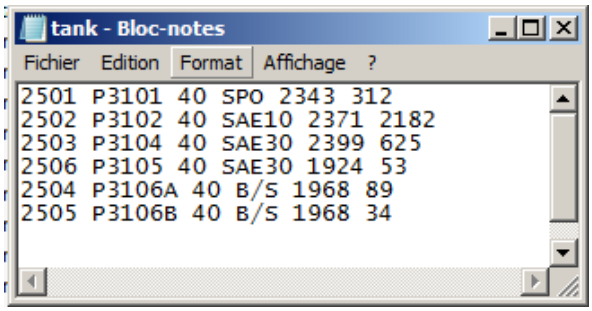


Figure 5.12. Fichier configuration de l'unité (Bacs)

Le chargement de ce fichier permet de créer et lancer les agents ressource du système avec les caractéristiques correspondantes.

2. Un fichier qui contient le ***plan de remplissage***, sont les dates d'arrivée des produits de base (l'huile de base) avec la quantité transférée:

```
Type huile     Nbr-tache quantité  date-début     date-fin
```

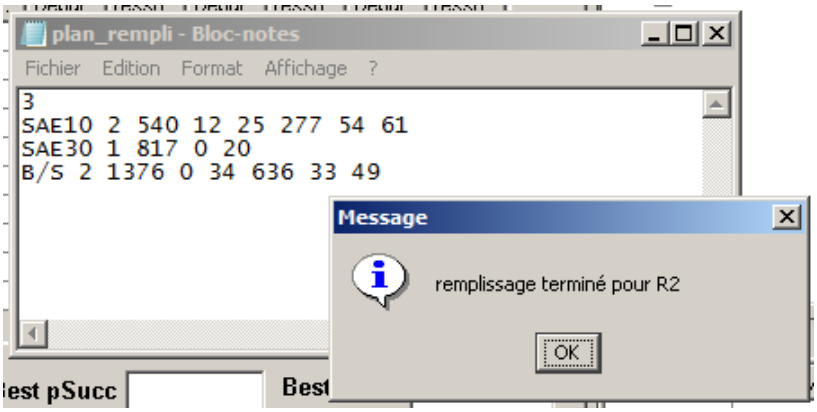


Figure 5.13. Fichier plan de remplissage

Le fichier 5.13 présentent les différentes tâches de remplissage, ce fichier permet la création et le lancement des agents produit: SAE10 avec 2 tâches de production, SAE30 avec une tâche de production et B/S avec 2 tâches de production.

3. Un fichier qui contient le ***plan de production***, qui concerne les tâches de production de l'huile finie. Ce fichier à la forme suivante:

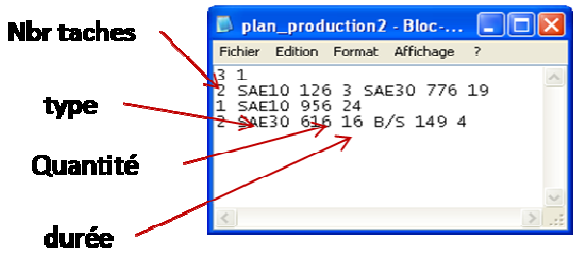


Figure 5.14. Fichier plan de production

Le fichier 5.14 permet de créer les agents produits "huile finie", par exemple le premier produit à une gamme opératoire de 2 tâches, la première de type SAE10 avec une quantité de 126tonnes et la durée 3 et la deuxième de type SAE30 avec une quantité de 776 avec une date de début 19.

4. Le fichier plan d'entretien, qui représente le calendrier des tâches de maintenance préventive. Le fichier a la forme suivante:

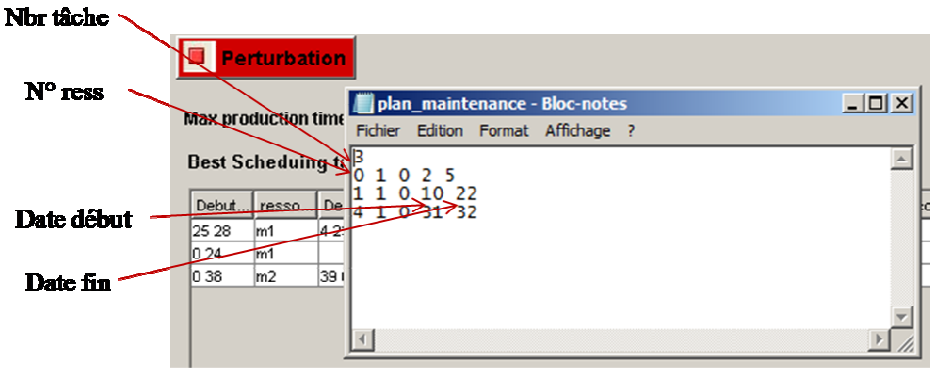


Figure 5.15. Fichier plan d'entretien

Le fichier 5.15 est décodé et les identifiants des tâches et leurs caractéristiques sont envoyés aux agents ressource correspondants.

Les données stockées dans ces fichiers sont des données issues des plans de production et plans d'entretien de la raffinerie pour l'unité 3100 pour l'année 2009 (Un exemple a été autorisé à être publié, il figure dans l'annexe 5).

5.3 Interface du simulateur

Ces fichiers sont chargés à partir de notre outil de simulation dont l'interface est présentée dans la figure 5.16. Cette interface intègre les fonctions suivantes:

1. Le menu qui permet de charger les fichiers contenant les données du problème, les fichiers: plan de production, plan de remplissage et plan d'entretien.
2. Le chargement de ces fichiers permet de créer et lancer les agents du système avec l'agent observateur dont l'interface est intégrée à l'interface principale (2).
3. Chaque solution proposée par le système est affichée sur le tableau (3) où chaque ligne représente les tâches d'un produit et chaque tâche est représentée par la ressource qui l'a exécuté, la date de début et la date de fin, puis une case montrant le Cmax de cette solution et le numéro de l'itération.
4. Dans le tableau (4) est enregistrée la meilleure solution à un instant donnée.
5. Dans la zone (5) sont affichés les graphes montrant les évolutions des différents critères de performance.

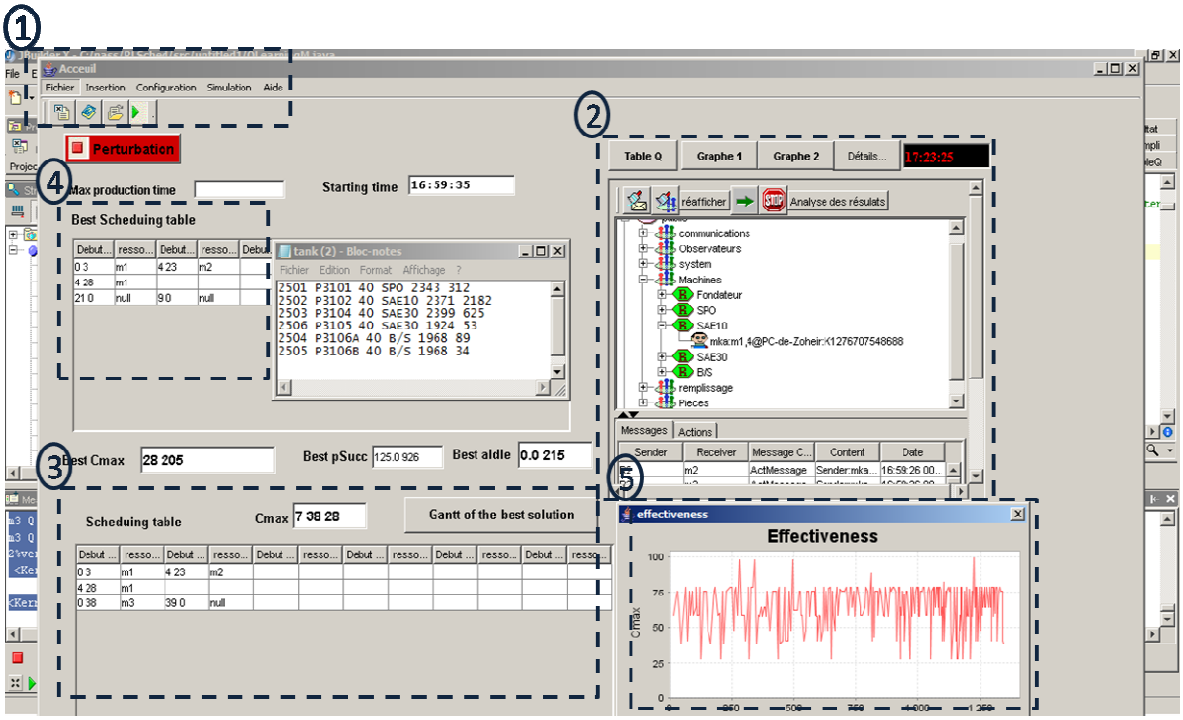


Figure 5.16. L'interface de l'outil développé

Avant de décrire les expérimentations, nous présenterons le paramétrage des algorithmes d'apprentissage.

5.4 Paramétrage des algorithmes d'apprentissage

Pour chaque simulation, un certain nombre de paramètres doivent être ajustés. Ces paramètres sont :

Les paramètres de la fonction de mise-à-jour de la valeur d'utilité d'un état ou plutôt d'utilité état/action donnée par la formule :

$$Q_{mts/pts}(s_{mts/pts},a) \leftarrow (1-\alpha)Q_{mts/pts}(s_{mts/pts},a)+\alpha\left[R_{mts/pts}(s_{mts/pts})+\gamma \max_{a\in A} Q'_{mts/pts}(s'_{mts/pts},a')\right]$$

Les coefficients α et γ sont à définir

α est le pas du gardien ou pas de l'apprentissage, c'est un réel positif compris entre 0 et 1. Plus ce nombre est grand plus la politique apprise a du poids, car le Q(s',a') a plus de poids. Cette politique apprise ne doit avoir de poids que si l'agent a réitéré suffisamment de fois son processus décisionnel.

De ce fait, nous avons choisi de définir α comme étant inversement proportionnel au nombre de fois dans lequel l'agent a été amené à décider (Aissani et al. 2008a) pour cet état.

$$\alpha = 1/IT$$

γ est le facteur de pondération sur le cumul des gains R à estimer à t, tel que :

$$R(t) = r(t) + \gamma\, r(t+1) + \ldots + \gamma^k\, r(t+k) + \ldots$$

Appelé aussi le Facteur d'oubli, Takadama and Fujita (2004) ont montré que plus γ est proche de 1 plus l'algorithme converge plus rapidement. Nous avons choisi $\gamma = 0.9$, ce qui correspond à une valeur usuelle dans la littérature.

En ce qui concerne la probabilité de choix de l'action, la probabilité de Boltzman présente un paramètre à fixer, la température. La probabilité de Boltzmann est donnée par la formule (1) et la température T est donnée par $T = \frac{1}{nIT}$ où *nIT* est le nombre d'itérations, c'est-à-dire le nombre de fois que l'état avec le choix de cette action avait été testé.

Il faut alors fixer des bornes pour la température, car si *nIT*=0, alors T→ ∞ ce qui engendrera une durée très longue pour que T converge vers 0.

Nous testons alors notre algorithme d'apprentissage sur un exemple très simple dont il est très facile de déterminer un ordonnancement optimal. C'est un problème 2produits X 3machines en considérant que les capacités des ressources sont illimitées.

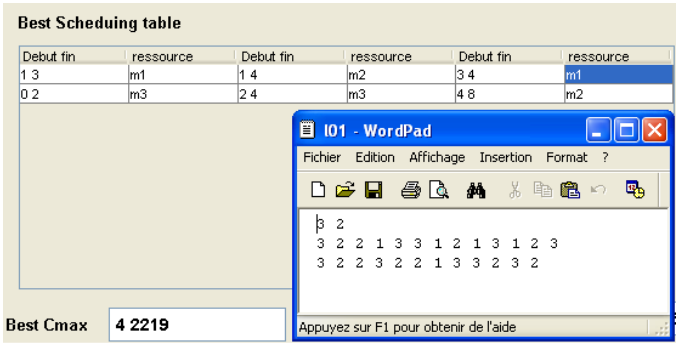


Figure 5.17. Définition de l'exemple

Le fichier L01 qui apparait sur la figure 5.17 est le fichier de définition du problème, selon la forme (figure 5.18):

Machine produi														
		Tache 1							Tache 2					
	nbr taches	Durée	N ressource	nbr ress 1	…	ress N			Durée	N ressource	nbr ress 1	…	ress N	…

Figure 5.18. Codification du problème

Les expérimentations sur la température ont donnée les résultats suivants (figure 5.19 et 5.20)

Des graphes 5.19 et 5.20, nous pouvons constater que plus l'intervalle de la température est grand plus la convergence vers un optimum est plus longue, par exemple avec Tmax=100, il a fallu 2493 itération et 44mn pour converger vers un Cmax optimum. Nous pouvons voir aussi que si l'intervalle est trop petit, par exemple Tmax=1, la convergence se fait trop rapidement, 72 n'est pas suffisante pour dire que l'agent a testé toutes les possibilités suffisamment de fois pour aboutir à la conclusion qu'il a appris la meilleure politique à entreprendre. De ce fait, nous choisissons pour nos algorithmes Tmax=10.

Remarque : il ne faut pas oublier que Tmin est toujours égale à 0.

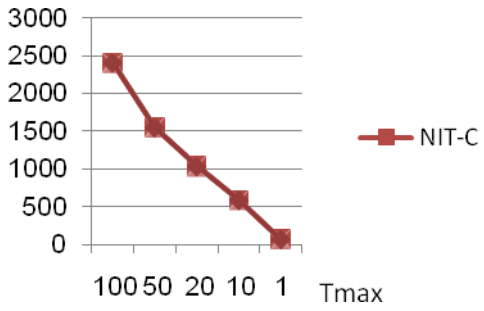


Figure 5.19. Graphe du nombre d'itération avant convergence par rapport à la température maximale

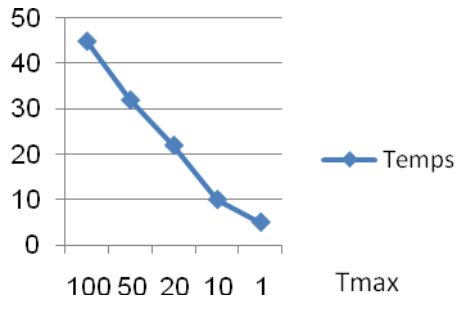


Figure 5.20. Graphe du temps écoulé avant convergence par rapport à la température maximale (temps mn)

5.5 Les critères de performance pour l'unité 3100

Afin d'évaluer les performances de notre système de pilotage ainsi décrit, des critères de performance doivent être mis en place. Ces critères doivent être très représentatifs des objectifs de l'unité 3100. Nous en avons identifié trois.

Comme la majorité des systèmes de production, leur premier objectif est de produire dans les plus cours délais, nous choisissons dès lors comme premier critère de performance

le Cmax

Fin de traitement $Cmax=max_{i=1,n}(C_i)$

C_i est la date de fin d'exécution de toutes les tâches du produit *i*.

Un autre critère est parfois considéré comme corrélé au Cmax mais très important, notamment dans les installations de production couteuses, telles que les installations pétrolières, où on voudra maximiser le gain par rapport à l'investissement, c'est le temps d'occupation des ressources, pompes dans notre contexte, il est noté par :

aIdle

C'est un critère d'efficience(les ressources mises en place ont-elles été nécessaires et suffisantes par rapport aux objectifs ? (Senechal (2004)), il permet de dire si les ressources ont été suffisamment utilisées ou pas. Pour l'estimer, on calcule la moyenne durant laquelle les ressources de production ont été inactives selon la formule suivante :

$$aIdle = \frac{\sum_{i=1}^{Nr} Idle_i}{Nr}$$

Avec : *Nr* est le nombre de ressources et $Idle_i$ est la durée d'inactivité pour la ressource *i*.

aFlowV

Le troisième critère que nous proposons de mesurer est lié à la nature de notre système de production qui est la production à flux. L'unité a pour objectif de produire tout le temps avec un flux du produit stable. Ici, c'est un critère d'efficacité, car il évalue si le flux de sortie change ou pas :

$$aFlowV = \frac{\sum_{i=0}^{n} FlowV_i}{n}$$

Avec *n* est le nombre de fois où le flux est observé.

6 Résultats

Dans cette partie, nous détaillons les résultats obtenus et critiquons leur portée. Auparavant, il est important de tester l'adaptabilité de notre simulateur, c'est-à-dire sa capacité à maintenir son fonctionnement et ses performances face aux changements de l'environnement: augmentation de la quantité des produits à fabriquer. Ensuite nous comparerons les deux modèles d'apprentissage: apprentissage concurrents et apprentissage coopératif. Puis, nous analyserons le comportement de notre système lorsqu'il fait face à deux objectifs: produire le plus et entretenir le plus. Nous analyserons aussi le caractère réactif, c'est-à-dire la capacité du système à maintenir ses performances face aux perturbations notamment le dysfonctionnement des ressources de production. Et enfin, nous comparerons notre modèle à une méta-heuristique les algorithmes génétiques, par rapport la qualité des solutions proposées et la réactivité du système.

6.1 Adaptabilité du système

Afin de tester l'adaptabilité de notre système, nous l'avons appliqué sur des données de l'unité 3100 (détaillées dans l'annexe 5), avec une modification des nombres de produits à fabriquer. Nous commençons par 1 produit et nous terminons par 11 puis nous analysons l'influence des nombres de produits sur la convergence du système.

Le graphe suivant (5.21) montre le nombre d'itérations auquel le système commence à converger par rapport au nombre de produits, selon les critères *Cmax* et *aIdle*.

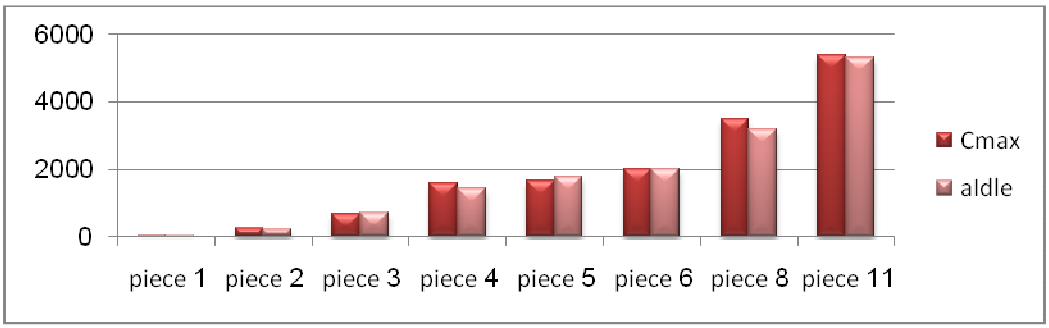


Figure 5.21. La convergence par rapport au nombre de produits

On remarque que le temps de convergence (nombre d'itérations) augmente sensiblement avec le nombre de produits à gérer, ceci était prévisible : plus le système est complexe plus il est difficile d'apprendre une politique optimale, il faut plus d'expériences donc plus de temps.

Nous remarquons aussi qu'il y'a une différence entre le temps qu'il faut pour prendre une décision dans la phase d'exploration et la phase d'exploitation (les définitions des deux phases sont données dans le chapitre 4 section 4.1.2 et annexe 6).

Pour illustrer ceci, nous prenons comme exemple un graphe montrant l'évolution du critère *Cmax* pour une expérience à 4 produits (graphe 5.22)

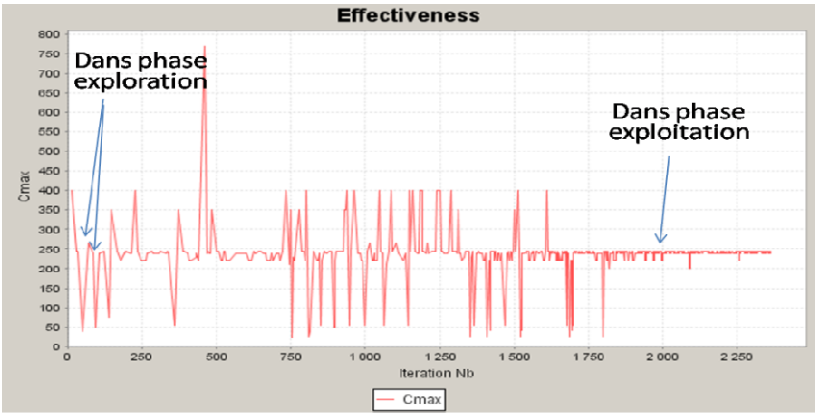


Figure 5.22. Convergence Cmax

Ce graphe montre que la prise de décision dans la phase d'exploitation est plus longue que dans la phase d'exploration. Ceci s'explique par le fait qu'un agent produit peut réitérer sa demande de production plusieurs fois avant que cette dernière ne soit enfin prise en charge par un agent machine. Nous remarquons aussi qu'après les 1500 itérations le système trouve sa stabilité autour d'un *Cmax* jugé optimal par l'algorithme.

6.2 Comparaison entre l'apprentissage concurrent et l'apprentissage coopératif

Nous avons vu dans la section 4.3 comment l'algorithme d'apprentissage des agents ressource a été paramétré. Dans les sections 4.3.2, 4.3.3 et dans la chapitre 4, nous avons montré que l'apprentissage par renforcement dans un système multi-agents peut-être soit *concurrent*, c'est-à-dire que chaque agent n'apprend que pour lui et sa réussite engendre l'échec des autres, ou alors *coopératif*, c'est-à-dire que l'apprentissage des agents est guidé par l'objectif global.

Nous avons alors comparé les deux implémentations :

Cas1, apprentissage concurrentiel : Chaque agent est doté d'un module d'apprentissage MDP avec une table fonction-Q, mais avec une récompense individuelle, donnée par l'expression (2)

Cas2, apprentissage coopératif : ici les agents coopèrent pour atteindre un objectif global, bien sûr, chacun a sa table fonction-Q, mais la fonction récompense est donnée par l'expression (3)

Sur un problème à 5 produits, les résultats sont donnés par les graphes suivants :

(Figure 5.23 5.24 5.25 5.26)

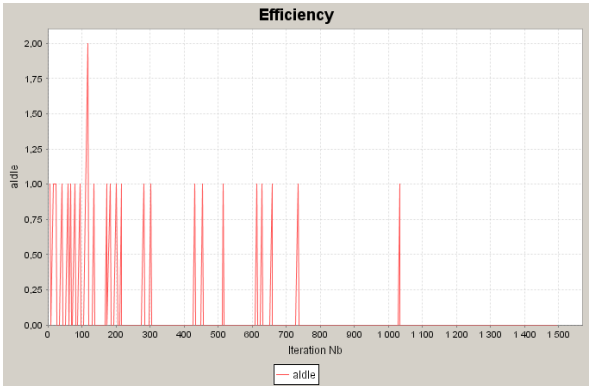


Figure 5.23. Cas 1 aIdle

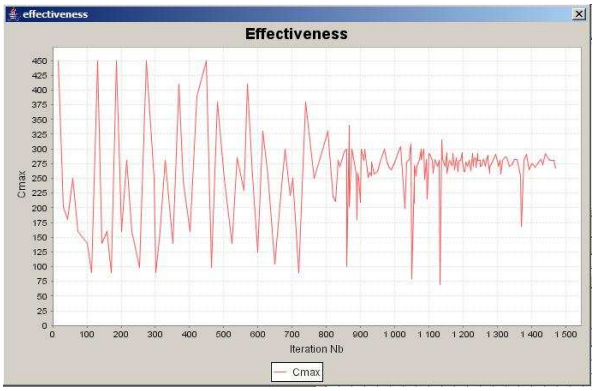


Figure 5.25. Cas 1 Cmax

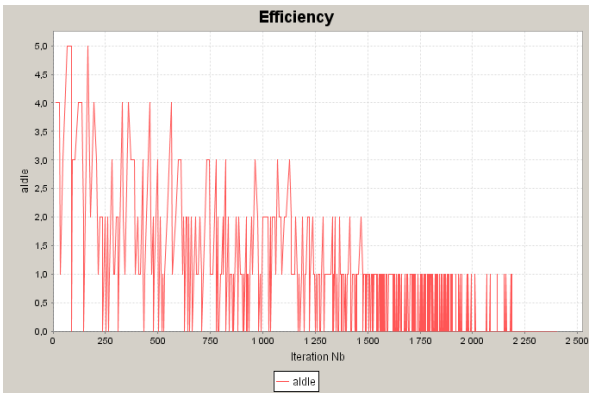


Figure 5.24. Cas 2 aIdle

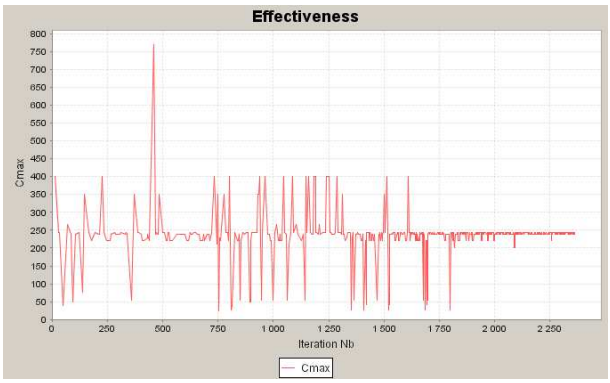


Figure 5.26. Cas 2 Cmax

Nous pouvons alors distinguer que la convergence vers un optimal dans le deuxième cas est plus longue (après 1800 itérations) que le premier cas (après 1000 itérations), ceci s'explique par le fait qu'un apprentissage guidé pour l'objectif global est plus difficile à atteindre que d'apprendre des sous-objectifs locaux.

Cependant, l'optimum atteint par le deuxième cas (Cmax=230, aIdle= 0) est plus intéressant que celui atteint par le premier cas (Cmax=280, aIdle=0). En effet, un apprentissage guidé par l'objectif global, c'est-à-dire surveillé par un critère global, est plus réaliste qu'un apprentissage égoïste de chaque agent, les performances atteintes sont plus intéressantes, donc il est préférable d'adopter un apprentissage coopératif.

6.3 Prise en compte du cadre bi-objectif (pilotage de production et de maintenance)

Nous voulons que notre système de pilotage gère efficacement la production, bien évidement, mais la maintenance aussi.

Un agent ressource doit pouvoir satisfaire les deux objectifs, d'où un apprentissage nécessairement multi-objectifs (chapitre 4). L'agent ressource aura deux modules d'apprentissage un pour la production avec comme fonction de récompense (3) et un module d'apprentissage pour la maintenance avec comme fonction de récompense (4). Sur un problème à 10 produits, nous appliquons un plan de maintenance et observons les variations *aFlowV* car l'objectif recherché est que le volume de production ne change pas au niveau de l'unité et ce, même si des tâches de maintenance sont exécutées.

6.3.1 Résultats en intégrant des tâches de maintenance préventive

Dans un premier lieu, nous intégrons au départ de la simulation le plan de maintenance pour que les tâches de maintenance soient considérées en même temps que les tâches de production (figure 5.15).

Les résultats de simulation sont donnés par les graphes suivants (figure 5.27 5.28):

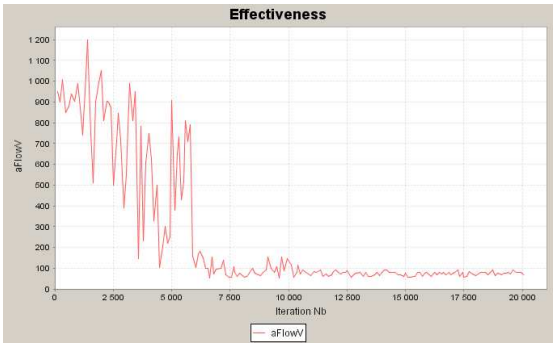


Figure 5.27. Aflow sans maintenance

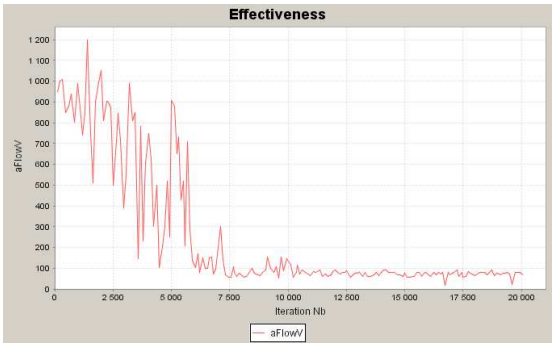


Figure 5.28. Aflow avec maintenance

Ces graphes montrent que même en intégrant les tâches de maintenance la convergence vers le *aFlowV* optimal est atteinte, car sur les deux graphes nous avons une convergence vers l'itération 6000, on peut conclure à partir de cet expérimentation que notre système atteint ses performances même en combinant deux objectifs.

6.3.2 Caractère réactif du système (tâches de maintenance curative)

Afin de tester le caractère réactif du système, nous avons provoqué des perturbations durant l'apprentissage à deux reprises. Cette perturbation est représentée par l'arrivée inopinée de tâches de maintenance curative suite à la panne provoquée sur toutes les ressources (figure 5.29). Nous avons provoqué cette perturbation dans les deux phases de l'apprentissage :

- La première, dans la phase d'*exploration* à l'itération 2000, lorsque l'agent n'a pas encore atteint sa politique optimale
- La deuxième, dans la phase d'*exploitation* à l'itération 15000, quand l'agent est considéré comme suffisamment intelligent pour prendre de bonnes décisions.

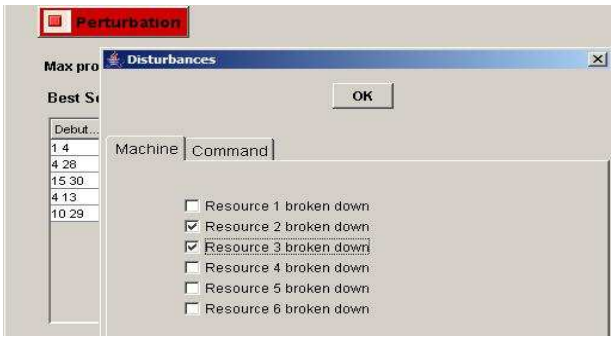


Figure 5.29. Provocation de perturbation

En regardants le graphe (5.30) nous pouvons voir que la perturbation provoquée dans la phase d'exploration vers l'itération 2000 à été difficilement récupérée. Nous remarquons de grandes variations du *aFlowV*. Dans la phase exploitation, la perturbation provoquée a été vite rejetée et le *aFlowV* a rapidement re-convergé vers sa valeur optimale. Nous pouvons conclure alors que selon notre approche, après avoir constitué sa politique décisionnelle, l'agent ressource trouve la meilleur décision à prendre pour maintenir ses performance après l'apparition d'une perturbation.

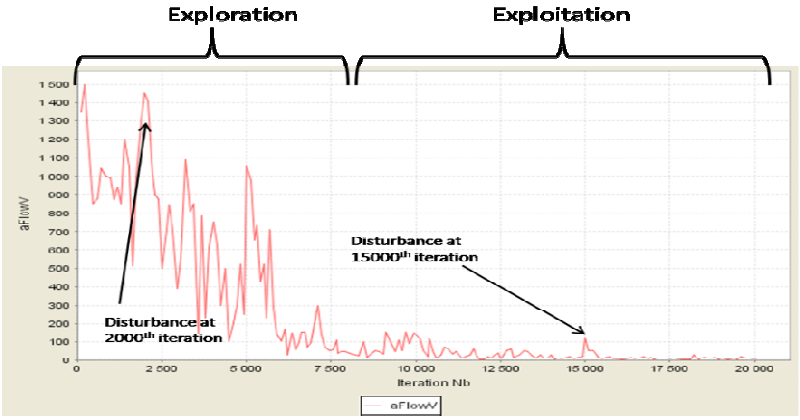


Figure 5.30. Réaction aux perturbations

6.4 Comparaison de notre approche avec une méta-heuristique

Afin de tester la performance de notre approche nous l'avons comparée avec une approche développée à base d'algorithme génétique, une méta-heuristique.

Les algorithmes génétiques sont connus pour leur succès dans la résolution des problèmes d'optimisation à grande complexité, car les algorithmes génétiques trouvent le compromis entre la rapidité de résolution (le temps qu'il faut pour trouver une solution) et la qualité de la solution (Holland, 1975). Les algorithmes génétiques présentent ainsi la caractéristique de trouver des solutions suffisamment optimales par rapport au temps mis pour la trouver, ce qui rend cohérent une comparaison de notre approche avec ceux-ci.

6.4.1 Description de l'algorithme génétique

Les algorithmes génétiques sont considérés par plusieurs chercheurs comme une méthode bien adaptée aux problèmes d'ordonnancement et d'insertion des tâches de maintenance dans un plan de production (Ait si larbi et al. 2008). Son déroulement est montré par le diagramme donné dans la figure 5.31.

Un algorithme génétique est caractérisé par : sa codification, sa fonction objectif et ses opérateurs.

Le codage est le déterminant important de l'efficacité de la méthode. Il signifie la transcription d'un ordonnancement réel en représentation adéquate permettant la réalisation des différents opérateurs génétiques.

Codage de chromosome

En considérant tous les types de tâches simultanément, nous avons codé les chromosomes de la façon suivante :

Code de la tache	Date de début de la tache	Date de fin de la tache	Durée de la tache	……..

La reproduction

La *sélection* est faite en utilisant le tournoi entre individus et l'élitisme par rapport à la fonction objectif, pour pouvoir obtenir le maximum de possibilités tout en gardant les individus les plus intéressants. Nous avons logiquement comme fonction objectif.

Fonction objectif: $f(t)=\min C_{max}$

L'opérateur de *croisement* qui a été utilisé est le croisement multi points (à deux points aléatoires) vu l'avantage qu'il présente en terme de diversité de chromosomes (un croisement entre deux individus génère deux autres individus après avoir combiné leurs gènes).

L'opérateur de *mutation* a été utilisé afin de créer des individus vraiment nouveaux, cet opérateur a été implémenté comme suit :

Prendre deux gènes d'un certain chromosome de la population et faire la permutation entre ces deux gènes, ce qui nous permet d'obtenir un nouvel individu prêt à participer aux prochaines étapes de l'algorithme génétique (sélection, croisement, …).

Notons que la probabilité de mutation que nous avons utilisée est assez faible par rapport à celle utilisée pour le croisement.

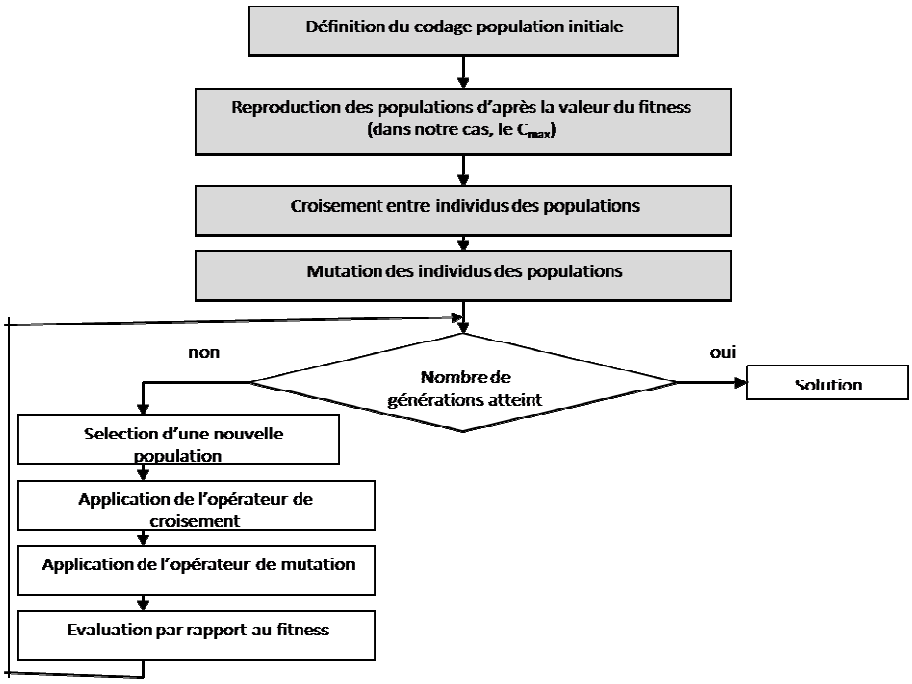


Figure 5.31. Organigramme de l'algorithme génétique

Pour plus de détails sur cet algorithme génétique, consulter Ait Si Larbi (2009), et le PFE Boudaoud (2009).

6.4.2 Résultats de comparaison

Sur l'exemple à 6 produits, nous avons testé les deux approches: notre approche par apprentissage par renforcement RL et l'algorithme génétique GA tel que présenté dans la section précédente. Les résultats sont donnés dans le graphe 5.32. Les solutions trouvées par notre approche et ceux trouvées par l'algorithme génétique sont approximativement de même qualité. Cependant, lorsqu'une perturbation est provoquée sur le système: panne d'une ressource, 2 ressources et plus… nous remarquons que notre approche répond plus rapidement que l'algorithme génétique. En effet, face à une perturbation les données du problème changent et face à cette situation un algorithme génétique lance un réordonnancement ce qui nécessite, approximativement, le même temps que pour la recherche de la première solution. L'on retrouve à ce niveau la discussion sur les approches d'ordonnancement (prévisionnel, dynamique, temps-réel,etc.).

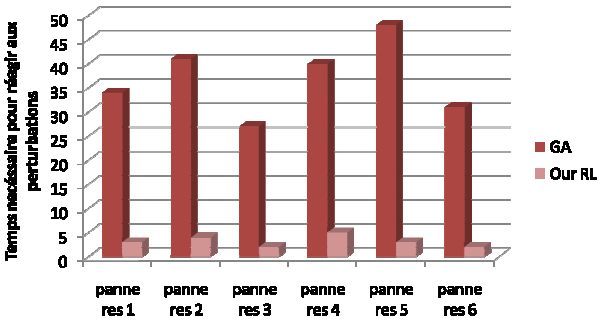


Figure 5.32. Comparaison par rapport au temps de réponse

7 Discussion par rapport aux performances du système (réactivité et adaptabilité)

Les expériences effectuées montrent que le système développé peut élaborer des décisions d'allocations de ressources aux tâches de deux manières:

- D'abord en utilisant un apprentissage *concurrent*, tel que chaque agent ressource cherche à maximiser son gain en essayant d'apprendre à produire le plus et d'avoir le minimum de temps mort.
- Puis, l'apprentissage *coopératif*, ici un agent ressource apprend à produire le plus mais en faisant attention à satisfaire l'objectif global, qui est le raccourcissement de toute la durée du projet.

Notre système utilise aussi un apprentissage multi-objectif qui permet d'établir une politique d'action pour satisfaire deux objectifs qui sont: produire le plus et maintenir le plus.

Les résultats présentés dans ce chapitre montrent que le système est adaptatif à son environnement puisqu'il établit une politique d'allocation tel que en fonction des entrées: demandes de production, demande d'entretien… le système maintient son niveau de performance en restant stable par rapport aux critères vérifiés (durée du projet *Cmax*, temps d'arrêt des ressources *aIdle* et la variation du flux de l'unité *aFlowV*).

Lorsque l'environnement de production change soudainement (par exemple : des pannes qui se produisent) notre système est capable d'adopter de bonnes décisions d'allocation grâce à sa politique d'action apprise, notamment en le comparant avec une heuristique ou méta-heuristique telle que les algorithmes génétiques.

Il faut noter aussi que les graphes obtenus par ces simulations ne sont pas uniques. En effet, un même problème peut générer des graphes différents ceci est dû au caractère aléatoire de la fonction Boltzmann intégrée dans la phase exploration.

Toutefois, face à un problème de grande taille notre système présente des limites, car de nombreuses expériences ont tourné à l'échec avant d'obtenir des résultats pertinents.

Le temps de convergence du système vers une politique optimale peut parfois constituer un problème car pour certains problèmes que nous avons étudiés, ce temps a grandement varié (jusqu'à une trentaine de minutes).

L'adaptabilité est aussi un point qui doit être encore étudié, car un changement permanent d'une consigne peut perturber le système.

8 Conclusion

Ce chapitre a été dédié à la tentative d'application de l'approche développée, basée sur l'apprentissage par renforcement décentralisé sur les agents et l'apprentissage multi-objectif. Nous avons montré les détails de la modélisation en tenant en compte des particularités du contexte industriel auquel nous nous intéressons: l'industrie pétrolière et plus particulièrement l'unité de fabrication des huiles finies de la raffinerie d'Arzew.

Nous avons d'abord présenté le simulateur et son environnement (données en entrées, paramètres, données en sorties).

Nous avons ensuite mené des expérimentations pour paramétrer les coefficients de l'algorithme d'apprentissage.

Les résultats obtenus ont alors été présentés au travers de plusieurs expérimentations :

- Une expérimentation pour déterminer lequel de l'apprentissage concurrent ou coopératif est le plus intéressant.
- Une expérimentation pour tester l'apprentissage multi-objectif.
- Et enfin une expérimentation pour comparer notre approche avec une autre approche basée sur un algorithme génétique. Cette comparaison a montré que notre système, en apprenant, a acquis un caractère réactif qui lui permet de répondre rapidement aux fluctuations de l'environnement, ceci grâce à la politique décisionnelle qu'il a pu construire.

Nous avons ainsi montré que notre système était capable de réactivité tout en s'adaptant aux évolutions de l'environnement et améliorant ses performances faisant face à plusieurs objectifs notamment augmenter la cadence de production en raccourcissant les délais et entretenir le plus possible les ressources de production.

Conclusion générale et perspectives

Ce travail s'inscrit dans le cadre de l'intelligence artificielle distribuée dédiée au pilotage de production. L'environnement de production actuel pousse les systèmes de production à être plus flexibles, réactifs et adaptatifs aux exigences du marché international en même temps qu'aux exigences internes au système de production lui-même. Nous nous sommes intéressés à un type particulier de systèmes de production qui sont les systèmes à flux continu issus de l'industrie pétrolière.

L'environnement de l'industrie pétrolière est caractérisé par sa complexité et son intolérance aux risques. Dans ce type d'industrie, la maintenance constitue l'une des plus importantes fonctions du système de pilotage et il est nécessaire de la considérer dans les systèmes de pilotage qui sont conçus.

Pour développer un tel système de pilotage, nous nous sommes basés sur une approche multi-agents qui a déjà fait ses preuves dans le domaine de la gestion, le contrôle de production et la résolution des problèmes décisionnels complexes. Les décisions prises par le système sont le résultat d'un travail de groupe d'agents dont l'organisation est en développement continuel. Une originalité de notre travail se situe dans la capacité de notre système de pilotage à assurer à la fois une efficacité opérationnelle en terme de réactivité, mais aussi d'intégrer des mécanismes d'amélioration constante de la performance en tenant compte à la fois des contraintes de production et des contraintes de maintenance.

Nous avons introduit pour ce faire la notion d'apprentissage au niveau des agents dans leurs différentes fonctions ou rôles (répondre aux requêtes (aléas), s'auto-organiser, planifier…) sans atténuer la qualité de réactivité temps-réel du système. Pour cela, nous avons choisi l'utilisation de l'apprentissage par renforcement, une technique très réactive qui consiste en la recherche de l'action la plus efficace à entreprendre dans chacun des états du système.

Les résultats obtenus sur un cas applicatif réel sont très prometteurs. Nous avons en effet montré que notre système de pilotage est capable de réagir face à certains aléas mais aussi de s'améliorer au fil du temps et donc, de s'adapter. Après perfectionnement, nous pensons que ce système peut être implémenté sur une couche de commande, tel que le DCS, pour aider à piloter une installation pétrolière en proposant au responsable un ordonnancement réactif performant. Les principaux résultats scientifiques obtenus sont les suivants :

- Nous avons proposé une approche de pilotage décentralisée basée sur les systèmes multi-agent qui offre un potentiel important en termes d'auto-adaptation et réactivité.
- L'intégration de l'apprentissage par renforcement permet aux agents d'être « intelligents » et adaptatifs à leur environnement.
- L'apprentissage décentralisé où chaque agent est autonome dans son apprentissage et titulaire de sa propre table de valeurs d'utilité, permet au système d'être adaptatif avec une durée de construction de politique optimale réduite.

- Lorsqu'un agent a plusieurs objectifs à atteindre une approche d'apprentissage par renforcement multi-objectif peut être utilisée.

Nos travaux ont fait l'objet de plusieurs publications :

- Aissani et al., (2008b), où nous avons expérimenté l'apprentissage par renforcement pour le pilotage d'un atelier JobShop pour montrer la faisabilité de la chose.
- Ensuite, Aissani et al, (2008a), où nous avons intégré la notion de multi-objectif, nous avons montré qu'un modèle d'apprentissage modulaire pouvait atteindre deux objectifs de performance à la fois: efficacité en produisant dans les plus courts délais et efficience en raccourcissant le temps d'arrêt des ressources de production.
- Dans Aissani et al, (2009a), nous nous sommes intéressé à un autre type de production, la production à flux continu en intégrant toujours plusieurs objectifs: produire le plus dans les plus courts délais et maintenir le plus.
- Et enfin, nous avons entamé une perspective, c'est de tester notre approche sur un nouveau modèle d'entreprise, c'est les entreprises multi-sites (Aissani et al., (2009b).

Notre étude a fait appel à de nombreux champs de recherche (gestion de production, pilotage des systèmes continus et discrets, ordonnancement des systèmes, systèmes multi-agents, techniques d'apprentissage, techniques d'optimisation, simulation…), ce qui a nécessité une étude assez précise de nombreux états de l'art qui ont été présentés dans cette thèse. Cette étude multidisciplinaire a été très féconde en termes d'idées sur lesquelles nous avons travaillé, mais a également mis en perspectives de nombreuses activités de recherche à mener dans les années à venir. Nous en listons ici quelques unes.

D'abord la scalabilité du système. Des expériences ont montré que face à un problème complexe composé d'un grand nombre d'agents, la convergence du système vers une politique optimale est assez difficile à obtenir. Un premier challenge est la réduction de la complexité liée à la représentation de la tâche (la complexité de l'état et le nombre d'actions possibles) et le volume de données nécessaire à l'apprentissage (les tables des valeurs Q).

Ensuite, l'algorithme d'apprentissage par renforcement multi-objectif que nous avons utilisé "Nearest neighbor" (chapitre 4) a montré ses limites face à la complexité du problème (de nombreuses expériences ont tourné à l'échec avant d'obtenir des résultats pertinents). Sur ce sujet, Pr Vamplew (Vamplew et al; 2008) nous a proposé de travailler ensemble sur l'amélioration de cet algorithme en donnant peut-être des coefficients dynamiques aux objectifs pour les adapter aux différentes situations. Une thèse est déjà lancée dans ce sens.

Un problème majeur dans les systèmes distribués apprenants est la convergence des politiques optimales lorsque chaque décision prise par un agent influe sur les résultats obtenus par les autres agents. Dans notre thèse, nos expérimentations n'ont pas clairement mis en évidence d'absence de convergence, mais ce point reste à étudier (voir la théorie des jeux de Markov présentée en annexe 6).

Un problème important également concerne les dérives lentes ou rapides de l'environnement (évolution du type d'huile utilisé à travers le monde, etc.) et la capacité du système apprenant à s'adapter à ses dérives plus ou moins rapides.

Nous avons supposé implicitement l'absence de ces dérives, mais il sera nécessaire de tester et de caractériser la capacité de notre système de pilotage à s'adapter à l'évolution de ces « points de fonctionnement ».

Dans notre proposition, c'est l'agent ressource qui décide et apprend, nous voulons tester maintenant le concept de pilotage par le produit en intégrant des capacités de décision et d'apprentissage au sein des agents produit (Sallez et al, 2009).

Un dernier point, en analysant l'architecture du module décisionnel de l'agent ressource, nous pouvons remarquer qu'il est constitué encore de deux autres modules décisionnels, ce qui semble être une récurrence. Ceci nous fait penser à la possibilité de modéliser ces agents apprenants en Holons (Sallez et al, 2009).

Concernant les aspects applicatifs, nous voulons généraliser cette expérience sur différents types de système de production, à commencer par d'autres unités au niveau de la raffinerie, notamment l'unité de distillation (voir annexe 1) qui a fait déjà l'objet d'une petite expérimentation dans Aissani et al (2009a).

Puis nous proposons de poursuivre cette généralisation à d'autres types de systèmes, tels que les systèmes de production de services (entreprise de Transit: entreprise qui s'occupe du transport de la marchandise exportée ou importée, thèse de Melle Chalabi) et enfin, aux chaines logistiques ou aux entreprise multi-sites (thèse de Melle AitSiLarbi; AitSiLarbi et al; 2008).

L'ensemble des perspectives de recherche présentées ici seront envisagées dans le cadre de travaux de coopération future entre les deux équipes de recherche (dont les compétences sont complémentaires) au travers d'autres cotutelles de thèses et projets de recherche.

Références et Bibliographie

Agre. P. E, “The dynamic structure of everyday life”. PHD thesis, Massachusetts Inst of Tech Cambridge (MIT) artificial Intelligence Lab, 1988

Aissani. N, D. Trentesaux and B. Beldjilali, “efficient and effective reactive scheduling of manufacturing system using SARSA-Multi-objective-agents”, MOSIM’08: 7th int. Conf. On Modelisation & Simulation, Editors : Lamouri. S Et Thomas. A, Artiba. A Et Vernadat. F, Published by Tec&Doc -Lavoisier, pp 698-707, 2008a.

Aissani .N, D. Trenteseaux, B. Beldjilali, « Use of Machine Learning for Continuous improvement of the Real Time Manufacturing control system performances”, IJISE: International Journal of Industrial System Engineering, Vol 3, No 4, ISSN 1748-5037, pp 474-497, 2008b

Aissani. N, D. Trenteseaux, B. Beldjilali , "Dynamic Scheduling of Maintenance Tasks in the Petroleum Industry: a Reinforcement Approach" , EAAI: International Scientific Journal Engineering Applications of Artificial Intelligence , Vol 22 , ISSN: 0952-1976, pp. 1089-1103, 2009a (Impact factor: 1.397)

Aissani. N, B. Beldjilali, D. Trentesaux, “Multi-agent reinforcement learning for adaptive scheduling: application to multi-site company», INCOM’09: 13th IFAC Symposium on Information Control Problems in Manufacturing, Moscow, June 3-5, pp. 1107-1112, 2009b

Ait Si Larbi. E. Y, N. Aissani, B. Beldjilali, “Un Modèle de Planification pour les Entreprises Multi Sites basé sur les Systèmes Multi Agents et les Algorithmes Génétiques», Proc. MCSEAI’08: 10th Maghrebian Conference on Information Technologies. Avril 28-30, Oran, Algeria, pp.506- 511, 2008

Ait Si Larbi. E. Y, “proposition d’un modèle pour la planification de production des entreprises multi sites », thèse de Magister, Université d’Oran, Juin 2009

Aloulou, M., and Portmann, M.C. “A genetic algorithm to achieve scheduling flexibility for a single machine problem”, RAIRO- operation research 19 p, 2002.

Austin. J, “How to do things with words”, 1962

Barto. A. G, Bradtke. S. J, Singh. S. P, “Real-time learning and control using asynchronous dynamic programming”, Technical report, Computer science department, University of Massachusetts, pp 57-91, 1991

Beasleya. J. E, and B. Cao, “A dynamic programming based algorithm for the crew scheduling problem”, Computers & Operations Research, Vol 25, ISSN 7-8, pp 567-582, July 1998

Becker. R, Zilberstein. S, Lesser. V, and. Goldman. C. V. « Transition-independent decentralized markov decision processes ». Proc. the Second International Conference on Autonomous Agents and Multi-agent Systems, pp 41–48, Melbourne, Australia, 2003.

Bellman, R. “Dynamic Programming”, Ed Princeton University Press, p.83, 1957

Bellmann R. et Drefus S.E. “Applied dynamic programming”, Princeton University Press, 1974

Berger T., Sallez Y., Tahon C. "Dynamic FMS scheduling using ant-based control". Proc. 5th International Confrence on integrated design and Manufacturing in Mechanical Enginering, Bath, R-U, Angleterre, avril, 2004.

Bernstein. D, R. Givan, and N. Immerman, and, S. Zilberstein, "The Complexity of Decentralized Control of Markov Decision Processes". Mathematics of Operations Research, Nu 27, Vol 4, pp 819 – 840, 2002

Blazewicz J., Ecker K., Pesch E. Schmidt G. & Weglarz J., "Scheduling computer and manufacturing processes", Springer-Verlag, 1996.

Bousbia, S. et Trentesaux, D. "Self-organization in distributed manufacturing control: state-of-the-art and future trends", in A. El Kamel, K. Mellouli and P. Borne (Eds). IEEE International Conference on Systems, Man and Cybernetics, Vol. 5, CD-Rom, ISBN: 2-9512309-4-X, (Hammamet, Tunisie, Octobre 2002), paper #WA1L1, p.6, 2002

Bousbia, S.,Trentesaux, D., "Towards an intelligent heterarchical manufacturing control system for continuous improvement of manufacturing systems performances". In: Fifth International Conference on Integrated Designand Manufacturing inMechanicalEngineering—IDMME'2004, University of Bath, UK, April2004, Paper#123,10p. 2004.

Buffet. O, « Une double approche modulaire de l'apprentissage par renforcement pour des agents intelligents adaptatifs ». Thèse UFR STEMIA. Nancy : Université Henri Poincaré-Nancy1, 2003

Burlat, P, X. Boucher, « Pilotage distribué des groupements d'entreprises - Modélisation et perspectives », Journal Européen des Systèmes Automatisés, Vol 35, No7-8 , pp.991-1018, 2001.

BRAIN, http://polaris.ing.unimo.it/MOON/BRAIN/index.html

Briand. C, H.T. La, « Une procédure par séparation et évaluation progressive pour l'ordonnancement robuste de problème à une machine », RVIF,03, premiere conference internationale associant chercheurs vietnamiens et francophones en informatique, 2003

Beer. R.D, Kacmarcik. G.J, Ritzmann. R.E," A model of distributed sensorimotor control in the cockroach escape turn", In R.P. Lippmann, J.E.moody et D.S.Tourezky (eds), Neural information processing systems, vol. 3, pp. 436-42. 1991

Brucker P., Scheduling algorithms, Spriger-Verlag, 1998.

Cabri. G, L. Leonardi, F. Zambonelli, "Separation of Concerns in Agent Applications by Roles", Proc. the 2nd International Workshop on Aspect Oriented Programming for Distributed Computing Systems (AOPDCS 2002), Wien, July 2002

Cabri. G, L. Ferrari, L. Leonardi, "BRAIN: a Framework for Flexible Role-based Interactions in Multiagent Systems", Proc. First European Workshop in Multi Agent Systems (EUMAS), Oxford, UK, December 2003

Cabri. G, L. Ferrari, L. Leonardi, “Agent Role-based Collaboration and Coordination: a Survey About Existing Approaches”, 2004 IEEE international conference on systems, man & cybernetics, The Hague, Netherlands, 10-13 october 2004

Cafaro. D. C, Cerdà. J, “Dynamic scheduling of multiproduct pipelines with multiple delivery due dates”, Computers and Chemical Engineering, Vol 32, pp. 728–753, 2008

Cavory. G, R. Dupas and G. Goncalves, “A genetic approach to the scheduling of preventive maintenance tasks on a single product manufacturing production line”, Int. J. Production Economics, Vol 74, pp 135-146, 2001.

Cicirello, V. A., et Smith, S. F. “Ant colony control for autonomous decentralized shop floor routing”. Proc. ISADS-2001, Fifth International Symposium on Autonomous Decentralized Systems. IEEE Computer Society Press, 2001.

Chaari. T, "Un algorithme génétique pour l'ordonnancement robuste: application au problème du flow shop hybride", thèse de doctorat université de valenciennes et du hainaut cambrésis et de la faculté des sciences economiques et de gestion de sfax, 2010

Chelbi, A., Ait-Kadi, D., « Joint optimal buffer inventory and preventive maintenance strategy for a randomly failing production unit”. Journal of Decision Systems, No 12, pp 21–30, 2003.

Chen, C.C., aet Yih, Y. “ Identifying attributes for knowledge-based development in dynamic scheduling environments”. International Journal of Production Research, Vol 34, No 6, pp 1739-1755, 1996.

Csaji B. C and L. Monostori, “Adaptive algorithms in distributed resource allocation”. Proc. the 6th International Workshop on Emergent Synthesis (IWES-06), August 18–19, The University of Tokyo, Japan, pp. 69-75, 2006

Csaji, B. Cs.; Monostori, L.: “Stochastic Reactive Production Scheduling by Multi-Agent Based Asynchronous Approximate Dynamic Programming”, Lecture Notes in Computer Science; 3690: Lecture Notes in Artificial Intelligence, Proceedings of the 4th International Central and Eastern European Conference on Multi-Agent Systems (CEEMAS), September 15–17, Budapest, Hungary, pp. 388–397, 2005.

Davenport Andrew J. et Beck J. Christopher , “A survey of techniques for scheduling with uncertainty “, Technical report, IBM and Ilog. 2000.

Degres. L, Pierreval. H, Caux. C, “Simulation of steel industry using system dynamics”, Proc. IFAC Information Control Problems in Manufacturing, Brazil, pp 461-469, 2004

Demmou, R. « Étude de familles remarquables d’ordonnancements en vue d’une aide à la décision », Thèse de doctorat, Université Paul Sabatier, Toulouse 1977

Dossier spécial Algérie (2ème partie), Pétrole et techniques, No.440, 2002.

Dunne, T., and Mu, X. “Investment spikes and uncertainty in the petroleum refining industry”, Paper provided by Federal Reserve Bank of Cleveland in its series, Working Paper, No 0805, 2008.

Elkhyari, A. » Outil d'aide à la décision pour des problèmes d'ordonnancement dynamique », Thèse de Doctorat, université de Nantes, 2003.

Erschler. J, G. Fontan, C. Merce, F. Roubellat, "A new dominance concept in scheduling n jobs on a single machine with ready times and due dates," Journal of Operation Research, Vol.1, pp. 114-127, 1983.

Erschler J., Huguet M.J., de Terssac G., « Décision distribuée en gestion de production : exploitation et régulation de l'autonomie », Concepts et outils pour les systèmes de production, coord. J.C. Hennet, éd. Cepadues, Toulouse, pp. 109-131, 1997.

Esswein. C, Artigue. C, Billaut, J.C, « Maximisation of solution flexibility for robust shop scheduling », Technical report, LIA, Report 324, 2002.

Esswein. C, Artigue. C, Billaut, J.C, « Maximiser la fléxibilité sur une machine et dans un job-shop: ordonnancements de groupes », Proc. 5ème congrés de la société française de Recherche Opérationnelle et d'Aide à la Décision, pp. 142-143, Avignon, France 2003.

Ferber. J, « Les Systèmes Multi-Agent - Vers une Intelligence Collective », InterEditions, 1995.

Ferber. J, Gutknecht.O, « Un méta-modèle organisationnel pour l'analyse, la conception et l'execution de systèmes multi-agents », Proc. ICMAS, 1998

Fowler. M, "Dealing with Roles", http://martinfowler.com/apsupp/roles.pdf, 1997.

Garey M.R., Johnson D.S., « Computers and intractability: a guide to the theory of NP-completeness », W.H. Freeman and Company, New-York, 1979

Giard, V., « Gestion de la production et des flux », 3e édition, Economica. 1988.

Ghédira. K, » Logistique de la production: Approches de modélisation et de résolution », Edition TECHNIP, 2006

Glover, F. « Future paths for integer programming and links to artificial intelligence», Computers and Operations Research, 1986.

Günther H O, Dirk C. Mattfeld and Leena Suhl , "Supply Chain Management and Advanced Planning Systems: A Tutorial", in Supply Chain Management und Logistik, hysica-Verlag HD, DOI 10.1007/b138875, ISBN 978-3-7908-1625-9 (Online), pp 3-40, 2005.

Hadeli, Valckenaers, P., Kollingbaum, M., Van Brussel, H. "Multi-Agent Coordination and Control Using Stigmergy". Computers in Industry, Vol 53, pp 75–96, 2004.

Haruno, M., Kawato,M., 2006. "Heterarchical reinforcement-learning model for Integration of multiple cortico-striatal loops", FMRI examination instimulus-action-reward association learning. Neural Networks, vol 19 (SpecialIssue), pp. 1242–1254, 2006.

Hirsh. B, Fisher. M, Ghidini. C, "Programming Group Computations", Proc. First European Workshop on Multi-Agent System (EUMAS), Oxford, UK, December 2003.

Holland J.H., « Adaptation in natural and artificial systems», Technical report, University of Michigan, Ann Arbor, 1975.

Hu. J et M. Wellman. “Multiagent reinforcement learning : theoretical framework and an algorithm”. Proc. the 15th International Conference on Machine Learning (ICML’98), pp 242–250, 1998.

Hu. J et M. Wellman. “Online learning about other agents in a dynamic multiagent system”. Proc. the 2nd International Conference on Autonomous Agents (Agents’98), pp 239–246, 1998.

Humphrys, M. “Action Selection Methods using Reinforcement Learning”. Proc. the Fourth International Conference on Simulation of Adaptive Behavior, pp 135–144.1996.

Huyet. A. L, Paris. J. L , « Apprentissage et optimisation conjoints : extraction de connaissances pertinentes sur les systèmes de production », p 599, EGC 2004

International Organization for Standards, The Ottawa Report on Reference Models for Manufacturing Standards, ISO TC184/SC5/WG1 N51, Version 1.1, 1986.

JADE, http://jade.tilab.com/

Joly. M, L.F.L.Moro et J.M.Pinto, « Planning And Scheduling For Petroleum Refineries Using Mathematical Programming », Vol. 19, No. 02, pp. 207 - 228, April - June 2002

Johannet. A et Sarda. I, “Behaviour learning by reward-penality algorithm: from gait learning to obstacle avoidance by neural networks”. Proc. International conference artificial neural nesard genetic algorithms, France, pp. 465-467, 1995

Karlsson. J, “Learning to solve Multiple Goals”, Phd Thesis, University of Rochester, New York, 1997

Kenne. JP, Boukas. EK.,”Hierarchical control of production and maintenance rates in manufacturing systems”. J Qual Maint Eng ,Vol 9, No 1,pp. 66–82, 2003.

Kenne. JP, Gharbib. “A, Stohastic optimal production control problem with corrective maintenance”, Computers & Industrial Engineering, Vol 46, pp. 865–875, 2004.

Kirkpatrick S., Gelatt, C.D. et Vecchi M.P., «Optimization by simulated annealing ». Science, 1983.

Kim J.Y., Kim T.G. « A Heterogeneous Simulation Framework Based on DEVS Bus and High Level Architecture », Proc. the 1998 Winter Simulation Conference, December 13-16, 1998, Washington, DC, USA, Vol. 1, pp. 13-16, 1998.

Koestler, A, “Ghost in the Machine”, The Danube Edition, Hardcover, 1967.

Kouvelis, P. and Yu, G. “Robust discrete optimisation and its applications”, Kluwer Academic Publisher, London, 1997.

Krothapalli, N. et A. Deshmukh, « Effects of Negotiation Mechanisms on Performance of Agent–Based Manufacturing Systems», Flexible Automation and Integrated Manufacturing, 1997.

Kutanoglu, E. et Sabuncuoglu, I. “An analysis of heuristics in a dynamic job shop with weighted tardiness objectives”, International Journal of Production Research, Vol 37, No 1, pp 165-187, 1999.

Lau. R, « L'apprentissage individuel et collectif des agents d'ordonnancement pour l'atelier flexible », Thèse de Doctorat à Institut national des sciences appliquées de Lyon, Villeurbanne, France, 1997.

Lawler E.L. et Wood D.E. “Branch and bound methods : A survey” , Operations Research, Vol 14, pp 699-719, 1966

Leitao. P, Restivo. F, “A holonic approach to dynamic manufacturing scheduling, Robotics and Computer-Integrated Manufacturing”, No 24, pp 625–634, 2008.

Le Quéré Y., Sevaux M., Tahon C., Trentesaux D., “Reactive scheduling of complex system maintenance in a cooperative environment with communication times", IEEE transactions on Systems, Man & Cybernetics, part C : applications & reviews, Vol. 33, No 2, pp.225-234, 2003.

Le Quéré. Y, M. Sevaux, C. Tahon, and D. Trentesaux. Modèle de coopération d'un processus de ré-ordonnancement distribué ». Proc. the Automatic control doctoral days (Journées doctorales d'automatique), pp 401-406, LAMIH, Valenciennes, France, 25-27 June 2003.

Littman. M. “Markov games as a framework for multi-agent reinforcement learning”. Proc. 11th International Conference on Machine Learning (ICML’94), San Fransisco, CA, 1994.

Lopez P. & Roubellat F., Ordonnancement de la production. Chapitres 1 &2. Edition Hermès Science, 2001.

MADKIT, Http://www.madkit.org/downloads

Maione B., Naso D., « Evolutionary adaptation of dispatching agents in heterarchical manufacturing systems », international journal of production research, Vol. 39, No 7, pp. 1481-1503, 2001.

Masuchun, R., et Ferrell, W.G. , “Dynamic rescheduling with stability”, Proc. the 5th Asian Control Conference, pp.1886-1894, 2004.

Mjolsness. E, Decoste D, "Machine Learning for Science: State of the Art and Future Prospects", Machine Learning Systems Group, Jet Propulsion Laboratory/California Institute of Technology, Pasadena, CA, 91109, USA. Science Vol 293 14 September 2001

NF X060-10, NF X 060-11, Norme AFNOR, http://www.afnor.fr/portail.asp.

OMG. 2005. OMG Unified Modeling Language Specification,m Version 2.0.

Omicini. A, A. Ricci, S. Ossowski, “Rethinking MAS Infrastructure based on Activity Theory”, First European Workshop on Multi-Agent System (EUMAS), Oxford, UK, December 2003.

Pannequin, R, Thomas, A, Morel, G, « Proposition d'un Environnement 2 avril 2008 – Paris-France

Groupe PDMIA, "Processus décisionnels de Markov en intelligence artificielle Tome 1: principes généraux et applications », 2004.

Pierreval H. et Ralambondrainy H., "A simulation and learning technique for generating knowledge about manufacturing systems behavior", Journal of Operational Research Society, No 41, Vol 6, pp. 461-474, 1990.

Pinedo, M. "Scheduling, theory, algorithms, and systems". Prentice Hall, Englewood Cliffs, 1995.

Priore, P., Garcia, D.D., and Quesada, I.F. " Manufacturing systems scheduling through machine learning", Neural Computation, NC'98, Vienna, Austria, pp 914 -917, 1998.

Priore, P., De la Fuente, D., Gomez, A., and Puente, J. « Review of machine learning in dynamic scheduling of flexible manufacturing systems". Artificial Intelligence for Engineering Design Analysis and Manufacturing, Vol 15, No 3, pp 251-263, 2001.

Prabhu,V.V," Stability and fault adaptation in distributed control of heterarchical manufacturing jobshops". IEEE Transactions on Robotics and Automation, Vol 19, No 1, pp. 142–147, 2003.

Pujo. P et D. Brun-Picard, « Pilotage sans plan prévisionnel ni ordonnancement préalable », Méthodes du pilotage des systèmes de production, Hermès, pp 129- 162, 2002.

Pujo, P et Kieffer, J.P, « Fondements du pilotage des systèmes de production », Hermes Science Publications, 2002.

Putterman M. L., "Markov Decision Proceses". Discrete stochastic dynamic programming. Wieley-Interscience, New York 1994.

Rabased. A - Loudcher .S, « Contributions à l'extraction automatique de connaissances : application à l'analyse clinique de la marche », Thése de Doctorat, L'universite Claude Bernard - Lyon I - 16 décembre 1996

Rajendran, C., et Holthaus, O. , "A comparative study of dispatching rules in dynamic flow shops and job shops", European Journal of Operational Research, Vol 116, No 1, pp 156-170, 1999.

Retour, D., Bouche, M., and Plauchu, V. « Où va la maintenance industrielle », Problèmes Économiques, No 2.159, pp 7-13, 1990.

Sallez Y., Trentesaux D., Berger T., Tahon C., "Product-based and resource-based heterarchical approaches for dynamic FMS scheduling", Computer and Industrial Engineering, Vol. 46, pp. 611–623, 2004.

Sallez Y., Berger T., Trentesaux D. « A stigmergic approach for dynamic routing of active products in FMS". Computers in industry, DOI: 10.1016/j.compind.2008.12.002, Vol. 60, No 3, pp. 204-216, 2009.

Schneider. J .G, Boyan. J. A, Moore. A. W, "Value Function based Production scheduling", Proc. ICML The Fifteenth International Conference on Machine Learning. July 24-27, in Madison, Wisconsin. 1998.

Searle. J – “Speech acts”, 1969.

Senechal. O, « Pilotage des systèmes de production vers la performance globale, Habilitation à diriger des recherches, l’université de valenciennes et du Hainaut cambresis, Septembre 2004

Sevaux, M., and Sorensen, K. S. “A genetic algorithm for robust schedules in a just-in-time environment”. Technical Report LAMIH/SP-2003-1, University of Valenciennes, 2002.

Singh S. P., Jaakkola T., Littman M. L., Szepesvari C., « Convergence Results for Single-Step On-Policy Reinforcement Learning Algorithms », Machine Learning, Vol. 38, No 3, pp. 287–308, 2000.

Simon. H, “Why should machines learn ?”. In Machine Learning : an artificial intelligence approach, volume 1. 1983.

Smith. R. G., “ The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver”, IEEE Transactions On Computers, Vol. C-29, No. 12, pp 1104-1113, 1980.

Sohier, C., Denis, B., Bourdet, P, “Applying eco problem solving to the control of an adaptive manufacturing cell”, Proc. of CESA'96 IMACS Multiconference, Lille, France, pp. 710-715, 1996.

Sprague. N, Ballard D., “Multiple Goal Reinforcement Learning with Modular Sarsa(0)”, Proc. International Joint Conference on Artificial Intelligence IJCAI 2003

Suresh, V., Chaudhuri, D. “Dynamic scheduling–A review”. International Journal of Production Economics 32, pp 53–63, 1993.

Sutton, R.S. and Barto, A.G. “ Reinforcement Learning: An Introduction”, Cambridge, MA: MIT Press, 1998.

Takadama. K et Fujita. H, (2004), "Lessons Learned from Comparison Between Q-learning and Sarsa Agents in Bargaining Game", North American Association for Computational Social and Organizational Science NAACSOS, June 27 – 29, Pittsburgh PA, 2004.

Taghezout, N, Zaraté, P. “Negotiation Process for Multi-Agents DSS for Manufacturing System. Dans : Collaborative Decision Making: Perspectives and Challenges”. Pascale Zaraté, Jean Pierre Belaud, Guy Camilleri, Franck Ravat (Eds.), IOS Press, pp. 49-60, Vol. 176, Frontiers in Artificial Intelligence and Applications, juillet 2008.

Trentesaux D., Dindeleux R., Tahon C., “A MultiCriteria Decision Support System for Dynamic task Allocation in a Distributed Production Activity Control Structure”, Int. Journal of Computer Integrated Manufacturing, Vol. 11, No 1, pp. 3-17, 1998.

Trentesaux D., Pesin P., Tahon C., “Distributed artificial intelligence for FMS scheduling, control and design support”, J. of Intelligent Manufacturing, Vol. 11, No 6, pp. 573-589, 2000.

Trentesaux, D, J. P. Campagne et J. Erschler, “A generic design framework for decentralized control: the DMU model”, 2nd IFAC/IFIP/IEEE Conf. On Management and Control of Production

and Logistics, MCPL 2000 (Grenoble, juillet 2000), Z. Binder (ed), Pergamon, ISBN 0-08-043621 8, 1st edition, Oxford, Elsevier Science Ltd., Vol. 3, 2001, pp. 1021-1026. 2000.

Trentesaux, D, "Pilotage hétérarchique des systèmes de production". Habilitation à diriger des recherches, Université de Valenciennes et du Hainaut-Cambrésis, 19 décembre 2002.

Trentesaux D., « Les systèmes de pilotage hétérarchiques: innovations réelles ou modèles stériles ? », Journal Européen des Systèmes Automatisés, Vol. 41, No 9-10, pp. 1165-1202, 2007.

Querrec R., Tarot S., Chevaillier P., Tisseau J., « Simulation d'une cellule de production. Utilisation d'un modèle à base d'agents contrôlés par réseaux de Petri », Colloque de recherche doctorale AGIS'97 – Automatique, Génie informatique, Image, Signal, Angers, pp. 209-214, 1997.

Ubayashi, T. Tamai, "RoleEP: role based evolutionary programming for cooperative mobile agent applications", Proc. International Symposium on Principles of Software Evolution, 2000.

Vacher. J.P, « Un système adaptatif par agents avec utilisation des algorithmes génétiques multi-objectifs : Application à l'ordonnancement d'atelier de type job-shop N x M », Thèse de Doctorat de l'université Le Havre, 2000.

Vamplew. P, Yearwood. J, Dazeley. R, et Berry. A, "On the Limitations of Scalarisation for Multi-objective Reinforcement Learning of Pareto Fronts", W. Wobcke and M. Zhang (Eds.): AI 2008, LNAI 5360, pp. 372–378, 2008, Springer-Verlag Berlin Heidelberg 2008

Van Brussel H., Wyns J., Valckenaers P., Bongaerts L. et Peeters P., « Reference architecture for holonic manufacturing systems: PROSA », Computers in industry, Vol. 37, pp. 255-274, 1998.

Van der Duyn Schouen, F.A., Vanneste, S.G., "Maintenance optimization of a production system with buffer capacity". European Journal of Operational Research, Vol 82, No 2, pp 323–338, 1995.

Vaquero, T.S ; Sette, F. ; Silva, J. R. ; Beck, J.C., "Planning & Scheduling of Crude Oil Distribution in a Petroleum Plant". Proc. 19th int. Conference on Automated Planning and Scheduling, Thesaloniki. Procc. of the 19th Int. Conf. on Automated Planning and Scheduling, 2009.

Vrba. P, Macrek. F, Vladimír. M," Using radio frequency identification in agent-based control systems for industrial applications", Engineering Applications of Artificial Intelligence, Vol 21, ISSN 3, pp 331-342, 2008

Watkins, C. J. C. H, "Learning from delayed rewards", PhD thesis, Cambridge University, Cambridge, England, 1989

Watkins C., Dayan P., « Q-learning », Machine Learning, Vol. 8, No 3, pp. 279–292, Elsevier, 1992.

Wu, L.H., Chen, Xin., Chen, X.D. and Chen, Q.X. "The Research on Proactive-Reactive Scheduling Framework Based on Real-Time Manufacturing Information". Materials Science Forum pp 789-794, 2009.

Wu. N, Zhou. M. C, Chu. F, "Short-term Scheduling for Refinery Process: Bridging the Gap between Theory and Application", International Journal Of Intelligent Control And Systems, Vol. 10, No. 2, pp 162-174, June 2005.

Xiang. W, H. P. Lee, « Ant colony intelligence in multi-agent dynamic manufacturing scheduling", Engineering Applications of Artificial Intelligence, Vol 21 ,ISSN 1, pp 73-85, 2008.

Zaffalon. L et P, Berguet, « Programmation concurrente et temps réel avec ADA95 », Collection Informatique, Presse polytechniques et universitaires romandes, 1995

Zennir. Y, «Apprentissage par renforcement et systèmes distribués : application à l'apprentissage de la marche d'un robot hexapodes », Thèse de Doctorat à l'Institut national des sciences appliquées de Lyon, 2004

Zighed D.A., Auray J.P., Duru. G., « SIPINA : Méthode et Logiciel », Lacassagne, 1992

Zulch. G et Stock , "Modelling and simulation of human decision-making in manufacturing systems", Proc. 38th conference on Winter simulation, pp 947 – 953, 2006.

Annexes

Annexe 1 : Présentation de la raffinerie d'Arzew NAFTEC/RA1Z

1 Présentation et petit historique de la raffinerie

Le géant pétrolier Algérien SONATRACH possède quatre raffineries sur le territoire national: Skikda, Alger, Arzew et Haoud-El-Hamra. Ces raffineries sont alimentées par un pipeline direct de la zone de stockage de Haoud-El-Hamra. Nous nous intéressons spécialement à la raffinerie d'Arzew RA1Z dans la wilaya d'Oran.

La capacité de production de cette raffinerie est de 7,2 million de tonne par an, elle alimente l'ouest du payé nord et sud avec un surplus exporté vers l'étranger via le terminal d'Arzew. Malgré ces chiffres a priori élevés, les raffineries ne sont exploitées qu'à 60% de leurs capacités (Dossier spécial Algérie, 2002).

2 Présentation générale des unités de la raffinerie

L'implantation de l'usine a été réalisée sur un site d'une superficie de 150 ha à 40 Km de la ville d'Oran.

La raffinerie a été conçue pour :

- Valoriser le pétrole brut de Hassi-Messaoud par son traitement local.
- Satisfaire les besoins de consommation en carburants pour la région Ouest et en lubrifiants et bitumes pour le marché national.

Le démarrage des unités a été lancé, par la mise en exploitation de l'unité des utilités, à partir du mois de juillet 1972. L'ensemble des unités de la Raffinerie est entré en service en mars 1973. Depuis, plusieurs travaux d'extension ont été menés.

La raffinerie est constituée de plusieurs unités complémentaires et successives qui déterminent le cheminement du flux produit.

2.1 Les utilités

Deux unités d'utilités *(zone 3 et zone 19)* produisent et assurent la distribution, pour les besoins de fonctionnement des différentes installations de : Eau distillée, Electricité, Air service et instrument, Eau de refroidissement traitée, Fuel gaz et Vapeur. Ces zones ne reçoivent pas de matière première, elles ne produisent que les besoins des autres zones.

2.2 Les carburants

L'Unité de distillation atmosphérique (U.ll) est l'unité principale du complexe qui traite le pétrole brut algérien. Les produits obtenus au niveau de cette unité sont: Gaz de pétrole liquéfié, Naphta lourd, Naphta léger, Kérosène, Gas oil et le brut réduit, qui constitue la charge des unités de distillation sous-vide pour la production des huiles de base.

L'Unité de reforming catalytique (U.12) : Le Naphta lourd de l'unité de distillation atmosphérique est traité dans cette unité dont le but est de produire une base à indice d'octane élevé (réformat), des GPL et un gaz riche en hydrogène.

L'Unité de traitement de gaz (U.13) : Les gaz de pétrole liquéfiés obtenus dans les unités de distillation atmosphérique et de reforming catalytique sont traités dans cette unité et séparés en produits suivants: Propane, Butane.

2.3 Les bitumes

L'Unité de flash sous-vide (U.14) Le brut réduit importé et fractionné en gas-oil sous vide et en produit visqueux obtenu en fond de colonne lequel est traité dans la section de soufflage à l'air pour obtenir du bitume pur (direct) communément appelé bitume routier. Les asphaltes provenant des unités de dé-asphaltage au propane sont mélangés au bitume direct pour obtenir les bitumes routiers. L'expédition des bitumes purs se fait par camion ou par bateau. Une ligne chauffée électriquement relie la raffinerie au port à cet effet.

L'Unité de bitume oxydé (U.IS) Du bitume direct mélangé avec du gazole sous vide constitue la charge de cette unité ou blown stock. Le bitume oxydé est obtenu par oxydation poussée avec de l'air. Ce bitume est conditionné dans des sacs plastiques de 25 Kg et dans des fûts de 200 Litres.

2.4 Les lubrifiants

La raffinerie d'Arzew dispose de :

- Deux chaînes de production d'huile de base de capacités annuelles respectives de 48 000 TM et 120 000 TM.

- Deux unités de fabrication, de mélange et de conditionnement des huiles finies. Ce sont ces unités qui vont nous intéresser par la suite.

- Deux unités de production et de conditionnement des graisses.

- Une unité de traitement et de deux unités de moulage de la paraffine.

Les deux chaines de production d'huiles de base disposent respectivement des unités suivantes: Distillation sous vide, Dé-asphaltage au propane, Extraction des aromatiques au furfural, Déparaffinage à la méthyl-éthylcétone et au toluène et Hydrofinishing des huiles. La 2éme chaîne dispose d'une unité d'hydrotraitement de la paraffine.

L'Unités de distillation sous vide (U.21 et 100) Le brut réduit de l'unité de distillation atmosphérique est fractionné dans les unités de distillation sous vide pour obtenir les produits intermédiaires suivants: Du gasoil sous vide, Les distillats: spindle, mivisqueuse ou SAE 10, visqueuse ou SAE 30 et un résidu court.

L'Unités de déasphaltage au propane (U.22 et 220) Le résidu court de l'unité de distillation sous vide y est traité avec un solvant sélectif, pour séparer l'huile lourde de l'asphalte. Le

produit obtenu s'appelle l'huile dé-asphaltée (DAO). L'asphalte est envoyé vers l'unité de bitume routier.

L'Unités d'éxtraction au furfural (U.23 et 300) Le furfural est utilisé comme solvant sélectif pour l'élimination des aromatiques et des naphtènes et permet d'améliorer l'indice de viscosité des trois distillats obtenus dans l'unité de distillation sous vide et de l'huile dé-asphaltée obtenue dans l'unité dé-asphaltage au propane. Quatre raffinats sont ainsi obtenus: Spindle, Mi-visqueuse ou SAE 10, Visqueuse ou SAE 30 et le Bright stock.

L'Unités de déparaffinage des huiles et de déshuilage des paraffines (U.24 et 400) Le mélange méthyl-éthyl-cétone (MEK) et toluène est utilisé comme solvant sélectif pour:

1- Extraire la paraffine ayant le point d'écoulement élevé des quatre raffinats et obtenir quatre huiles déparaffinées : Spindle, Mi-visqueuse ou SAE 10, Visqueuse ou SAE 30 et le Bright stock.

2- Extraire l'huile contenue dans la paraffine pour améliorer la consistance et le point de fusion de cette dernière.

L'Unités de traitement des huiles à l'hydrogène (U.25 et 500) Les quatre huiles déparaffinées y sont traitées alternativement avec de l'hydrogène dans un réacteur contenant un catalyseur à base de fer, de cobalt et de molybdène. Pour les quatre huiles de base finies obtenues, ce traitement améliore: La couleur, la stabilité thermique et la résistance à l'oxydation.

L'Unités de traitement de la paraffine (U.53 et 600) La raffinerie dispose de deux unités de traitement de la paraffine. La première qui utilise le "procédé" de la percolation par la bauxite est abandonnée pour des raisons d'environnement. La seconde utilise le traitement à J'hydrogène à travers un catalyseur. La stabilité et la couleur de la paraffine y sont améliorées.

L'Unités de fabrication et de remplissage des huiles finies (U51et U3100) Les quatre huiles de base obtenues après l'unité de traitement à l'hydrogène sont mélangées à des additifs selon une formulation pour chaque qualité d'huile moteur et industrielle à fabriquer: 14 Huiles moteur, 39 Huiles industrielles. Le conditionnement est fait en fûts de 180 Kgs pour les 2 types, en bidons de 2L et 5L pour les huiles moteur. Par la suite, c'est cette unité qui nous intéresse.

L'Unité d'emballage divisionnaire (U.3900) Cette unité permet de fabriquer des bidons de 2L et 5L, destinés au conditionnement des huiles moteur, à partir du polyéthylène haute densité (PEHD). Cette unité fabrique également des boites de l kg pour le conditionnement des graisses. Les opérations de remplissage, de capsulage et de fardage sont complètement automatisées.

L'Unités de moulage de la paraffine (U54 et 3300) La raffinerie dispose de deux unités de moulage de la paraffine: Pour la première, le refroidissement est assuré avec de l'eau. Pour la seconde, un système de réfrigération à l'ammoniac est utilisé. La paraffine, à l'état liquide après son passage dans l'unité de traitement à l'hydrogène, est refroidie et moulée sous forme de pains de 5 Kgs.

L'Unités de production et de conditionnement des graisses (U52 et 3200). Cinq qualités de graisses sont produites, selon une formulation, en mélangeant les huiles de base avec des additifs et des produits chimiques. Deux unités de production de graisse, existent: La première est une unité de fabrication par batch discontinu. La seconde est une unité de fabrication en continu suivant le "procédé texaco". Le conditionnement se fait dans des fûts de 180 Kgs, des sceaux de 16 Kgs et des boites de 1 Kg.

L'unité 3100 de fabrication des huiles finies est l'unité qui a attiré notre attention pour la réalisation de notre projet, cette unité est une jonction entre une production en continu qui est le traitement des huiles de base pour la fabrication des huiles finies. Le conditionnement et l'emballage est une production qui est faite en discontinu. L'unité 3100 doit pouvoir coordonner ces deux types de production, en gérant leurs différentes contraintes internes telle que les conditions fonctionnelles : qualité de l'huile dans les bacs, capacité des bacs… et des contraintes externes, principalement la demande des clients. Le contrôle de production dans cette unité implique la satisfaction simultanée de ces contraintes ce qui rend le problème difficile, sans oublier la soumission de l'unité à des perturbations liés aux pannes et aux arrêts de maintenance.

Annexe 2 : La plateforme MADKIT

1 La Plate-Forme Madkit

Pour valider le modèle Aalaadin, une plate-forme multi-agents appelée MADKIT (pour "Multi-Agent Development Kit", figure A2.1) a été construite.

Cette plate-forme base son implémentation sur les concepts d'agents, de groupe et de rôle, ainsi que trois grands principes d'architecture:

- Micro-noyau agent
- Agentification des services
- Modèle graphique componentiel

La philosophie de base d'Aalaadin/MADKIT est d'utiliser autant que possible la plate-forme pour son propre fonctionnement. Tout service non fourni par le noyau est confié à des agents spécialisés, structurés en groupes et identifiés par des rôles.

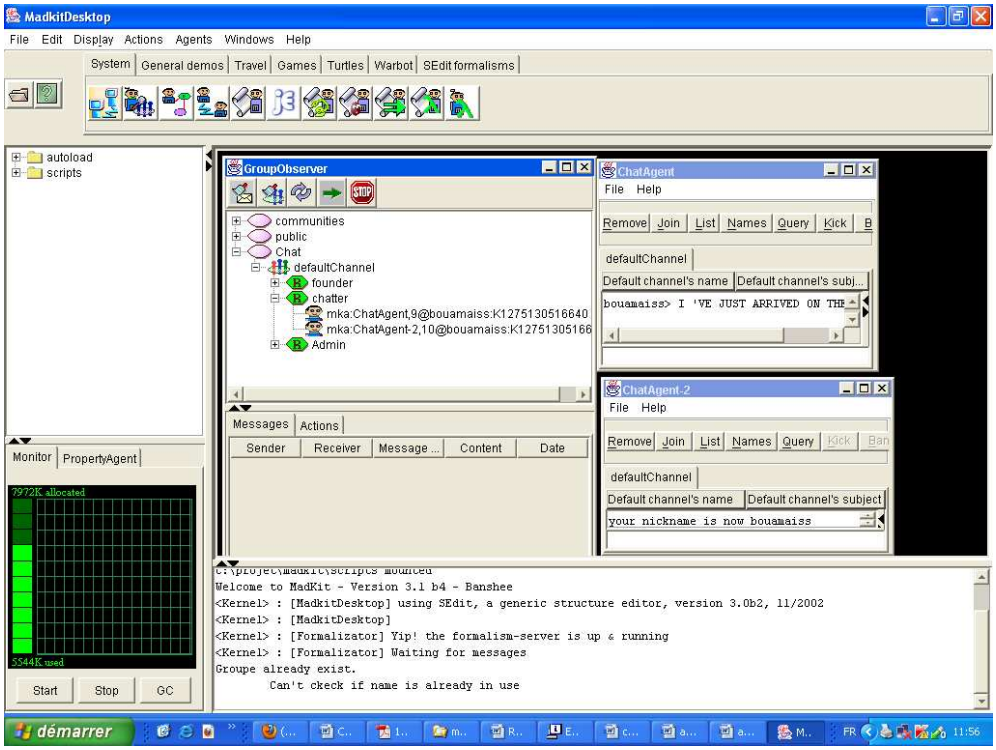


Figure A2.1. MadKit: Architecture générale

1.1 Le micro-noyau agent

Le micro-noyau agent de MADKIT est un environnement d'exécution d'agents compacts ; le terme "micro-noyau" est intentionnellement utilisé en référence à la philosophie des systèmes d'exploitation à micro-noyaux.

Le noyau ne gère que les tâches suivantes:

•Contrôle des groupes et rôles locaux. Le micro-noyau a la responsabilité de maintenir une information correcte sur les membres des groupes et sur les différents rôles tenus. Il transmet également les demandes d'admission ou de renseignements aux agents gestionnaires des groupes idoines.

•Gestion du cycle de vie. Le micro-noyau lance les agents, peut les suspendre ou les arrêter, et leur assigne des identifiants uniques.

•Passage de message local. Les messages envoyés d'un agent à l'autre sont remis par le noyau si le receveur et l'émetteur sont locaux.

Si ce n'est pas le cas, le message sera éventuellement remis par le biais d'un agent système spécialisé.

1.2 Agentification des services

1.2.1 Agents, groupes et rôles dans la plate-forme MADKIT

Les agents sont définis en héritant d'une classe abstraite qui fournit des mécanismes d'identifications, l'envoi et la réception de messages, et une API liée au système de groupe et rôle.

Ces méthodes permettent la création de groupe, l'admission, et l'émission de requêtes pour identifier les rôles présents dans un groupe, déterminer les agents en charge d'un rôle, ainsi que la demande, la délégation ou le retrait d'un rôle.

Quelques groupes particuliers sont définis:

•Un groupe local rassemble tous les agents s'exécutant sur le micro-noyau local. L'admission à ce groupe est automatique, et l'identification d'agents particuliers sur la plate-forme passe par l'implémentation standard de groupe et rôle.

•Le second groupe particulier est le groupe système, qui rassemble les agents ayant la possibilité d'agir sur le noyau, et donc un pouvoir sur les autres agents de la plate-forme. L'accès à ce groupe est donc sévèrement réglementé et restreint à certains agents lancés au démarrage ou validés par une signature.

•L'interaction entre agents systèmes et le noyau passe elle-même par une interaction agent standard.

Le tout premier agent créé à l'amorçage du noyau est en fait un agent "wrapper" qui sera le seul à avoir accès à toutes les possibilités et références internes du noyau. Il fonde le groupe local et le groupe système et y assure le rôle de gestionnaire de groupe.

Ensuite, les membres du groupe système pourront demander certaines actions privilégiées en dialoguant avec lui, d'où une sécurité accrue.

1.2.2 Services agents

Contrairement à d'autres plate-formes, MADKIT utilise des agents standard pour gérer l'envoi de message en distribué, la migration, la sécurité et d'autres aspects similaires. Cela rend la plate-forme particulièrement adaptable, vu que les services agents peuvent être remplacés à loisir.

Par exemple, il est possible d'implémenter un mécanisme de gestion de la distribution complètement différent de celui fourni sans avoir à changer quoi que ce soit dans les autres agents de la plate-forme. Ces services peuvent également être adaptés au moment de l'exécution en délégant des rôles d'agent à agent.

Ce principe de délégation de rôle a l'autre effet intéressant de permettre une adaptation à la charge: un agent peut tenir plusieurs rôles au début de la vie d'un groupe, et au fur et à mesure que le groupe grandit, lancer de nouveaux agents et leur confier certains de ses rôles.

1.3 Communication et distribution

Le passage de message utilise d'une manière classique un identifiant pour chaque agent, dont l'unicité est garantie non seulement au niveau de la plate-forme locale, mais également dans un contexte d'exécution distribuée. Ces identifiants sont ceux utilisés par le noyau pour tenir à jour les tables de groupes et de rôles.

Par conséquent, les groupes peuvent s'étendre sur plusieurs noyaux et les agents MADKIT fonctionnent en distribué de façon transparente.

La gestion de la distribution dans MADKIT correspond à deux rôles dans le groupe système:

•L'agent de rôle communicator est utilisé par le micro-noyau pour router les messages non locaux sur des plate-formes distantes, vers d'autres agents communicator qui réinjecteront les messages dans leurs noyaux respectifs.

•Le group synchronizer permet de distribuer les modifications faites aux tables de rôles et de groupes à l'échelle de plusieurs noyaux. L'agent ayant ce rôle intercepte les modifications et les transmet à ses pairs distants, qui les injecteront à leur tour dans leurs noyaux respectifs.

Ces synchronisateurs de groupes utilisent leur propre groupe distribué pour leur usage, après une phase de bootstrap.

Comme les mécanismes de la gestion de la distribution sont construits comme des agents MADKIT, la communication ou la migration peuvent être transformés en changeant l'agent concerné.

Une plate-forme MADKIT peut également fonctionner en mode purement local: il suffit de ne pas lancer les agents qui gèrent les communications.

On évite alors toute consommation de ressource inutile puisque seul le noyau est nécessaire.

Chacun de ces services n'est pas forcément géré par un seul agent. Par exemple, l'agent communicator peut être le représentant d'un groupe plus vaste rassemblant des agents spécialisés dans la communication par SMTP, Sockets, or Corba IIOP. Il confiera la tâche à l'agent spécialisé.

Ces caractéristiques font que MADKIT n'est pas exactement "une" plate-forme agent dans le sens classique. La taille réduite du noyau combinée avec le principe de services modulaires gérés par des agents permet en fait de déployer de multiples plate-formes , comme le montre la figure A2.2.

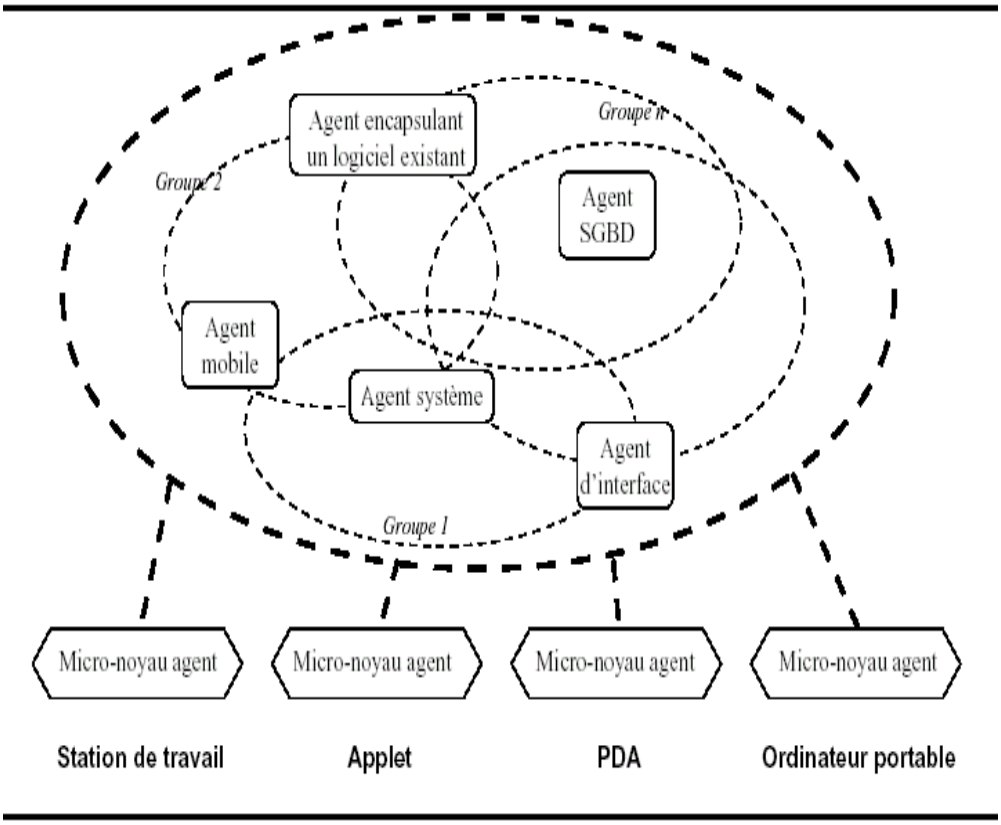


Figure A2.2. L'architecture MADKIT

2 Conclusion

La plate forme Madkit a répondu à nos attentes et nous a facilité la conception et la réalisation de notre système multi-agents.

Son utilisation a élargi notre champ de vision, car elle offre une liberté réelle au concepteur, nous avons ainsi totalement créé nos agents selon nos besoins sans contraintes.

Annexe 2 : La plateforme MADKIT

1 La Plate-Forme Madkit

Pour valider le modèle Aalaadin, une plate-forme multi-agents appelée MADKIT (pour "Multi-Agent Development Kit", figure A2.1) a été construite.

Cette plate-forme base son implémentation sur les concepts d'agents, de groupe et de rôle, ainsi que trois grands principes d'architecture:

- Micro-noyau agent
- Agentification des services
- Modèle graphique componentiel

La philosophie de base d'Aalaadin/MADKIT est d'utiliser autant que possible la plate-forme pour son propre fonctionnement. Tout service non fourni par le noyau est confié à des agents spécialisés, structurés en groupes et identifiés par des rôles.

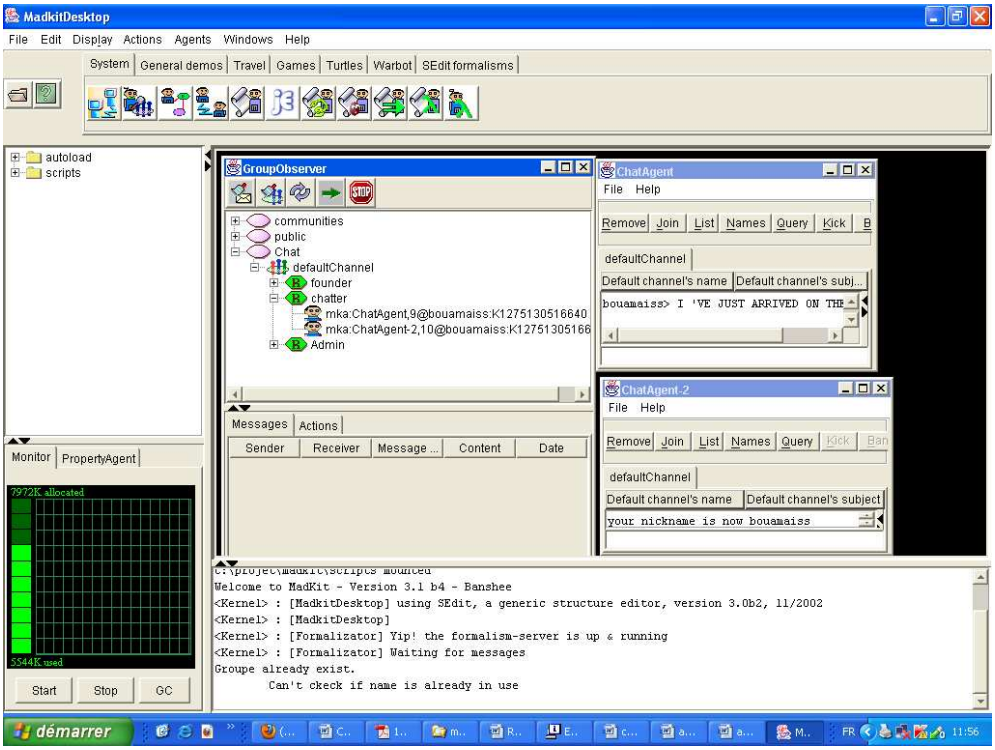


Figure A2.1. MadKit: Architecture générale

1.1 Le micro-noyau agent

Le micro-noyau agent de MADKIT est un environnement d'exécution d'agents compacts ; le terme "micro-noyau" est intentionnellement utilisé en référence à la philosophie des systèmes d'exploitation à micro-noyaux.

Le noyau ne gère que les tâches suivantes:

•Contrôle des groupes et rôles locaux. Le micro-noyau a la responsabilité de maintenir une information correcte sur les membres des groupes et sur les différents rôles tenus. Il transmet également les demandes d'admission ou de renseignements aux agents gestionnaires des groupes idoines.

•Gestion du cycle de vie. Le micro-noyau lance les agents, peut les suspendre ou les arrêter, et leur assigne des identifiants uniques.

•Passage de message local. Les messages envoyés d'un agent à l'autre sont remis par le noyau si le receveur et l'émetteur sont locaux.

Si ce n'est pas le cas, le message sera éventuellement remis par le biais d'un agent système spécialisé.

1.2 Agentification des services

1.2.1 Agents, groupes et rôles dans la plate-forme MADKIT

Les agents sont définis en héritant d'une classe abstraite qui fournit des mécanismes d'identifications, l'envoi et la réception de messages, et une API liée au système de groupe et rôle.

Ces méthodes permettent la création de groupe, l'admission, et l'émission de requêtes pour identifier les rôles présents dans un groupe, déterminer les agents en charge d'un rôle, ainsi que la demande, la délégation ou le retrait d'un rôle.

Quelques groupes particuliers sont définis:

•Un groupe local rassemble tous les agents s'exécutant sur le micro-noyau local. L'admission à ce groupe est automatique, et l'identification d'agents particuliers sur la plate-forme passe par l'implémentation standard de groupe et rôle.

•Le second groupe particulier est le groupe système, qui rassemble les agents ayant la possibilité d'agir sur le noyau, et donc un pouvoir sur les autres agents de la plate-forme. L'accès à ce groupe est donc sévèrement réglementé et restreint à certains agents lancés au démarrage ou validés par une signature.

•L'interaction entre agents systèmes et le noyau passe elle-même par une interaction agent standard.

Le tout premier agent créé à l'amorçage du noyau est en fait un agent "wrapper" qui sera le seul à avoir accès à toutes les possibilités et références internes du noyau. Il fonde le groupe local et le groupe système et y assure le rôle de gestionnaire de groupe.

Ensuite, les membres du groupe système pourront demander certaines actions privilégiées en dialoguant avec lui, d'où une sécurité accrue.

1.2.2 Services agents

Contrairement à d'autres plate-formes, MADKIT utilise des agents standard pour gérer l'envoi de message en distribué, la migration, la sécurité et d'autres aspects similaires. Cela rend la plate-forme particulièrement adaptable, vu que les services agents peuvent être remplacés à loisir.

Par exemple, il est possible d'implémenter un mécanisme de gestion de la distribution complètement différent de celui fourni sans avoir à changer quoi que ce soit dans les autres agents de la plate-forme. Ces services peuvent également être adaptés au moment de l'exécution en délégant des rôles d'agent à agent.

Ce principe de délégation de rôle a l'autre effet intéressant de permettre une adaptation à la charge: un agent peut tenir plusieurs rôles au début de la vie d'un groupe, et au fur et à mesure que le groupe grandit, lancer de nouveaux agents et leur confier certains de ses rôles.

1.3 Communication et distribution

Le passage de message utilise d'une manière classique un identifiant pour chaque agent, dont l'unicité est garantie non seulement au niveau de la plate-forme locale, mais également dans un contexte d'exécution distribuée. Ces identifiants sont ceux utilisés par le noyau pour tenir à jour les tables de groupes et de rôles.

Par conséquent, les groupes peuvent s'étendre sur plusieurs noyaux et les agents MADKIT fonctionnent en distribué de façon transparente.

La gestion de la distribution dans MADKIT correspond à deux rôles dans le groupe système:

•L'agent de rôle communicator est utilisé par le micro-noyau pour router les messages non locaux sur des plate-formes distantes, vers d'autres agents communicator qui réinjecteront les messages dans leurs noyaux respectifs.

•Le group synchronizer permet de distribuer les modifications faites aux tables de rôles et de groupes à l'échelle de plusieurs noyaux. L'agent ayant ce rôle intercepte les modifications et les transmet à ses pairs distants, qui les injecteront à leur tour dans leurs noyaux respectifs.

Ces synchronisateurs de groupes utilisent leur propre groupe distribué pour leur usage, après une phase de bootstrap.

Comme les mécanismes de la gestion de la distribution sont construits comme des agents MADKIT, la communication ou la migration peuvent être transformés en changeant l'agent concerné.

Une plate-forme MADKIT peut également fonctionner en mode purement local: il suffit de ne pas lancer les agents qui gèrent les communications.

On évite alors toute consommation de ressource inutile puisque seul le noyau est nécessaire.

Chacun de ces services n'est pas forcément géré par un seul agent. Par exemple, l'agent communicator peut être le représentant d'un groupe plus vaste rassemblant des agents spécialisés dans la communication par SMTP, Sockets, or Corba IIOP. Il confiera la tâche à l'agent spécialisé.

Ces caractéristiques font que MADKIT n'est pas exactement "une" plate-forme agent dans le sens classique. La taille réduite du noyau combinée avec le principe de services modulaires gérés par des agents permet en fait de déployer de multiples plate-formes , comme le montre la figure A2.2.

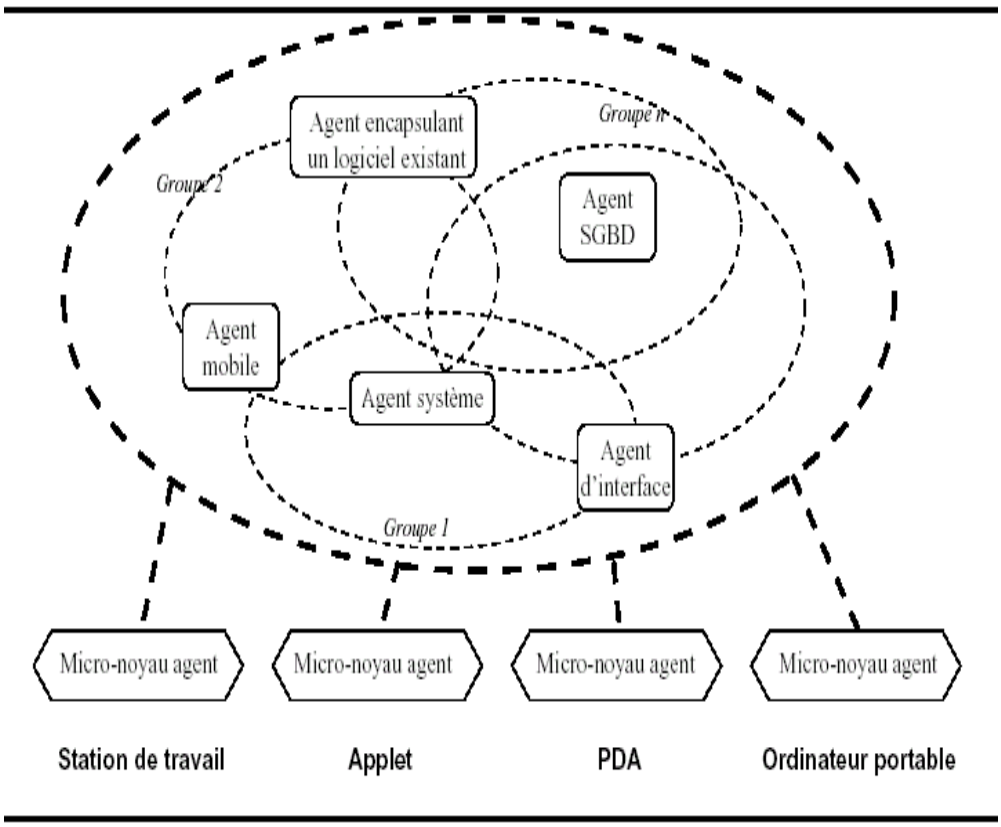


Figure A2.2. L'architecture MADKIT

2 Conclusion

La plate forme Madkit a répondu à nos attentes et nous a facilité la conception et la réalisation de notre système multi-agents.

Son utilisation a élargi notre champ de vision, car elle offre une liberté réelle au concepteur, nous avons ainsi totalement créé nos agents selon nos besoins sans contraintes.

Annexe 3 : Présentation du système de commande DCS

1 Introduction

Le progrès technologique dans le monde de l'électronique et de l'informatique a permis une évolution considérable dans le domaine du contrôle des procédés industriels. Cette évolution est traduite par un changement dans les techniques de contrôle : Passage des systèmes pneumatiques aux systèmes électroniques analogiques puis numériques, du contrôle centralisé au contrôle distribué DCS et des systèmes a relais aux systèmes à base d'Automates Programmables.

2 Système de contrôle distribué DCS

Ce système appelé DCS (distributed control system), est composé d'un ensemble d'unités à base des microprocesseurs pour garantir le contrôle, la commande, et l'exécution des taches industrielles (régulation, ouverture/fermeture des vannes, arrêt/mise en marche des machines). Les traitements et les données son répartis sur les différentes unités du système d'où l'appellation (system de contrôle distribué).

Le principe de fonctionnement de ce système est le même que celui du DDC (Display Data Channel). La différence est que, pour le DCS, les taches sont distribuées entre plusieurs modules ou abonnés. Le DCS a donc toutes les capacités d'un DDC en plus de l'amélioration du temps de réponse dû à la présence de plusieurs calculateurs (processeurs) et à la répartition des taches.

Les autres avantages du DCS sont les suivants :

- Présence d'une redondance efficace
- Grande capacité de traitement et d'exploitation.
- Architecture évolutive.
- Facilité des développements, modifications, extensions …etc.
- Aspect économique : vue la distribution physique, les coûts de maintenance du système et le manque à produire sont considérablement réduits.
- La réalisation de modification se fait sans perturbation du procédé.
- Facilité d'exploitation à tous les niveaux.
- Augmentation de la sécurité de l'unité.
- Possibilité d'intégration de sous-systèmes.

La figure A3.1 montre l'évolution du contrôle des installations pétrolière, du contrôle manuel au DCS.

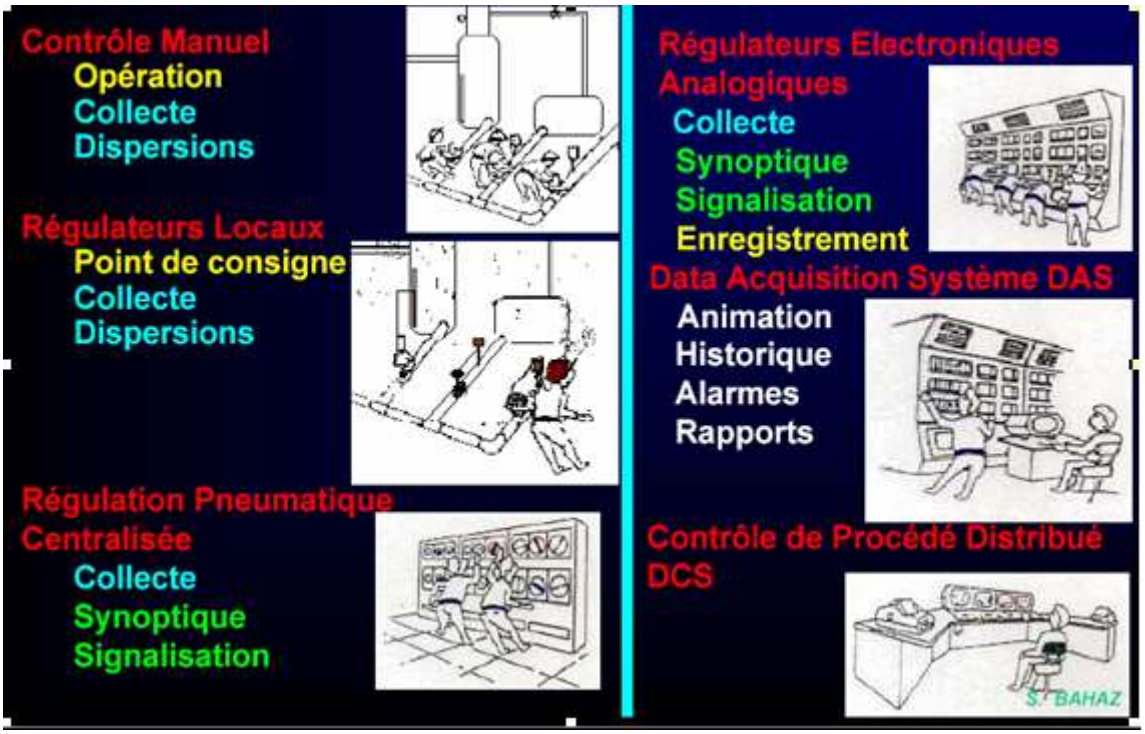


Figure A3.1. Historique des systèmes de contrôle

3 Boucle de régulation d'un système DCS

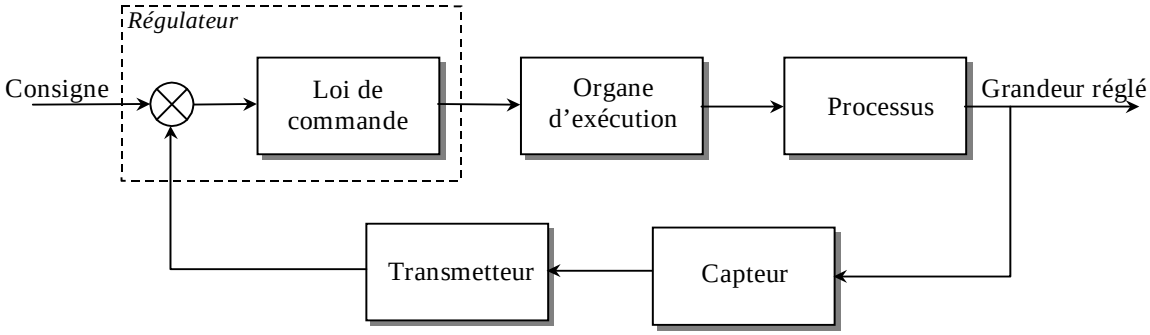


Figure A3.2. Schéma de la boucle de régulation

Le fonctionnement du système DCS est basé sur la boucle de régulation présentée dans la figure A3.2, cette boucle peut être décomposée en deux chaînes :

- Une chaîne directe ou chaîne d'action,
- Une chaîne de retour ou chaîne d'information.

L'idéal serait que le signal d'erreur soit nul à tout instant.

Annexe 4 : Exemple de déroulement du Q-Learning

1 Introduction

Dans cette section, nous détaillons un exemple très simple d'une entité apprenante qui utilise l'apprentissage par renforcement. C'est un agent machine avec des états est actions réduits. Nous avons montré comment une prise de décision se déroule en apprentissage par renforcement et comment une politique est renforcée.

2 Agent Machine

Dans cet exemple, nous traitons un agent machine primitif (Utilisé dans les premières expérimentations de notre approche).

2.1 Les données de l'apprentissage

Cet agent machine a les propriétés suivantes :

L'ensemble des états:

S1 en marche

S2 en panne

S3 arrêtée

S4 en marche et lancement tâche

S5 en panne et lancement tâche

S6 arrêtée et lancement tâche

S7 en marche et fin tâche

S8 en panne et fin tâche

S9 arrêtée et fin tâche

Actions:

A1 Mise en attente

A2 Arrêter machine

A3 Changer machine

A4 Lancement

Récompense:

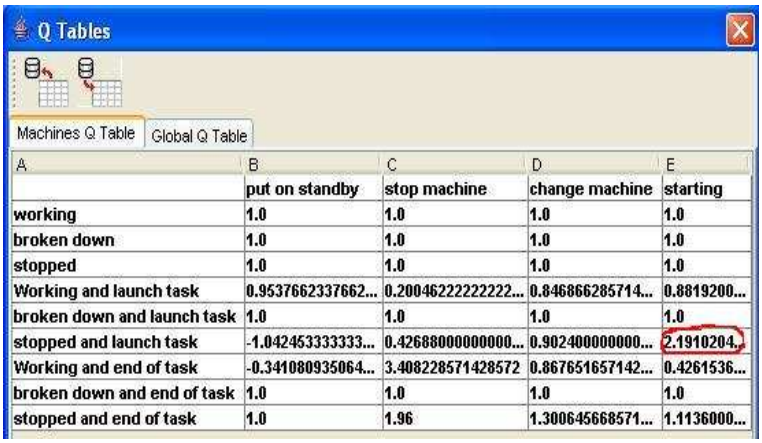
R1 En marche et tâche encours **0**

R2 Lancement et tâche encours **+4**

R3 Arrêt et fin de tâche **+5**

R4 Arrêt et tâche encours **-5**

Nous observons sa table des Q-valeur à un instant t (Figure A4.1):



A	B	C	D	E
	put on standby	stop machine	change machine	starting
working	1.0	1.0	1.0	1.0
broken down	1.0	1.0	1.0	1.0
stopped	1.0	1.0	1.0	1.0
Working and launch task	0.9537662337662...	0.20046222222222...	0.846866285714...	0.8819200...
broken down and launch task	1.0	1.0	1.0	1.0
stopped and launch task	-1.042453333333...	0.42688000000000...	0.902400000000...	2.1910204...
Working and end of task	-0.341080935064...	3.408228571428572	0.867651657142...	0.4261536...
broken down and end of task	1.0	1.0	1.0	1.0
stopped and end of task	1.0	1.96	1.300645668571...	1.1136000...

Figure A4.1. La table Q avant prise de décision

2.2 La perception de l'état

L'agent perçoit son état et l'état de son environnement. L'état actuel est « Machine arrêtée et lancement en cours». Nous nous intéressons à l'agent machine, son état est Machine arrêtée, c'est l'état S6

2.3 Choix de l'action

L'agent doit sélectionner l'action à entreprendre, sur la ligne S6, l'action dont la valeur Q est la plus élevée sera sélectionnée. C'est A4 qui correspond à « Lancement ».

Suite à ce choix, il faut mettre à jour l'état de la machine qui sera « En marche et lancement tâche » correspondant à S4.

Il faut mettre également à jour aussi la valeur Q(S6, A4) , ce que nous allons détailler.

2.4 La mise à jour de la valeur Q

Le renforcement perçu pour cette action est R2 (+4) car nous avons une « Mise en marche » et une tâche en cours. D'où :

$Q_{t+1}(s_6, a_4)$ =

$$(1-\alpha)Q_t(s_6,a_4)+\alpha\left[R_t(s_6,a_4)+\gamma \underset{a_i \in A}{Max} Q(s_4,a_i)\right]=$$

$$(1-0.256)*2.191+0.256\left[4+0.8*0.95\right]=2.85$$

La table Q mise à jour et observée à t+1 ainsi (figure A4.2):

Working and launch task	0.8883327463705...	0.14023322313599...	0.828935748267...	0.8339422...
broken down and launch task	1.0	1.0	1.0	1.0
stopped and launch task	-1.032789461966...	0.30171406722644...	0.848561396542...	2.8545881...
Working and end of task	-0.251808423279...	4.048822013475247	1.104770737947...	0.3176238...
broken down and end of task	1.0	1.0	1.0	1.0
stopped and end of task	1.0	1.96	1.476473101782...	1.8569867...

Figure A4.2. La table Q après mise-à-jour

2.5 Le renforcement de la politique

Cette manipulation a montré comment un choix peut être renforcé. La politique renforcée dans cet exemple est : A chaque fois qu'une machine est à l'arrêt et qu'il y'a une demande de fabrication, il est préférable de la mettre en marche pour lancer une tâche lorsqu'elle est demandée.

Annexe 5 : Détail des données expérimentales

1 Configuration des ressources du système

Bac	Pompe principale	Débit M3/h	Huile de base	TONNAGE
TK2501	P3101	45	SPO	2343
TK2502	P3102	45	SAE10	2371
TK2503	P3104	45	SAE30	2399
TK2506	P3105	45	SAE30	1924
TK2504	P3106A	45	B/S	1968
TK2505	P3106B	45	B/S	1968

Figure A5.1. Caractéristiques des bacs de stockage de l'huile de base

Les huiles de base sont classées de la moins dense à la plus dense (SPO →B/S)

Chaque bac est associé à deux pompes ou trois avec un débit de 45m^3/h. ces pompes sont utilisées pour le remplissage ou le vidange des bacs (figure A5.1).

2 Plan de production

Finished oil			Base oil needed				
Code	Grades	quantity to	SPO	SAE10	SAE30	BS	Total
P1	NAFTILIA 20W50	1000		126,3	776,2		**902,5**
P2	CHELIA 10W	1000		956			**956**
P3	NAFTILIA 40	800			616,8	149,6	**766,4**
P4	CHIFFA 40	500			380	106	**486**
P5	CHELIA 40	1000			766,6	200	**966,6**
P6	CHELIA TD 20W40	2000					**0**
P7	TISKA 32	1000		992			**992**
P8	TISKA 68	2000					**0**
P9	CHELIA VPS 20W40	2000		475	1057	300	**1832**
P10	TASSILIA EP 90	1200		240	120	780	**1140**
P11	TASSADIT A2	1200		210		600	**810**
	Total	**13700**	**0**	**2999,3**	**3716,6**	**2135,6**	**8851,5**

Figure A5.2. Tables des lubrifiants à fabriquer

3 Plan de maintenance

Tâche d'entretien	Durée	date début
Changer courroie	3	2
rinçage bac	12	10
vérification étanchéité	1	31

Figure A5.3. Plan maintenance globale

Annexe 6 : Les méthodes de construction de politique dans un MDP

1 Méthodes de Programmation dynamique

Les méthodes de programmation dynamique utilisent l'expression (chap4 1) pour le calcul de l'utilité des états. La politique π est estimée suite à la convergence de l'expression (chap4 1), l'amélioration de cette politique consiste à choisir les actions qui privilégient les transitions vers les états dont la somme des utilités est maximale. Parmi ces méthodes citons la programmation dynamique asynchrone (Barto et al., 1991).

Les méthodes de programmation dynamique supposent un modèle parfaitement connu (P et R sont définis explicitement), tel que dans l'exemple donné dans la figure 4.2 où tous les états et actions sont connus avec un modèle de transition parfaitement déterminé. La DP (Dynamic Programming) suppose aussi que l'espace d'états et l'espace d'actions A sont finis, donc les DPs sont surtout utilisées pour les MDP dis finis (PDMF).

2 Méthode Monte Carlo (MC)

Les méthodes dites Monte Carlo visent à calculer une valeur numérique en utilisant des procédés aléatoires, c'est-à-dire des techniques probabilistes. La méthode Monte Carlo simule un grand nombre de trajectoires issues de chaque état *s* de *S*, et calcule $V^{\pi}(s)$ en moyennant les coûts observés sur chacune de ces trajectoires. À chaque expérience réalisée l'agent mémorise les transitions qu'il a effectuées et les récompenses qu'il a reçues. A la fin de la trajectoire il met à jour l'estimation de la valeur des états parcourus en associant à chacun d'eux la part de la récompense reçue qui lui revient.

Ces méthodes sont coûteuses en temps calculatoire, leurs propriétés de convergence ne sont pas encore claires et leur efficacité a en pratique été peu évaluée (PDMIA, 2004). La méthode MC est aussi limitée en général aux processus statiques puisque le temps n'est pas une variable explicite. La méthode MC n'est alors pas adaptée aux systèmes dynamiques tels que le pilotage de production où le temps a une grande influence sur le système.

3 Les méthodes de différences temporelles (TD)

Les méthodes de différences temporelles sont venues justement pour compléter la méthode Monte Carlo et palier ses inconvénients en profitant de l'incrémentalité (mise-à-jour à chaque étape) de la programmation dynamique.

Les méthodes TD permettent de ce fait une mise à jour de la fonction d'état après chaque transition du système et non à la fin de la trajectoire (l'ensemble des séquences). Sans se baser sur un modèle, elles effectuent un ensemble d'expériences sur lesquelles elles se basent pour mettre à jour la fonction d'état.

Soit *s* un état non terminal visité à l'instant *t*, la mise à jour de la fonction V^{π} est menée sur la base de ce qui arrive après cette visite (équation 1).

$$V_{t+1}^{\pi}(s) = V_t^{\pi}(s) + \alpha\left[r_{t+1} + \gamma.V_t^{\pi}(s') - V_t^{\pi}(s)\right] \qquad (1)$$

$\alpha \in [0,1]$ est un taux d'apprentissage

L'erreur de prédiction du gain estimé est

$$r_{t+1} + \gamma.V_t^{\pi}(s') - V_t^{\pi}(s)$$

Un algorithme TD a la forme générale suivante (Figure A6.1):

```
Initialisation arbitraire V^Π(s)
Répéter pour (chaque épisode):
Initialiser  s
      Répéter pour chaque transition :
            a ← action donnée par Π pour s
            Exécuter a, observer la
            récompense r et le nouvel état s'
            V(s) = V(s) + α[r + γ.V(s') − V(s)]
            S← s'
      Jusqu'à l'état s terminal
```

Figure A6.1. Algorithme générale d'une méthode TD (Sutton, 1998)

3.1 Dilemme exploration vs exploitation

Pour régler la politique de façon à maximiser sa récompense sur le long terme, l'apprentissage doit assurer un compromis entre l'*exploitation*, qui consiste à refaire les actions dont on connaît déjà la récompense qu'elle procure, et l'*exploration*, qui consiste à parcourir de nouveaux couples (état, action) à la recherche d'une récompense cumulée plus grande, mais au risque d'adopter parfois un comportement qui n'est pas optimal. En effet, tant que l'agent n'a pas exploré tout son environnement, il n'est pas certain que la meilleure politique qu'il connaît est la politique optimale.

L'espace d'état est exploré naturellement car la dynamique du système permet le passage d'état en état sans avoir à prendre des décisions. Par contre le choix d'action doit être géré par l'algorithme. Ce choix peut se faire en choisissant aléatoirement les actions à effectuer suivant une distribution uniforme. On peut aussi exploiter la connaissance déjà apprise pour déduire une probabilité de la prochaine action à tester. Une des probabilités les plus utilisées est une distribution de probabilité de Boltzmann qui se formalise comme suit:

$$P(a|s) = e^{Q(s,a)/T} \Big/ \sum_{a_i \in A} e^{Q(s,a_i)} \qquad (2)$$

(PDMIA, 2004)

La température T est prise assez élevée au début, puis décroit vers 0 au fur et à mesure que l'apprentissage progresse (figure A6.2). Pour ce faire, T a la formule suivante :

$$T = \frac{1}{nIT}$$

Où *nIT* est le nombre de fois où la paire *(s,a)* a été visitée (nombre d'itérations sur (s,a)).

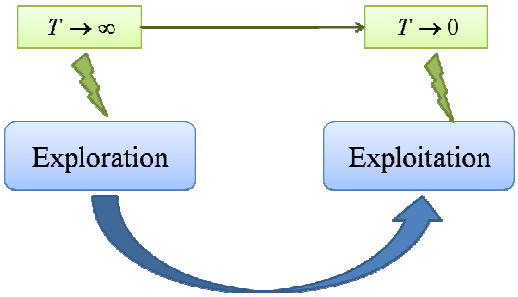


Figure A6.2. Progression de l'apprentissage par renforcement sous l'influence de la température de Boltzmann

Dans notre système de pilotage nous avons un ensemble d'agents qui se doivent d'être adaptatifs et d'apprendre un comportement optimal pour la résolution des problèmes d'ordonnancement. le contrôle du système multi-agent est donc primordial.

3.2 Le contrôle de l'apprentissage Multi-agents

Lorsqu'on a un ensemble d'agents qui peuvent être en compétition ou en coopération. Les agents (joueurs) ne suivent pas nécessairement un objectif commun. Des mécanismes de contrôle peuvent être mis en place.

En s'appuyant sur la théorie des jeux de Markov ou jeux stochastiques des critères de performance de stratégie apprise ont été développés. Nous commencerons par donner un aperçu sur les jeux de Markov

Jeu de Markov

Le cadre des jeux de Markov va nous permettre de mieux comprendre les effets de la concurrence entre agents.

Définition (jeu de Markov)

Un jeu de Markov est défini par un tuple $G = \langle S, N, A_1, ..., A_n, T, u_1, ..., u_n \rangle$, *où :*

- *S est un ensemble fini d'états,*
- *N est un ensemble fini de* n *joueurs,*
- A_i *est un ensemble fini d'actions possibles pour le joueur* i. *On pose* $A = A_1 \times ... \times A_n$ *l'ensemble des actions jointes.*
- $T : S \times A \times S \to [0,1]$ *est la fonction de transition du système, laquelle donne la probabilité d'aller d'un état* s *à un état* s_0 *quand l'action jointe* a *est accomplie.*

- $r_i : S \times A \to \mathbb{R}$ *est la fonction d'utilité réelle pour le joueur* i (i.e. *sa fonction de récompense).*

Ainsi, chaque joueur *i* tente de maximiser son espérance de gain individuellement, en utilisant le critère de performance γ-pondéré suivant :

$$E(\sum_{j=0}^{\infty} \gamma^j r_{i,t+j})$$

Où $r_{i,t+j}$ est la récompense de l'agent *i* après *j* pas de temps dans le futur. On pourrait définir un critère non pondéré, mais dans ce cas tous les jeux de Markov n'ont pas de stratégie optimale tel que montré dans Buffet (2003).

Concernant les DEC-MDP, une politique de Markov *pure* pour un joueur *i* est une fonction $\pi_i : S \to A_i$. Un ensemble de politiques pour tous les joueurs $\pi = \{\pi_1, ..., \pi_n\}$ est appelé vecteur de politiques.

Une politique de Markov *mixte* pour un joueur *i* est une fonction $\pi_i : S \to \Pi(A_i)$ qui associe à un état une distribution de probabilité sur les coups possibles, c'est aussi une politique stochastique.

Pour résoudre les jeux de Markov, deux méthodes d'apprentissage par renforcement sont identifiées :

Apprentissage par renforcement à somme nulle

Dans cette méthode, les deux joueurs sont considérés comme de purs adversaires, le gain de l'un engendre la perte de l'autre et la somme de leurs renforcements est nulle.

Dans ce contexte, Littman (1994) a présenté une approche pour apprendre par renforcement une politique "optimale" : Sans avoir d'idées sur le jeu de l'adversaire, on fait l'hypothèse qu'il joue au mieux, donc on essaie de maximiser notre gain tout en sachant que l'autre essaie de le minimiser. De ce fait, dans la fonction de mise-à-jour le terme $\max_{a \in A}$ est remplacé par $\max_{\pi \in \Pi(A_1)} \max_{a_2 \in A_2}$ où A_1 est l'ensemble d'actions de l'agent*1* et A_2 l'ensemble d'actions de l'agent *2* ce qui donne la formule suivante, extension de (1) :

$$Q(s, a_1, a_2) \leftarrow (1-\alpha) Q(s, a_1, a_2) + \alpha \left[r + \gamma \max_{\pi(A_1)} \min_{a_2} Q(s', \pi(A_1), a_2) \right]$$

Apprentissage par renforcement à somme quelconque

Dans cette méthode, les joueurs sont considérés comme des coéquipiers qui coopèrent pour atteindre un équilibre ou un but global (Hu et Wellman, 1998 a,b)

Un équilibre est une situation qui permet à tous les agents d'acquérir une politique suffisamment satisfaisante et qu'aucun autre agent ne peut espérer améliorer encore plus son gain (figure A6.3).

Cet équilibre engendre la notion d'optimum Pareto qui est un équilibre où tout agent a une stratégie suffisamment satisfaisante qui ne peut pas être améliorée sans engendrer la perte d'un autre agent (Buffet, 2003).

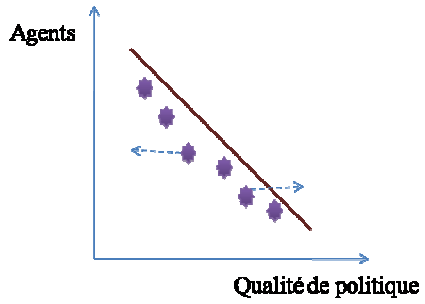


Figure A6.3. Situation d'équilibre et possibilité de le perdre

Dans un apprentissage à somme quelconque, c'est cet équilibre qui est recherché. En effet, il s'agit de considérer *n* agents (pas nécessairement 2), chacun ayant ses propres gains indépendants des gains des autres agents. Chaque agent cherche à maximiser ses gains en considérant que ses congénères font de même. Les agents vont devoir se mettre d'accord sur le point fixe à atteindre, le point d'équilibre, et ce sans communication.

A chaque étape du jeu (situation *s*), chaque agent choisit de résoudre le problème en exécutant une action *a*, l'exécution conjointe de leurs actions les conduit à une situation s' et à une nouvelle étape du jeu.

Trois niveaux de coopérations sont définis, en considérant ou non la modélisation de l'autre (Buffet, 2003):

- Le niveau 0 correspond à un agent compétitif, c'est-à-dire au simple apprentissage par l'observation du comportement de l'autre (l'agent n'a aucune idée sur ses congénères, comme s'ils faisaient partie de l'environnement, en négligeant le fait qu'ils ont des objectifs propres).
- Le niveau 1 au contraire considère que l'autre est compétitif : qu'il a un but, mais qu'il adopte pour moi une modélisation de niveau 0 (il ne me connaît pas).
- Un niveau 2 de modélisation est défini par récurrence comme une situation où un agent considère son congénère comme étant de niveau 1 (souvent l'hypothèse peut être fausse).

Hu et Wellman (1998a) ont montré que le fait qu'une modélisation soit mal faite est bien pire qu'une absence de modélisation.

De ce fait, notre but sera alors, de permettre aux agents d'être suffisamment coopératifs en s'échangeant des informations sur leurs capacités et leurs intentions sans une modélisation directe du comportement de l'autre.

Pour bien contrôler leur convergence (Zennir, 2004), il faudra déterminer :

- La perception d'états pour chaque agent et les actions qu'ils sont capables d'exécuter

- La fonction de renforcement individuelle qui leur permet d'atteindre le point d'équilibre et l'objectif global
- Les motivations de communication et les informations échangées

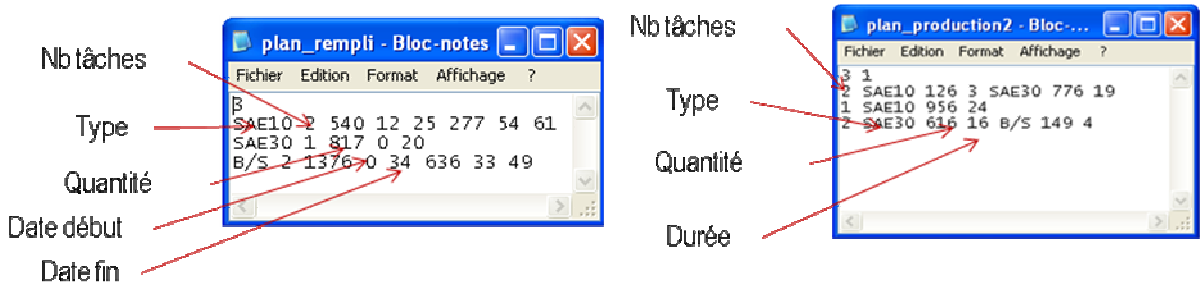
Annexe 7 : Prise de décision

1 Introduction

Dans cette annexe, nous détaillons un exemple de prise de décision pour l'allocation des ressources. Nous montrons comment une prise de décision se déroule en apprentissage par renforcement et comment une politique est renforcée.

2 Présentation du problème

Nous supposons les plans décadaires de la vue (figure A7.1). Le plan (A) est le plan de remplissage provenant de l'unité en amont. Le plan (B) est le plan de production, le détail de ce plan est donné par la figure A7.2.



(A) (B)

Figure A7.1. Les plans décadaires

		Tache1				Tache2					Termi optim
	Tache	**Quant**	**Durée**	**D-début présum**	**Ress**	**Tache**	**Quant**	**Durée**	**D-début présum**	**Ress**	
P1	T10	126	3	0	M1	T11	776	19	4	M2,M3	23
P2	T20	956	24	0	M1						24
P3	T30	616	16	0	M2,M3	T31	149	4	17	M4,M5	21

Figure A7.2. Plan production

3 Agent Ressource

L'état d'un agent ressource est donné par l'état de la ressource à cet instant « en marche » ou « à l'arrêt » avec sa séquence opératoire à cet instant selon la requête perçue.

Nous supposons dans un premier temps que les tâches de remplissage ont des dates débuts exigées, de ce fait, elles sont directement intégrées dans les séquences des ressources. Ce qui donne le Gantt suivant :

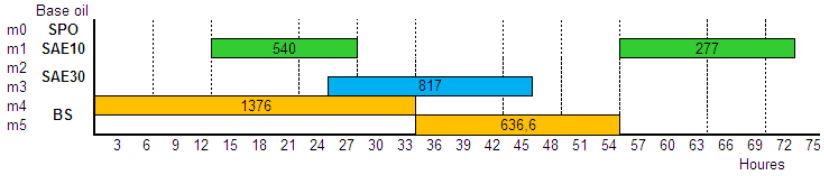


Figure A7.3. Gantt de remplissage

L'agent M1 a la séquence suivante :

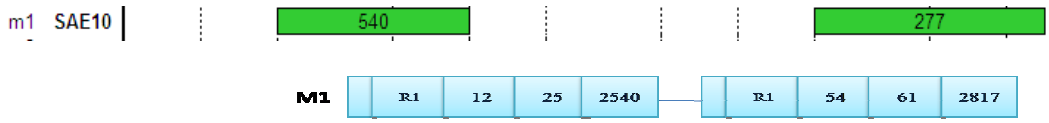


Figure A7.4. Séquence de la ressource

Ensuite, les agents produit demandent à se faire exécuter leurs tâches de production (de vidange), en supposant que l'agent ressource accepte d'exécuter la tâche et l'agent produit accepte la proposition. Nous aurons le digramme de séquence suivant (figure A7.5) :

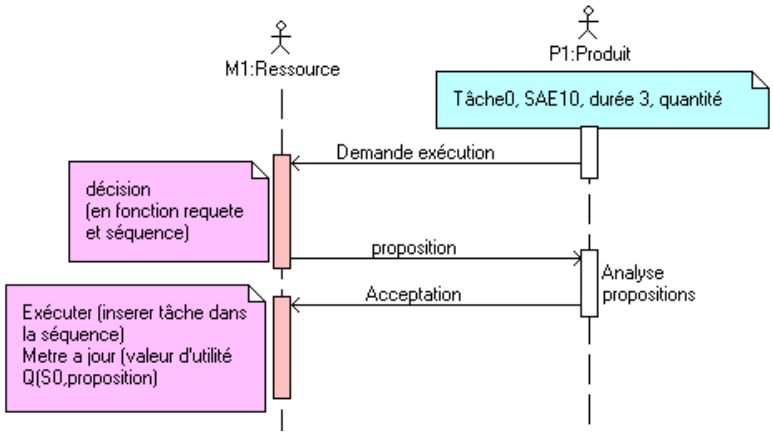


Figure A7.5. Prise de décision par l'agent ressource pour la tâche 0 de P1

La tâche est alors intégrée dans la séquence de l'agent ressource:

Figure A7.6. Séquence de la ressource 2

L'agent ressource met à jour sa table d'utilité en fonction de la décision qu'il avait prit avec les conséquences qu'elle avait engendré. L'agent cherche d'abord si cet état existe ou non dans sa table, s'il n'existe pas il l'ajoute sinon il met juste à jour la valeur d'utilité (figure A7.7).

Q(s/a)	Ne pas	Proposer	Proposer
S_0	$Q(s_0,a_0)$	$Q(s_0,a_1)$	
⋮			
S_i : 		$Q(s_i,a_1)+\alpha[1/Cmax_P+\gamma Q(s_{i+1},a_\cdot)- Q(s_0,a_0)]$	
S_{i+1} 			

Figure A7.7. Tables des valeurs d'utilité pour la ressource M1

La mise à jour de la valeur d'utilité montre que plus le *Cmax partiel* (Figure 7.8) est petit plus le choix de l'action pour cet état (séquence ressource et tâche) est renforcé.

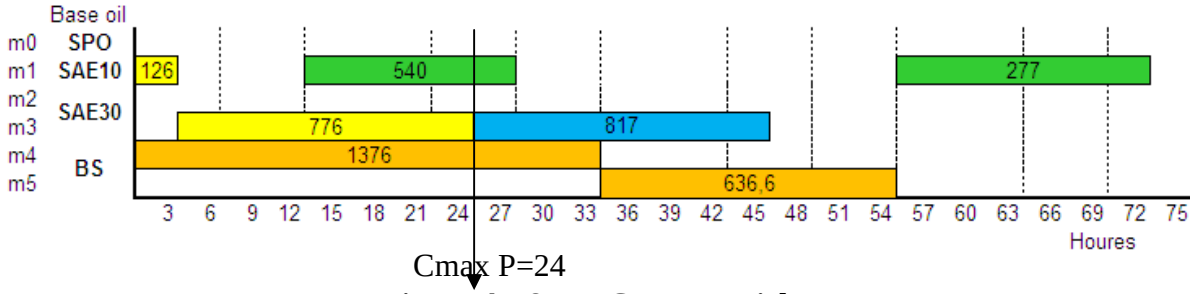


Figure A7.8. Cmax partiel

L'agent P2 demande aussi de se faire exécuter une tâche, on suppose aussi que l'agent ressource accepte de l'exécuter et l'agent produit accepte la proposition (figure A7.9):

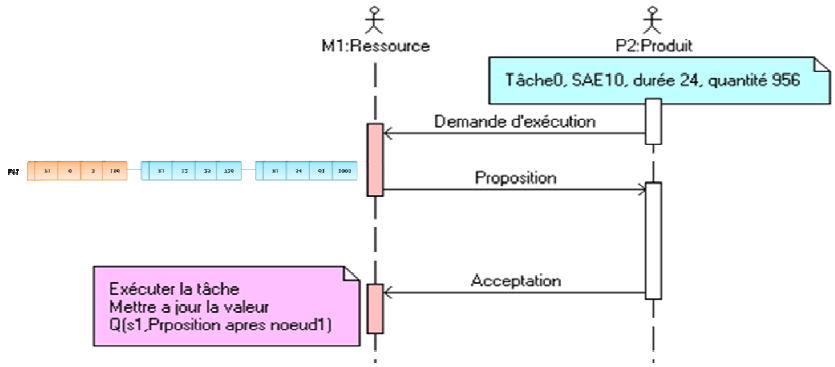


Figure A7.9. Prise de décision par l'agent ressource pour la tâche 0 de P1

Alors le tableau d'utilité sera comme suit (figure A7.10):

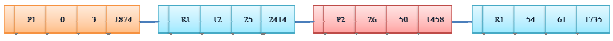
Q(s/a)	Ne pas	Proposer	Proposer
S_0	$Q(s_0,a_0)$	$Q(s_0,a_1)$	
⋮			
S_i : 			
S_{i+1} 		$Q(s_{i+1},a_1)+\alpha[1/Cmax_P+\gamma Q(s_{i+2},a')- Q(s_0,a_0)]$	
S_{i+2} 			

Figure A7.10. Tables des valeurs d'utilité pour la ressource M1

Ainsi de suite pour toutes les tâches, jusqu'à terminaison de la production de tout les produits, ce qui donnera le Gantt suivant :

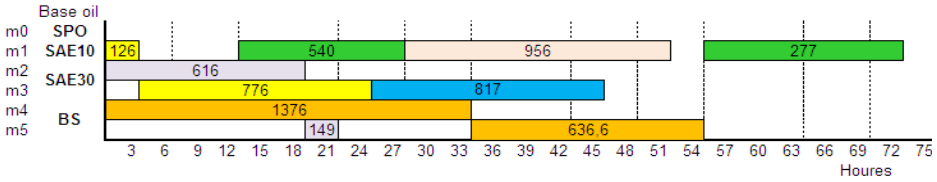


Figure A7.11. Gantt résultat

L'exploitation de cette table d'utilité a permit d'arriver à un Cmax optimal, en autorisant le déplacement des tâches de remplissage :

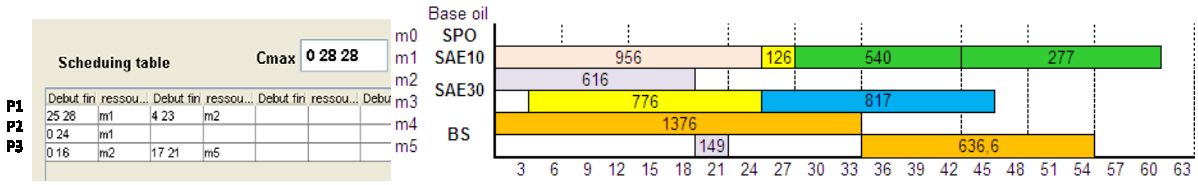


Figure A7.12. Gantt solution optimale