NNE DEMOCRATIQUE ET POPULAIRE
L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE D'ORAN

FACULTE DES SCIENCES
DEPARTEMENT D'INFORMATIQUE

MEMOIRE

Présenté par :
Melle AIT SI LARBI El Yasmine

Pour obtenir
LE DIPLOME DE MAGISTER

Spécialité : Informatique
Option : Informatique et automatique

Intitulé :

PROPOSITION D'UN MODELE POUR LA PLANIFICATION DE PRODUCTION DES ENTREPRISES MULTI SITES

Soutenu le : / /

Devant le jury composé de:

Mr BENHAMAMOUCH D.	Professeur, Université d'Oran. (Président)
Mr HAFFAF H.	Professeur, Université d'Oran. (Examinateur)
Mr BOUAMRANE K.	Maître de conférences, Université d'Oran. (Examinateur)
Mr BELDJILALI B.	Professeur, Université d'Oran. (Encadreur)
Mme AISSANI N.	Chargée de cours, Université d'Oran.

A mes très chers parents,
A ma sœur, à mon frère,
A ma grand mère

Remerciements

Je tiens avant tout à exprimer ma profonde reconnaissance à Monsieur BELDJILALI Bouziane, Professeur à lôUniversité dôOran, qui a encadré mon travail pour lôobtention du diplôme de magister, pour son suivi et ses conseils judicieux, pour les nombreuses discussions qui ont permis lôaboutissement de ce travail. Quôil trouve ici lôexpression de mon profond respect.

Je tiens à remercier Madame AISSANI Nassima, chargée de cours à lôUniversité dôOran pour ses précieux conseils, ses commentaires et ses encouragements tout au long de ce travail.

Je remercie Monsieur BENHAMAMOUCH Djilali, Professeur à lôUniversité dôOran, de môavoir fait lôhonneur de présider le jury de ce mémoire.

Je remercie également Monsieur HAFFAF Hafid, Professeur à lôUniversité dôOran ainsi que Monsieur BOUAMRANE Karim, Maître de Conférence à lôUniversité dôOran de môavoir fait lôhonneur dôêtre membres de jury de ce mémoire et de lôintérêt quôils ont porté à ce travail.

Mes remerciements à mes parents dont lôaide et la disponibilité furent si précieuses.

Que toutes les personnes qui môont soutenue et orientée soient assurées de ma profonde reconnaissance.

CHAPITRE 4 : LES SYSTEMES MULTI AGENTS POUR LA PLANIFICATION ET L'ORDONNANCEMENT DISTRIBUES

M ULTI-TIERS POUR UNE ORGANISATION HETERARCHI QUE

CHAPITRE 6 : CONCEPTION, REALISATION ET RESULTATS

RÉSUMÉ

Les exigences des marchés actuels (compétitivité, dynamique,é) ont poussé beaucoup dœentreprises à adopter de nouvelles organisations telles que les organisations multi sites afin dœaméliorer leur compétitivité en terme de qualité, de coût et de raccourcissement des délais. Dans le domaine de la production, les fonctionnalités de planification et dœordonnancement sont dœune grande importance. Ces fonctionnalités constituent les préoccupations primaires des concepteurs de systèmes de pilotage pour les entreprises multi sites, de plus elles sœavèrent très complexes dans ce contexte. Le but de notre projet est de proposer un système de planification et dœordonnancement pour une entreprise multi sites. Nous développerons ainsi une démarche de modélisation basée sur les techniques de lœintelligence artificielle (Systèmes multi agents) dotée de capacité décisionnelle par apprentissage par algorithmes génétiques dans une organisation hétérarchique pour faire de la planification à court, moyen et long terme.

Mots clés: entreprises multi sites, planification et ordonnancement, systèmes multi agents, algorithmes génétiques.

ABSTRACT

Many enterprises have to develop their organizations to become more competitive and dynamic; one of existing solutions is to develop their self to become multi sites organizations. We will present an approach of multi sites planning and scheduling.

In the production systems, the planning and scheduling tasks are very important.

These represent a worry for the designers of production control systems. More that these, features are very complex in that context.

The aim of our project is to provide a planning and scheduling system for a multi-site enterprise, we develop an approach based on artificial intelligence modelling techniques (Multi agents) with learning decision making capacity by genetic algorithms within a hétérarchique organization to make planning in short, medium and long term.

Keywords: Multi sites enterprises, planning and scheduling, multi agent systems, genetic algorithms.

INTRODUCTION

INTRODUCTION

La plupart des entreprises ont eu recours ces dernières années à une organisation multi sites afin dôaméliorer leur compétitivité en terme de qualité, de coût et de raccourcissement des délais.

Ce changement est un moyen pour ces firmes de faire face à la concurrence et de développer leurs avantages concurrentiels, tout en revoyant la structure de leur réseau logistique pour quôil soit plus performant. Cette structure se caractérise par le nombre, la localisation, la taille, la mission et le mode de fonctionnement des différentes entités qui la constituent [PIRA 06].

Ainsi, les entreprises ont recours à une organisation en réseau qui prend des formes diverses sous lôinfluence de quatre facteurs principaux :

- Le premier facteur est le passage dôun marché de lôoffre (production standardisée) à un marché de la demande (production différenciée), dôoù le raccourcissement du cycle de vie des produits en poussant les entreprises à plus de réactivité.
- Le second facteur est la globalisation de lôéconomie.
- Les innovations technologiques constituent le troisième facteur car elles complexifient les produits et ne permettent pas aux entreprises de maîtriser lôintégralité des savoir- faire de leur domaine de production.
- Et enfin, la financiarisation de lôéconomie représente le quatrième facteur [RORI 05].

Les entreprises sont ainsi contraintes de revoir leur organisation interne en adoptant des formes se rapprochant des réseaux externes. Elles se scindent en unités autonomes. De nombreux exemples de mutations dôentreprises ont été décrits dans la littérature [DAVI 95]. Les opérations de reengineering se sont multipliées, conduisant ainsi à une forme proche de lôentreprise en réseau [TARO 01].

Dôautre part, la planification stratégique a été remise en cause [MINT 93] et le processus de planification stratégique est devenu émergent [MINT 85] pour des stratégies « tâtonnantes » [AVEN 96].

réseaux, catégorie correspondant à des configurations
ributions, nous citons la firme Z de Ouchi & Jaeger cherchant une prise de décision consensuelle de [MINT 80] sôorganisant par projets. [MILE 92] parle de réseau pour réintroduire les bienfaits des marchés au sein de la firme. Pour un grand nombre de recherches, côest une forme de transversalité qui est recherchée.

Le présent travail sôinscrit dans cet optique, la réalisation dôun système pour la planification et lôordonnancement dans les entreprises multi sites qui soit le plus performant possible dans la mesure où il profite des expériences déjà faites, ceci en utilisant un système multi agents et une méthode dôoptimisation basée sur les algorithmes génétiques à travers une architecture hétérarchique.

Notre approche est alors hybride, elle prend en compte lôaspect **préventif** à travers une planification par un **algorithme génétique** et lôaspect **réactif** en utilisant un **système multi agents.**

Dans le but de présenter notre démarche, ce mémoire est organisé en six chapitres :

Un premier chapitre présente le contexte et la problématique traités: la planification de la production multi sites, les contraintes et les inconvénients de la production multi sites et notre contribution pour pallier ces derniers.

Le second chapitre est consacré à lôordonnancement et lôoptimisation de celui-ci. Nous préciserons ainsi la notion dôordonnancement en gestion de production, en présentant les méthodes de résolutions et leurs caractéristiques.

Ensuite nous passerons aux algorithmes génétiques qui sont une des méthodes dôoptimisation les plus prometteuses pour ce type de problème.

Dans le quatrième chapitre, nous parlerons des Systèmes multi- agents en présentant un bref historique de lôintelligence artificielle distribuée qui a donné naissance aux Systèmes multi agents. Nous expliquerons aussi les notions élémentaires dôagents, leurs architectures et typologie, comme nous définirons ce quôest un système multi- agents en faisant référence au modèle Aalaadin de Jack Ferber [FERB 98] que nous utiliserons par la suite.

résentés au niveau du cinquième chapitre, et leur ique en justifiant notre choix, nous présenterons aussi la plate forme J2EE, plate forme utilisée pour la réalisation de notre modèle en passant par la reconnaissance des architectures multi tiers.

Ensuite, sera abordé le chapitre concernant la conception et lôimplémentation de notre approche. Nous expérimenterons notre modèle afin dôanalyser lôapport de notre approche pour la résolution des problèmes de planification et dôordonnancement dans les entreprises multi- sites.

Enfin, nous clôturons ce mémoire par une conclusion et quelques perspectives.

CHAPITRE 1 :

ENTREPRISES DE PRODUCTION MULTI SITES

De nombreuses entreprises, qui sont des unités de production [LARO 83], ont une production répartie sur plusieurs sites et sont qualifiées d'entreprises multi- sites.

Ces unités de production s'échangent des composants pour la production d'un produit final à livrer aux clients et certaines d'entre elles fabriquent les mêmes composants ou produits finaux. Le problème de la planification multi sites concerne la répartition et la coordination des productions [THIE 03].

Nous nous intéressons dans ce chapitre au domaine de la gestion de production au sein de telles entreprises et en particulier à la fonction d'ordonnancement et de planification de la production.

2. LA GESTION DE PRODUCTION DANS L'ENTREPRISE

Depuis les années 1970 et plus généralement depuis la fin de la deuxième guerre mondiale, compte tenu de l'évolution économique et technologique, les systèmes de production industrielle se sont diversifiés et compliqués d'une manière considérable.

Nous citons par exemple les études d'ordonnancement, commencées au début des années 50 avec les travaux de Johnson [JOHN 54], de Jackson [JACK 55] et de Smith [SMIT 56] qui sont particulièrement riches et fécondes, du fait de la présence de la diversité des problèmes d'ordonnancement d'une part, et des nombreux domaines d'applications potentielles existants d'autre part (production, informatique, administration, etc.) [VACH 00].

La diversité des entreprises et des positions par rapport au marché, la complexité des traitements et la fréquence des différents types d'aléas représentent les conditions contemporaines qui assurent un niveau de complexité toujours croissant de la gestion de production. Selon Bénassy [BENA 90] "*La gestion de production, ce n'est pas difficile, c'est seulement compliqué.*"

Ainsi, de nombreuses études ont été faites pour pouvoir répondre aux besoins et aux objectifs poursuivis par les diverses entreprises de production.

Les principaux objectifs de la gestion de la production, pour quêelle soit efficace, peuvent être considérés de la manière suivante [VACH 00]:

-Le respect des délais

Cet objectif est sûrement prioritaire vis à vis du client, car les commandes ont un caractère contractuel et leur non-respect peut avoir des conséquences financières pénalisantes, aussi bien sur le plan économique que sur l'image de marque de lôentreprise. Cela implique entre autres, un challenge permanent des cycles de fabrication et des dates de lancement de fabrication.

Ainsi, il est nécessaire pour un industriel de fournir un bien à un client avec le retard le plus faible possible. Notons que le retard correspond donc à la différence entre la date de fin de fabrication ou la date de livraison et la date de fin théorique de fabrication ou la date d'engagement de livraison.

-La minimisation des stocks intermédiaires

A chaque niveau de fonctionnalité, les différents intervenants ne cernent pas de manière précise les flux et l'ensemble des paramètres entrant dans un processus complexe de fabrication et ne cherchent donc pas ou ne peuvent pas optimiser les stocks de matières et pièces en cours de fabrication ou d'assemblage. Une tendance naturelle, mais coûteuse, est d'accroître les quantités de produits intermédiaires pour éviter toute rupture d'enchaînement.

Par opposition, il est aussi important de ne pas avoir d'avance lors de la fabrication. L'avance correspond donc à la différence entre la date de fin théorique de fabrication et la date de fin réelle de fabrication.

-L'optimisation de l'utilisation des moyens

L'adéquation des charges et des capacités est un problème important, qui nécessite des outils d'aide au diagnostic et d'analyse. Il faut, en particulier, optimiser en permanence la

r la présence de goulots d'étranglement qui peuvent
n goulot d'étranglement ou goulet est le résultat dûne capacité dûne ressource inférieure à la charge résultant d'une demande.

-Autres objectifs

Outre les objectifs précédemment cités, la gestion de production doit poursuivre, simultanément, un certain nombre d'objectifs de qualité, de quantité et de coûts. Ces objectifs ne sont pas indépendants. En effet, par exemple, si l'on augmente les quantités, cela ne doit pas se faire en altérant la qualité et/ou en ayant un effet défavorable sur les coûts.

Par conséquent, le but schématique de la gestion de la production est de répondre aux questions suivantes : que produire? Quand? Comment? Par qui? A quel coût?
Les deux questions qui nous intéresseront plus particulièrement ici sont quand ?, par qui? La réponse est généralement donnée par le service planification et ordonnancement.

Nous passons à présent à la présentation du contexte de notre travail qui se situe au niveau des entreprises de production multi sites.

3. LES ENTREPRISES DE PRODUCTION MULTI SITES

3.1 Introduction :

Beaucoup dœntreprises ont eu recours ces dernières années à une organisation multi sites. Ceci est un moyen pour ces entreprises de faire évoluer leur avantage concurrentiel, en revoyant la structure de leur réseau logistique afin quœil soit le plus performant possible.
Une telle structure se caractérise par le nombre, la localisation, la taille, la mission et le mode de fonctionnement des différentes entités qui la constituent [PIRA 06].

Les entreprises revoient ainsi leur organisation interne et adoptent des formes se rapprochant des réseaux externes. Elles se scindent en unités autonomes. De nombreux exemples de mutations dœntreprises sont déjà décrits [DAVI 95]. Les opérations de

duisant à une forme proche de lôentreprise en réseau

La réorganisation des entreprises en réseau interne ou externe fait partie des réponses apportées à ce jour aux pressions exercées par plusieurs facteurs dont ceux macro-économiques [RORI 05].

Les entreprises multi sites sont alors des entreprises dotées dôune nouvelle organisation, elles sont apparues dans le but de répondre à de nouveaux besoins, car la compétitivité internationale exige:

- Une minimisation des coûts,
- Un raccourcissement des délais,
- Une amélioration continue de la qualité des produits.

Nous pouvons citer plusieurs exemples de mutations dôentreprises ([MASO 00], [JOSS 04], [JACO 06]). Dans [GALLI 07], une étude a été faite sur les facteurs déterminants et décisifs du choix des compagnies pour une organisation multi sites; nous citons : lôéconomie internationale, et lôinfluence de lôéconomie spécifique aux agglomérations.

Nous pouvons ainsi dire quôune entreprise de production multi sites est une entreprise dont lôorganisation est en réseau, c'est-à-dire que les différentes entités constituant celle-ci sont distribuées géographiquement.

Nous mentionnerons aussi lôimportance de la notion de chaîne logistique. Une chaîne logistique est un système de sous- traitants, de producteurs, de distributeurs, de détaillants et de clients entre lesquels sôéchangent les flux matériels dans le sens des fournisseurs vers les clients et des flux dôinformations dans les deux sens [TAYU 99].

Pour une entreprise multi sites, une chaîne logistique est composée alors de sites de production, chacun pouvant contenir plusieurs ateliers qui peuvent contenir à leur tour plusieurs machines de production.

ou plusieurs sous traitants (sous traitant de capacité et ne chaîne logistique, les sous-traitants de capacité sont liés par contrat à leurs donneurs dôordres. Ces contrats stipulent généralement que le donneur dôordre sôengage à sous-traiter une quantité minimum sur une période relativement longue (6 mois, 1 an) et à fournir des commandes prévisionnelles au sous-traitant. Le sous-traitant sôengage à absorber des variations par rapport aux prévisions sur le court terme. Le sous-traitant est donc dépendant des fluctuations de charge du donneur dôordres.

Il est ainsi possible de considérer trois indicateurs de performance du sous-traitant [THIE 03]:

- Le taux de service,
- Le niveau de stocks,
- La quantité de produits sous-traités pour des raisons de capacité.

La figure 1.1 montre le schéma classique dôune entreprise multi sites :

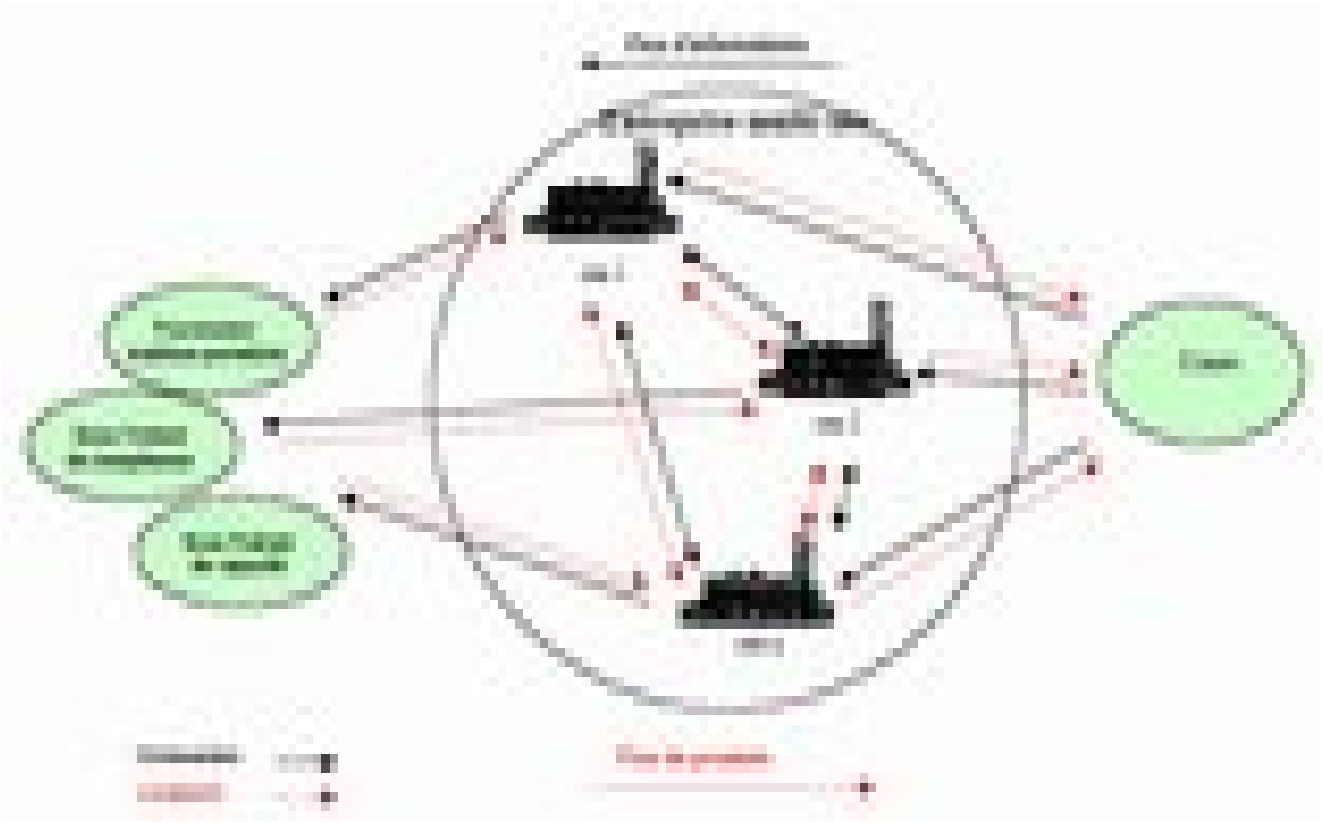

Figure 1.1 Lôentreprise multi sites

Les processus de production sont ainsi distribués sur plusieurs sites manufacturiers responsables de la production de parts variées dôensembles de produits finis [SAUE 00].

lti sites :

3.2.1. Planification :

La planification de la production dans les entreprises à site unique est une tâche facile à réaliser vu la simplicité de la topologie et surtout grâce aux recherches effectuées depuis plusieurs années sur le terrain [GRAT 05], et qui ont mené à lôapparition de plusieurs méthodes et techniques.

Parmi les techniques mises en place, nous citons :

- La technique PERT (Planning Evaluation and Review Technique) qui a été mise au point en 1959 pour lôarmé Américaine.

- La technologie juste à temps (just in time- livraison JIT) qui prend en charge la marchandise pour arriver aux sites au moment exact où ils en ont besoin; de ce fait, elle réduit les niveaux des stocks et de là, lôinvestissement et les frais associés à chacun dôeux ([TAII 88], [HIRO 06]).

- La méthode MRP II (Manufacturing Ressource Planning II), une méthode de planification des ressources de bout en bout au niveau des entreprises industrielles.

- La planification de ressources de lôentreprise (ERP) qui gère tous les processus de lôentreprise, incorporant toutes les fonctions de cette dernière (exemple: la gestion des ressources humaines, la comptabilité et la gestion financière, le support de décision, la distribution, lôapprovisionnement, le e- commerce, la gestion des chaînes logistiques (Supply Chain Management) et les informations physiques). Ceci vise lôoptimisation du processus de commande, de production et de livraison [SHEI 02].

Des méthodes avancées ont par ailleurs émergé, parmi lesquelles nous pouvons citer les systèmes de planification avancés (APS) de lôensemble des flux (informations matérielles et financières, é) qui synchronisent et optimisent les activités et les interfaces, tout en permettant dôatteindre la compréhension et la collaboration, en termes dôobjectifs de services habituels et de marges dôactivités. ([TEMP 01]; [MEYR 05])

ı de la production dans une entreprise :

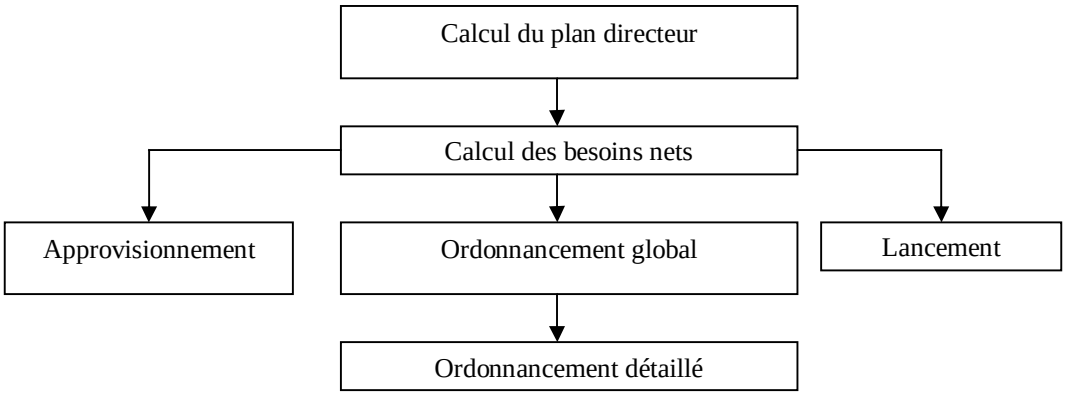

Figure 1.2 La planification de la production

Cela dit, au niveau des entreprises multi sites, la planification nœst pas une tâche aussi facile que pour une entreprise à site unique de production. En effet, l'activité de planification consiste à concevoir pour l'entreprise un "futur souhaitable" et de disposer de moyens nécessaires pour parvenir à la réalisation de ce futur. Cœst une activité de choix parmi différentes possibilités. Elle s'exprime au travers de plans et elle est une préparation à l'action tout en considérant et en respectant plusieurs contraintes que nous allons détailler par la suite.

Au niveau tactique, la planification vise une programmation prévisionnelle de la production, des approvisionnements et de la distribution, à partir de demandes commerciales prévues ou réelles, en conformité avec les décisions effectuées au niveau stratégique. Ces décisions prises aux niveaux supérieurs, imposent notamment des contraintes sur le processus de production (contraintes potentielles entre les tâches ou contraintes de conservation au niveau des stocks) et sur les ressources (contraintes de capacité) qui doivent être prises en compte au niveau de la planification [THIE 03].

La figure 1.3 montre la planification au sein dœune entreprise multi sites :

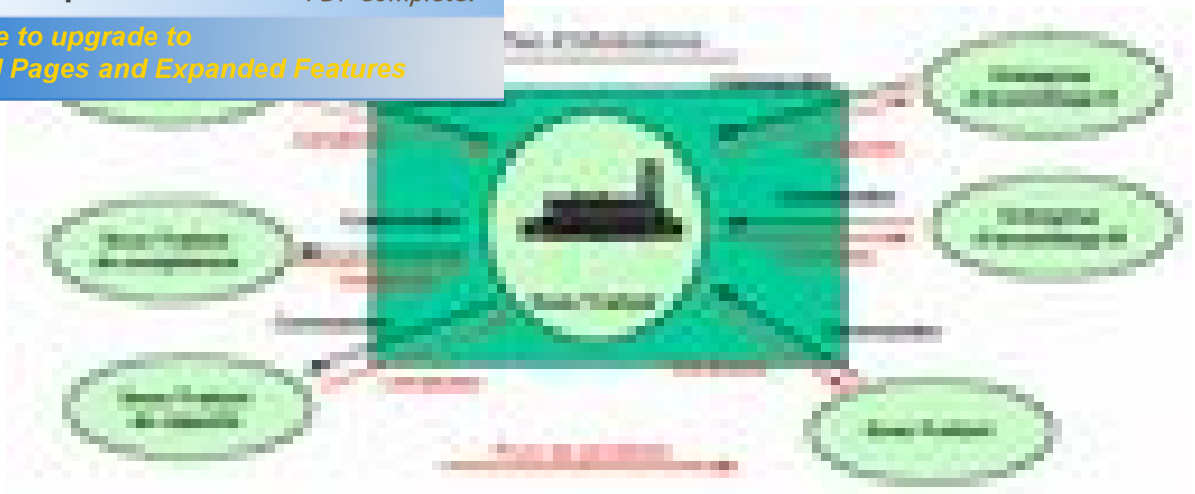

Figure 1.3 Un exemple de planification pour un acteur de la chaîne logistique

Pour les entreprises ayant choisi une telle organisation (multi sites), les méthodes de planification sont plus complexes car elles dépendent de l'allocation et de la coordination des productions dans de multiples sites [THIE 03].

Plusieurs chercheurs se sont fixés comme objectif de résoudre ce problème. Nous citons par exemple les travaux de [BENA 06] recherchant pour chacun des sites d'une entreprise la quantité optimale pour chacune des ressources, en utilisant un système multi agents (SMA) dans lequel un agent nommé agent prévention prend des décisions de sur -stockage ou de sous -stockage à partir de sa base de connaissances, et l'implémentation d'un web service pour l'entreprise.

Nous avons remarqué qu'il n'y a pas d'adéquation entre le court, le moyen et le long terme car les décisions n'étaient prises que pour permettre la prise en charge de l'état des stocks.

Il y a aussi [THIE 03], qui traite de la gestion de chaînes logistiques et propose des modèles et mises en oeuvre pour l'aide à la décision à moyen terme seulement. Ce problème de planification concerne la répartition et la coordination des productions dans une entreprise multi sites. Il a été abordé en considérant une entreprise où il existe un service logistique central et où la planification au niveau de chacun des sites de production est effectuée de manière locale avec un module de type MRP. Les CSP (Constraint Satisfaction Problem) ont

ntration a été faite beaucoup plus sur les acteurs de la de lôentreprise.

Quant à [SAUE 00], ils proposent une architecture hiérarchique sur laquelle sont implémentés des groupes dôagents. Cette architecture ne résout cependant pas tous les problèmes dans la mesure où les seuls liens qui existent sont des liens hiérarchiques verticaux.

Nous constatons que lôarchitecture hiérarchique nôest pas dôun grand apport car il nôy a pas de communication entre des entités dôune même classe pour résoudre des problèmes de bas niveau (exemple: transfert vers une autre ressource en cas de panne) dans des délais acceptables. Ce travail est néanmoins une expérimentation intéressante dans lôutilisation des SMAs facilitant la gestion de la communication entre les sites.

Nous avons aussi remarqué quôau niveau de ces travaux, la prise en compte du court, moyen et long termes nôétait pas un objectif commun pour la gestion des entreprises multi sites.

En conséquence, il nous semble intéressant de proposer dans ce présent mémoire, une approche de planification multi sites qui prend en charge les besoins de lôentreprise sur le court, le moyen et le long terme, c'est-à-dire un système de planification et dôordonnancement par lôimplémentation dôun SMA doté de capacités de prise de décisions à travers une méthode dôapprentissage supervisé au moyen des algorithmes génétiques (AG).

Ces agents sont organisés selon une architecture hétérarchique, car celle-ci permet une meilleure prise en charge du court terme, du moyen terme et du long terme. Nous aborderons dans la prochaine section la problématique de la production multi sites et les contraintes à satisfaire pour sa modélisation.

3.2.2. Les contraintes de la planification multi sites :

Les fonctions de production comme la gestion des stocks et des ressources ou la planification et lôordonnancement dans les entreprises multi sites relèvent des problèmes de répartition et de coordination des productions dans les sites ou unités de production [THIE 03].

tralisées pour tous les plans est nécessaire si plusieurs niveaux dôordonnancement ayant des systèmes dôordonnancement spécifiques coopèrent dans un environnement manufacturier dynamique et distribué.

Lôincertitude de la situation et les buts différents des plans individuels doivent être pris en compte dans différents niveaux [SAUE 00]. Il existe ainsi des problèmes distribués que seule une résolution distribuée spécifique peut résoudre, comme par exemple, la gestion décentralisée dôun réseau électrique [RICO 01].

Nous faisons face, dans notre contexte, à un environnement distribué. Un système de pilotage distribué est défini comme suit : « *Les systèmes de pilotage distribués sont un ensemble d'unités de décisions locales assurant une réponse réactive au système. Généralement, ces systèmes distribués nécessitent un coordinateur qui améliore la collaboration dynamique d'entités intelligentes, ayant pour but d'atteindre des objectifs locaux et globaux.* » [MAIO 01].

De par la distribution des processus de production, quelques problèmes spécifiques apparaissent en plus des problèmes dôenvironnement dôordonnancement dynamique et complexe:

- Indépendance entre les processus de production qui sôexécutent dans différents plans pris en considération.

- Dans lôordonnancement global, des données généralisées et imprécises sont utilisées à la place de données exactes.

- Des systèmes dôordonnancement existants (locaux) pour des plans individuels qui accomplissent une réalisation locale pour des demandes globales peuvent être intégrés.

- La coordination des activités décentralisées dôordonnancement pour tous les plans dans une entreprise est nécessaire si plusieurs niveaux dôordonnancement avec leurs propres systèmes dôordonnancement spécifiques doivent travailler en coopération dans un environnement manufacturier dynamique et distribué.

- Lôincertitude de la situation actuelle des plans individuels doit être prise en compte.

en considération à différents niveaux [SAUE 00].

Une classification des contraintes liées aux entreprises multi sites a été proposée dans [THIE 03]. Nous pouvons l'expliciter ainsi :

3.2.2.1 Contraintes temporelles:

Comme tous les systèmes manufacturiers, les entreprises multi sites ont pour objectif de produire le plus possible et de répondre aux requêtes dans un temps opportun. Cela dit, ces entreprises ont certaines contraintes plus spécifiques que d'autres, en relation avec le temps: car la grandeur de la taille de leur réseau décisionnel peut être coûteuse en termes de temps à cause du nombre d'entités décisionnelles.

Il faut aussi prendre en considération le temps de transfert de produits dans le cas des entreprises internationales.

Dans ce but, plusieurs approches se sont focalisées sur l'aspect temps au niveau de la planification dans les entreprises multi sites [THIE 03].

2.2.2.1.1 Modèles par date de début:

Dans les modèles par dates de début, les variables de décision sont les dates correspondant aux tâches à effectuer au plus tôt au niveau des différentes ressources.

Chaque ressource ne peut exécuter qu'un nombre limité de tâches en raison des contraintes de capacité, de précédence et de succession.

En respectant les dates d'échéances, [KOLI 01] a utilisé ce modèle d'ordonnancement dans une unité de production.

Ce modèle a été aussi utilisé par [AITS 08 a][AITS 08 b] avec une approche génétique pour un contexte d'ordonnancement multi sites.

3.2.2.2 Contraintes de quantités et d'espace:

Elles sont liées aux capacités des chaînes logistiques: leurs capacités de ressources et de stocks.

Plusieurs modèles ont été développés par rapport à ces contraintes, ils sont utilisés parfois pour la planification à moyen terme.

Les contraintes concernent la conservation de stocks (date et quantité), la dépendance des produits, la capacité dont on a besoin pour un autre produit et la capacité des ressources en termes de quantités: chaque ressource ne peut pas être utilisée pendant plus dûne durée précise, exemple [FONT 01].

Notons que la gestion de chaînes logistiques (SCM) affecte lôorganisation toute entière, de la planification de production à la gestion de clients [VACH 00].

Notons aussi que la gestion de production, et en particulier la gestion des stocks et des ressources pour un magasin ou pour un site local, est actuellement une tâche réalisable. Cette gestion, quand elle sôapplique à une entreprise étendue, où les ressources sont distribuées, devient plus complexe [BENA 06] car les problèmes dôordonnancement sont généralement traités dans un seul environnement, où un certain ordre de produits suit un plan dôordonnancement sur un ensemble de machines [SAUE 00]. Ceci dit, comme nous lôavons précisé précédemment, le problème de la planification multi sites se présente en termes de répartition et de coordination des productions dans une entreprise multi sites [THIE 03].

Nous présentons dans ce qui suit, lôarchitecture hétérarchique appliquée à une organisation multi sites:

4. DE L'ORGANISATION MULTI SITES VERS UNE ARCHITECTURE HETERARCHIQUE

Dans un système, on distingue deux mécanismes dôagencement complémentaires. Nous avons:

-Un mécanisme dôagencement vertical, faisant référence à la notion de hiérarchie: une hiérarchie correspond à un agencement vertical dôentités composant un système global où la priorité est accordée, à un niveau i de la hiérarchie, à lôaction ou au droit dôintervention sur une entité de niveau i+1 et où la performance du niveau i+1 conditionne celle du niveau i.

ontal faisant référence à lôabsence de toute hiérarchie qualifié de structure « hétérarchique » où les entités se situent au même niveau hiérarchique i, même si le sens original du terme signifie absence totale de hiérarchie. Nous parlerons ainsi *d'hétérarchie au sens strict du terme* dans le second cas [TREN 02].

Lôarchitecture hiérarchique présente plusieurs inconvénients. Elle est limitée en termes dôévolutivité, de reconfigurabilité, de fiabilité et de redondance.

Lôarchitecture hétérarchique qui, à la base, est une organisation hiérarchique possédant des liaisons entre les entités dôun même niveau, permet dôobtenir une meilleure réactivité, une réduction des coûts, etc. [PIRA 06].

Le terme hétérarchie décrit ainsi la relation entre les entités dôun même niveau. Initialement proposée dans le domaine de la biologie médicale (Mc Culloch 1945), elle a été expérimentée dans de nombreux domaines (Prabhu 2003, Haruni et Kawato, 2006).

Lôorganisation hétérarchique offre un grand potentiel de communication inter entités de décision de même niveau, ce qui encourage une plus grande réaction on-line. En plus de cela, elle permet lôutilisation de lôarchitecture hiérarchique pour la prise en charge de la notion de long terme. En bref, il apparaît que lôorganisation hétérarchique est lôorganisation idéale pour les systèmes de pilotage on-line [AISS 08 b].

La communauté scientifique internationale considère les systèmes hétérarchiques comme lôun des enjeux mondiaux, tant au niveau dôun réseau dôentreprise, dans le domaine de la production de biens, que dans celui de la production de services. Le principal but de lôutilisation de lôapproche hétérarchique réside dans sa capacité à faciliter la mise en oeuvre de mécanismes relatifs à la vue évolutionniste des systèmes, tels que lôadaptation et lôauto-organisation (les approches hiérarchiques au sens strict ne permettent pas dôintégrer ces mécanismes) [TREN 02].

Ceci nous amène à son utilisation dans notre contexte qui concerne la planification et lôordonnancement de la production multi sites.

pour lôarchitecture dôun système hétérarchique, la gents est préférable, car lôadéquation entre un agent et le concept de pilotage hétérarchique résulte du fait que la définition minimale dôun agent est celle dôune entité qui perçoit son environnement, agit sur celui ci et se comporte de manière rationnelle (raisonnement).

Par conséquent, il est aisé et efficace dôadapter cette approche au pilotage hétérarchique. A ce propos, il est possible de citer lôexemple de lôexpérimentation faite par [AISS 06] pour lôordonnancement des ateliers Job shop.

La figure 1.4 donne plus de précisions en ce qui concerne lôarchitecture hétérarchique [TREN 02]:

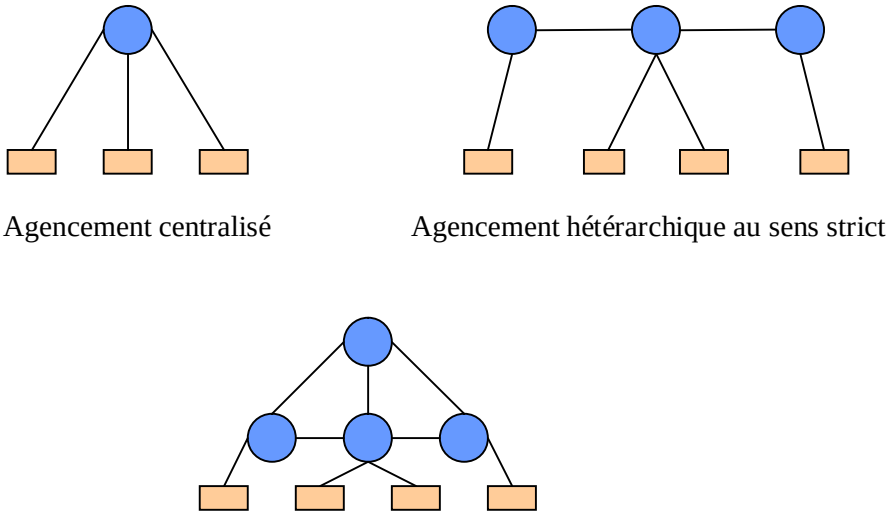

Figure 1.4 Les architectures: centralisée/hiérarchique/ hétérarchique

La figure 1.5 montre un exemple dôune entreprise multi sites ayant une architecture hétérarchique.

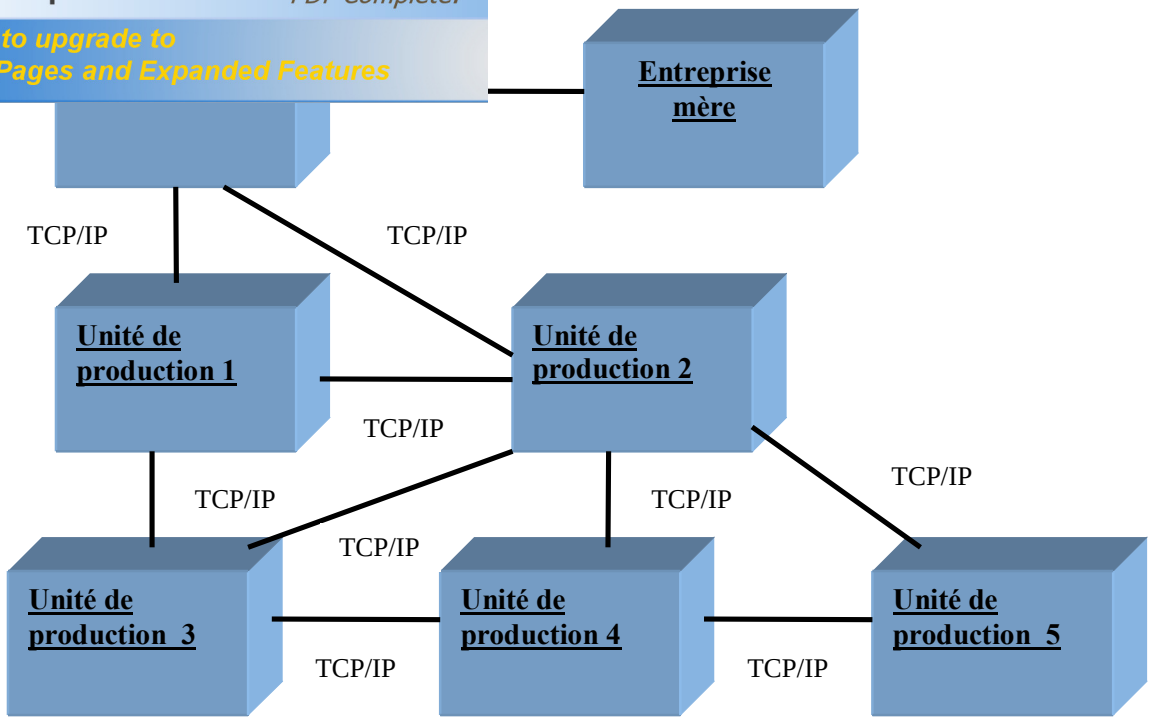

Figure 1.5 Les entités dôune entreprise multi sites suivant une architecture hétérarchique (cas de cinq postes client)

5. CONCLUSION

A travers ce chapitre, nous avons présenté la notion de gestion de la production et ses buts principaux. Nous nous sommes intéressés ensuite aux entreprises de production multi sites de façon générale, en nous concentrant sur les besoins de telles entreprises en matière de planification et dôordonnancement de la production au niveau du court, du moyen et du long terme.

Nous avons aussi présenté lôarchitecture que nous avons choisie pour lôimplémentation de notre système, à savoir, lôarchitecture hétérarchique.

Nous présenterons dans le chapitre suivant la fonction dôordonnancement de la production et son optimisation.

CHAPITRE 2 :

ORDONNANCEMENT ET OPTIMISATION

Parmi les différentes fonctions de la gestion de la production, nous nous intéressons plus particulièrement à la fonction d'ordonnancement.

L'ordonnancement, dans le domaine de la gestion de la production, consiste à placer les différentes opérations d'un ensemble de lots en respectant les contraintes de ressources et de précédences afin de permettre la fabrication de produits sur les machines d'un atelier de façon optimale. Ceci est un problème complexe qui a fait l'objet de nombreuses études, notamment en Intelligence Artificielle car il est difficile de le résoudre efficacement avec des algorithmes classiques.

Dôune manière plus générale, un problème d'ordonnancement consiste à affecter des tâches aux ressources et à décider de leur répartition dans le temps, de manière à optimiser un critère ou à trouver un compromis entre plusieurs critères [VACH 00].

Dans notre contexte, qui est la production multi sites, nous pouvons dire que lôordonnancement multi sites consiste à placer les opérations au niveau des différents sites contenant chacun plusieurs ateliers et machines, dôoù lôimportance de la détermination dôune bonne solution en un temps raisonnable. Si, en plus, cette solution est optimale, alors la réalisation de lôobjectif principal qui est de trouver un bon ordonnancement qui puisse satisfaire un maximum de contraintes du système est possible.

Dans la littérature, plusieurs approches ont été proposées pour résoudre le problème dôordonnancement. Ces approches que nous développerons par la suite, utilisent des méthodes et des algorithmes.

Il existe des algorithmes partiellement aléatoires qui permettent d'améliorer la recherche d'optimum dans des problèmes complexes tels que la méthode du gradient, la méthode du recuit simulé, et aussi des algorithmes basés sur les lois de la génétique appelés algorithmes génétiques.

pe de l'évolution d'une population candidate composée
as, il s'agit de la représentation d'un ordonnancement.

Cette technique permet d'explorer l'espace des solutions et d'aboutir à lôune dôentre elles après un nombre plus ou moins important de générations. Chaque génération correspond à une évolution de la population par croisements, mutations et sélections des meilleures solutions. Cette méthode peut comporter un ou plusieurs objectifs à améliorer (les temps de cycle, le retard, l'avance é), des contraintes et des objectifs que l'entreprise souhaite atteindre.

2. L'ORDONNANCEM ENT

L'ordonnancement est une fonction essentielle en gestion de production. C'est un problème difficile en raison du nombre de calculs à effectuer pour obtenir un ordonnancement qui optimise le critère retenu. En outre, il existe de nombreux problèmes d'ordonnancement où diverses méthodes exactes ou approchées ont été proposées pour résoudre une partie d'entre eux. Nous allons tout dôabord définir l'ordonnancement.

2.1 Définition de l'ordonnancement :

« Ordonnancer un ensemble de tâches, côest programmer leur exécution en leur allouant les ressources requises et en fixant leurs dates de début. » [GOTH 93]
Cette programmation se base sur des variables qui sont les tâches, les ressources, les contraintes et les objectifs.

Face à un nouveau problème dôordonnancement, il faut dôabord préciser à quoi correspondent ces notions pour pouvoir décider ensuite des méthodes de résolution à appliquer. Dans le tableau ci-dessous, nous illustrons lôimportance de cette étape en montrant les différents sens que les notions de tâche et de ressource peuvent avoir selon le domaine dôapplication.

[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]

Tableau 2.1 : Exemple d'interprétation des notions de tâches et de ressources en fonction du domaine d'application

Ainsi, on retrouve l'aspect commun de l'affectation de tâches dans le temps et dans l'espace au moindre coût et en un temps raisonnable.

Il y a problème d'ordonnancement :

- Quand un ensemble de travaux (jobs ou lots) est à réaliser,
- quand cette réalisation est décomposable en tâches (ou opérations),
- quand le problème consiste à définir la localisation temporelle des tâches et/ou la manière de leur affecter les moyens nécessaires.

Pour être plus précis, nous avons opté pour la définition de l'ordonnancement donnée par [GOTH 02] [ESQU 99]:

« *L'ordonnancement est la programmation dans le temps de l'exécution d'une série de tâches (ou activités, opérations) sur un ensemble de ressources physiques (humaines et techniques), en cherchant à optimiser certains critères, financiers ou technologiques, et en respectant les contraintes de fabrication et d'organisation*. »

Notons qu'un ordonnancement est qualifié de statique (predictive scheduling) s'il est calculé à partir de données prévisionnelles. Le calcul est lancé en mode hors ligne ou déconnecté (off- line) en ce sens que les données sur les ressources ne sont pas obtenues en temps réel et que les solutions calculées ne sont pas appliquées immédiatement.

ent dynamique (reactive scheduling) est effectué en
s au fur et à mesure quôelles arrivent.

Les objectifs de lôordonnancement imposent de mesurer la qualité dôune solution à répondre au problème étudié. Ainsi, des critères dôefficacité sont définis et utilisés pour isoler les solutions optimales ou satisfaisantes. Les critères généraux les plus souvent employés sont la minimisation de la durée totale de fabrication, des temps dôinactivité des machines, du coût de lôordonnancement (ex : le nombre de produits en cours peut conduire à des coûts de stockage supplémentaires) ou le respect des dates de fin des tâches à accomplir. Certains cas particuliers peuvent nécessiter des critères plus spécifiques (la minimisation de goulot dôétranglement sur une ressource é) [TRAN 01].

Dans ce qui suit, nous présentons différentes approches pour la résolution de problèmes dôordonnancement.

2.2 Différentes approches d'ordonnancement

Dans la littérature, plusieurs approches dôordonnancement dans un milieu incertain ont été proposées. [MEHT 99] et [DAVE 00] ont proposés deux classifications différentes de ces approches. Une autre classification a été proposée dans [CHAA 09] permettant de couvrir toutes les approches à savoir : approches proactives, approches réactives (dynamique) et les approches hybrides qui sont composées de deux approches telles que les approches prédictivesï réactives et les approches proactivesï réactives (figure 2.1).

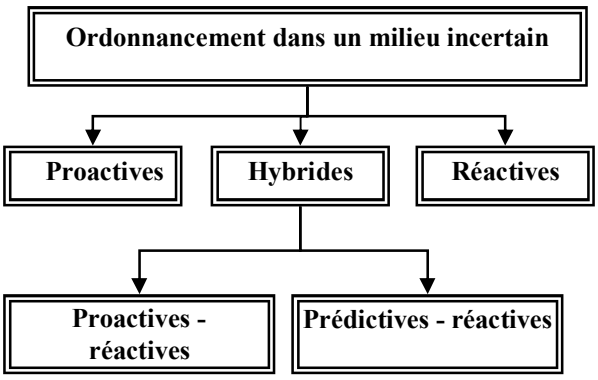

Figure 2.1 Classification des différentes approches dôordonnancement

Les approches proactives tentent de prendre en compte l'incertain lors de la phase d'ordonnancement hors-ligne uniquement. Il s'agit d'anticiper les incertitudes, en jouant sur la flexibilité, de sorte à produire un ordonnancement, ou une famille d'ordonnancements, relativement insensible aux incertitudes. Un modèle d'incertitudes est pour cela généralement supposé disponible.

2.2.2. Approches réactives

Les méthodes d'ordonnancement réactives sont parfois qualifiées de « totalement réactives », « réactives pures », ou « dynamiques ». On parle aussi de pilotage réactif. Ces approches réactives sont souvent utilisées dans des environnements perturbés, où les incertitudes sont fréquentes et de fortes amplitudes. Dans un tel environnement, un ordonnancement de référence peut rapidement s'avérer de mauvaise qualité ou même pas faisable.

Par conséquent, on suppose qu'il n'existe pas de tel ordonnancement, et que toutes les décisions sont prises en temps réel, en utilisant des stratégies qui privilégient la rapidité des décisions sur leur qualité. Ces approches présentent plusieurs avantages, dont le principal réside dans le fait que les prises de décision sont très rapides et intuitivement faciles à comprendre pour les utilisateurs.

Parmi les méthodes utilisées pour la résolution de ces approches, il existe de simples règles de priorité qui consistent, quand une ressource devient disponible, à sélectionner une activité dans la file dôattente suivant un certain critère.

2.2.3. Les approches hybrides

2.2.3.1. Approches prédictives - réactives

Les approches prédictives ï réactives sont souvent référencées dans la littérature pour faire face à des aléas. Ces approches sont constituées de deux phases. La première phase est un ordonnancement déterministe construit hors-ligne. Durant la deuxième phase, cet ordonnancement est ensuite utilisé, en ligne, pour guider les décisions. Il est alors éventuellement adapté en temps réel, pour tenir compte des perturbations, lorsque sa flexibilité temporelle ne permet plus dôabsorber un aléa.

…se sur la réponse aux deux questions « Quand réï …nancer ? ».

Quand ré–ordonnancer ?

Plusieurs cas de réï ordonnancement ont été proposés dans la littérature, parmi lesquels nous pouvons citer :

- Réï ordonnancer à chaque fois quôun événement inattendu apparaît
- Réï ordonnancer à chaque fois quôun nouveau job arrive. Cêest le cas dans [BIER 99] où les auteurs présentent un algorithme génétique réutilisant la solution courante pour résoudre un problème de Job ï Shop chaque fois quôun nouveau job arrive
- Réï ordonnancer à des intervalles de temps réguliers. Lôordonnancement est moins performant que dans la première solution. Cette approche est utilisée dans les travaux de [CHUR 92].
- Réï ordonnancer dés que le nombre dêarrivée de jobs atteint un seuil donné. Cêest le cas dans [VIEI 00] où les auteurs étudient le problème à une machine dans le quel les activités arrivent dynamiquement. Ils montrent que Réï ordonnancer fréquemment permet dêobtenir de meilleures performances du système mais entraîne une augmentation du nombre de réglages, tandis que des ré- ordonnancements moins fréquents augmentent les temps de cycles.

Comment ré–ordonnancer ?

Pour cette question, quatre réponses sont proposées :

- **La méthode de décalage à droite**: consiste à retarder lêexécution des tâches posant problème, en maintenant la stabilité de lêatelier [SMIT 95].
- **Ré–ordonnancement total:** consiste à décomposer le problème en sousï problèmes qui sont résolus en séquence.
- **Ré–ordonnancement multi- objectif:** qui sêintéresse à la recherche dôun compromis entre la performance du nouvel ordonnancement produit et la stabilité de lêatelier. [WU 93].
- **Ré–ordonnancement par retour vers l'ordonnancement initial (match-up rescheduling):** qui consiste à ne modifier que sur une période de temps, la plus réduite

nnancement de référence, de sorte que ce dernier soit la période [AKTU 99].

Les méthodes prédictivesï réactives sont généralement plus performantes que les méthodes réactives pures. Pour améliorer cette performance, il est possible de tenter dôanticiper les perturbations, soit lors de la phase hors-ligne, soit en ligne, en tirant par exemple parti dôinformations disponibles sur les incertitudes [LAHO 05].

2.2.3.2. Approches proactives – réactives

Dôaprès [ELKH 03] ces approches sont constituées de deux phases:

- Phase 1 : Un ordonnancement robuste tenant compte de lôaspect incertain de certaines données est construit par un algorithme statique ou horsï ligne.
- Phase 2 : Cet ordonnancement est ensuite adapté en fonction de lôétat du système au moment de son exécution grâce à un algorithme dynamique (en ï ligne).

La plupart des méthodes de résolution de ces approches permettent de construire un ensemble dôordonnancements statiques tels quôil soit facile de passer de lôun à lôautre en cas dôaléa.

Lô**OR**donnancement dô**A**telier **B**asé sur lô**AI**de à la **D**écision (ORABAID), est lôune des méthodes qui se basent sur la recherche de groupes de tâches permutables sur chaque ressource, cô est-à-dire que toutes les tâches dôun même groupe sont totalement permutables sans diminuer la performance souhaitée. Le résultat est donc un ensemble dôordonnancements obtenus en énumérant toutes les permutations possibles au sein de chaque groupe, parmi lesquels le décideur peut choisir en temps réel celui quôil souhaite mettre en place en fonction de ses préférences, des contraintes non modélisées, ou des aléas.

[ALOU 02] proposent un algorithme génétique construisant plusieurs ordonnancements équivalents et garantissant un bon compromis entre flexibilité et performance. Ensuite, à chaque fois quôune décision doit être prise suite à un aléa (exemple: retards de livraison de matières premières), plusieurs alternatives sont proposées au décideur.

ntes approches de lôordonnancement dans un milieu
s, approches réactives (dynamique) et les approches hybrides qui sont composées de deux approches telles que les approches prédictivesï réactives et les approches proactivesï réactives.

Nous passons à la présentation des différentes méthodes et algorithmes pouvant être implémentés au niveau des approches que nous avons vues :

2. 3 Méthodes de résolution de problèmes d'ordonnancement

Nous allons présenter les méthodes de résolution de problèmes dôordonnancement, qui se divisent en deux grands axes [TRAN 01]: les méthodes exactes et les méthodes approchées.

2.3.1 Les méthodes exactes :

Ces méthodes sont issues de la recherche opérationnelle; elles sont dites exactes car leur convergence vers une solution optimale à un problème donné a été démontrée.
Elles se basent pour cela sur des calculs mathématiques complexes et coûteux qui rendent difficile leur mise en î uvre et leur utilisation en dehors de cas très spécifiques en nécessitant des ressources calculatoires importantes. Nous citons parmi ces méthodes, celle du *Branch and Bound.*

Remarquons que les méthodes exactes ne permettent pas de représenter toutes les contraintes pouvant exister sur les jobs dans un contexte dôatelier de production, ni de prendre en compte les préférences contextuelles (fonctions objectifs multi variables) du responsable dôatelier.

2.3.2 Les méthodes approchées :

Ces méthodes se basent sur les moyens dôobtenir une solution satisfaisante plutôt quôoptimale. La solution recherchée doit vérifier un certain nombre de caractéristiques avec un coût calculatoire raisonnable. Ces méthodes procèdent souvent à des simulations informatiques pour démontrer leur efficacité. Elles peuvent être classées en cinq catégories :

ve: Ces méthodes procèdent à la construction dûne partielle complétée au fur et à mesure de la résolution.

-Méthodes par voisinage: Consistent à explorer lôespace des solutions dôordonnancement en partant dûne solution initiale complète de laquelle on extrait une solution voisine. Nous citons la méthode du recuit simulé et la méthode tabou.

-Méthodes par décomposition: Procèdent à la décomposition du problème dôordonnancement afin de faciliter sa résolution. Ces décompositions sont: la décomposition hiérarchique, la décomposition structurelle, la décomposition de lôensemble de solutions du problème, la décomposition temporelle, et enfin, la décomposition spatiale. Notons quôil existe aussi des méthodes issues de la combinaison de deux ou de plusieurs méthodes.

-Méthodes par relaxation des contraintes : Elles sont utilisées afin de favoriser lôobtention dûne solution. La solution obtenue sera proche de la solution idéale, moins optimale mais réalisable.

-Méthodes liées à lôintelligence artificielle (I.A.): Leur principale caractéristique commune se résume à leur origine I.A. Nous distinguons parmi ces méthodes plusieurs classes :

-Les systèmes à base de connaissance (SBC) dotés de capacité dôapprentissage;
-Les règles de priorité ;
-La propagation des contraintes ;
-Les algorithmes génétiques, qui font lôobjet du prochain chapitre.

3. CONCLUSION :

Dans ce chapitre, nous avons présenté une des fonctions les plus importantes en gestion de production qui est la fonction dôordonnancement.

Nous avons aussi explicité les différentes approches utilisées pour résoudre ce problème dans un milieu incertain.

Le chapitre suivant est consacré à la méthode dôoptimisation que nous avons choisie pour résoudre notre problème de planification et dôordonnancement multi sites: les algorithmes génétiques.

CHAPITRE 3 :

LES ALGORITHMES GENETIQUES

Les techniques d'optimisation présentent un intérêt constant du fait de leur capacité à améliorer de façon considérable les performances et/ou la conception des systèmes auxquels elles sont appliquées. Il existe des méthodes qui, tout en restant très efficaces dans beaucoup de situations, ne permettent pas d'aboutir à une solution pratique lorsque les problèmes abordés atteignent une taille et/ou une complexité importante. Afin de répondre à ces problèmes, des approches heuristiques spécifiques à chacun de ces problèmes ont été développées.

Une alternative à ces méthodes peut être trouvée dans le développement des algorithmes basés sur des méthodes aléatoires de recherche de solutions: gradient, recuit simulé, algorithmes génétiques (AG), ...

Caractérisés par leur propre théorie et structure, les AG se sont différenciés des autres techniques d'optimisation traditionnelles. Aussi, divers problèmes reconnus difficiles ont été résolus par les AG, tels que le problème du Voyageur de Commerce, celui de la Classification, celui d'Ordonnancement é [VACH 00]

Les algorithmes génétiques proposent de reproduire les mécanismes dœévolution et dœadaptation génétique de la vie pour optimiser des problèmes complexes dont les données peuvent varier au cours du temps. Ainsi, dans ce qui suit, nous allons présenter les notions préliminaires relatives aux algorithmes génétiques ainsi que leur fonctionnement.

2. DEFINITIONS

Parmi les définitions des AG, nous en avons retenu deux: la première définition nous renseigne sur le fondement des AG et la seconde la complète en nous renseignant sur leur utilisation.

Définition 1 :

« *Les Algorithmes génétiques sont des algorithmes d'exploration fondés sur les mécanismes de la sélection naturelle et de la génétique, utilisant le principe de la survie des structures les mieux adaptées et les échanges d'informations pseudo aléatoires* » [GOLD 94].

Définition 2 :

« *Les algorithmes génétiques sont des métaphores biologiques inspirées des mécanismes de l'évolution Darwinienne et de la génétique moderne et utilisées comme outils d'optimisation ou de recherche combinatoire*» [REND 94].

3. HISTORIQUE

En 1859, lôouvrage *« **The Origin Of Species** »*, écrit par le biologiste **Charles Darwin**, est paru. Lôauteur a exposé sa théorie de lôévolution qui repose sur les deux postulats suivants: *« Dans chaque environnement, seules les espèces les mieux adaptées perdurent au cours des temps, les autres étant condamnées à disparaître ».*

« Au sein de chaque espèce, le renouvellement des populations est essentiellement dû aux meilleurs individus de l'espèce » [GOLD 94].

Au début des années 1960, le professeur J. Holland [HOLL 75] a entamé une étude à lôuniversité du Michigan. Ses recherches visaient deux principaux objectifs :

1- Mettre en évidence et expliquer rigoureusement les processus dôadaptation des systèmes naturels.

2- Concevoir des systèmes artificiels possédant les propriétés les plus importantes des systèmes naturels.

En 1975, J.Holland [HOLL 75] a exposé les résultats de ses recherches dans *« **Adaptation In Natural And Artificial System** »* où il a expliqué les fondements des AGs, calqués sur les principes de Darwin, et a prouvé théoriquement leur robustesse dans lôexploration dôespaces complexes. Ces travaux ont suscité un intérêt croissant pour les mathématiciens tel que Koza qui a validé rigoureusement leurs mécanismes [LABE 07].

4. CARACTERISTIQUES DES ALGORITHMES GENETIQUES

Les algorithmes génétiques (AG) sont des procédures robustes d'exploration d'espaces complexes.

Un grand nombre de travaux établissent la validité de la technique pour les applications d'optimisation de fonctions et de contrôles. Ayant été reconnus comme une approche valide des problèmes nécessitant une exploration performante et économique du point de vue du calcul, les AGs sont maintenant appliqués plus largement, aux domaines des affaires, à la recherche scientifique en général, ainsi que pour l'ingénierie.

Ces algorithmes sont simples du point de vue du calcul, mais sont cependant très performants dans leur recherche d'amélioration. De plus, ils ne sont pas fondamentalement limités par des hypothèses contraignantes sur le domaine d'exploitation (hypothèses concernant la continuité, l'existence de dérivées, etc).

Un AG peut être considéré comme un processus aléatoire. Cependant, les informations qui viennent de fonctions objectives sont toujours utilisées pour paramétrer ce processus. Le travail de recherche commence en plusieurs points dans l'espace des solutions et non pas en un point singulier comme dans la plupart des techniques d'optimisation. Par les opérations de sélection, de mutation et de croisement, les bons membres de cette population survivent et peuvent ainsi passer à la prochaine génération.

A la fin de l'algorithme, une meilleure solution est engendrée parmi les membres suivants. Les caractéristiques des algorithmes génétiques peuvent se résumer à travers les points suivants :

-Les AGs travaillent sur un codage de l'ensemble des paramètres et non pas avec les paramètres eux-mêmes;

-Les AGs cherchent l'optimum à partir d'une population de points et non à partir d'un seul point;

-Les AGs utilisent l'information (ou le coût) de la fonction objectif et non pas les informations provenant des fonctions dérivées de quelque ordre que ce soit;

-Les AGs utilisent des règles de transition probabilistes et non déterministes [VACH 00].

5. VOCABULAIRE UTILISE PAR LES ALGORITHMES GENETIQUES

Voici les termes utilisés au niveau des algorithmes génétiques en parallèle avec ceux utilisés pour la nature et la génétique :

	Algorithme génétique
chromosome	Chaîne ou chromosome
gène	Trait, caractéristique ou détecteur
allèle	Valeur de caractéristique
locus	Position dans la chaîne
génotype	structure
phénotype	Ensemble de paramètres
épistasie	Non linéarité

Tableau 3.1 Vocabulaire utilisé par les AG

Une population est composée d'un ensemble d'individus où chaque individu (membre) est caractérisé par un seul chromosome (tous les AG réalisés jusqu'à maintenant utilisent cette caractéristique). Un chromosome correspond à une chaîne de caractères (gènes) qui sont caractérisés par une position sur le chromosome (locus) et une valeur du gène (allèle). Chaque gène a la possibilité de se mouvoir de façon non linéaire (épistasie) sur un chromosome après une opération de croisement. L'ensemble des chromosomes constituant une population représente le génotype et il peut agir sur son environnement (phénotype).

Les AG font appel à l'approche où les paramètres naturels du problème d'optimisation se codent comme une chaîne ayant une longueur finie. Les AG ont déjà un vaste champ d'application dans divers domaines, aussi variés que l'automatique, la chimie, la physique statistique, l'intelligence artificielle (IA), etc.. A l'heure actuelle, ces techniques sont en période de maturation et de mutation.

Un des problèmes rencontrés lors de l'utilisation des AGs de même que pour les méthodes du gradient ou du recuit simulé est la "convergence prématurée". Par ces termes, nous entendons la convergence vers une solution nettement non optimale [VACH 00].

6. L'IMPORTANCE DU CODAGE DES CHROMOSOMES

La première étape est de définir et de coder convenablement le problème. Historiquement, le codage utilisé par les AGs était représenté sous forme de chaîne binaire contenant toute l'information nécessaire à la description d'un individu (chromosome). D'autres formes de

age réel, le codage Gray, etc. Il existe deux types de
e. Dôune part, celui-ci doit pouvoir être adapté au problème de façon à limiter au mieux lôespace de recherche, et aussi de façon que les nouveaux chromosomes engendrés par les opérateurs de recherche soient significatifs le plus souvent possible, c'est-à-dire quôils puissent coder des solutions valides respectant les contraintes du problème.

7. LA FONCTION D'ADAPTATION (fitness)

Les variables codées sont reliées entre elles pour former une chaîne (chromosome), laquelle représente une solution possible (individu) du problème. La seule information nécessaire pour la procédure est la connaissance de la manière dont chaque individu se rapproche de son objectif, c'est-à-dire son fitness.

L'évaluation du fitness permet une classification des individus dans la population; le critère utilisé dépend du problème spécifique à résoudre. Le mécanisme qui est utilisé pour améliorer le fitness des individus constitue la plus importante partie de la procédure. Dans la phase de reproduction, les individus destinés à muter doivent être sélectionnés de telle sorte que la probabilité de sélection soit plus importante pour les individus ayant un fort fitness.

Dépendant essentiellement de la pression (probabilité) de sélection qui est désirée, différentes méthodes de sélection sont utilisées :

-la méthode de la roulette,

-la méthode de l'échantillonnage stochastique de la partie restante sans remplacement,

-et enfin la méthode du tournoi stochastique [BRIN 81].

Ces dernières seront abordées dans la suite du chapitre.

8. STRUCTURE GENERALE D'UN ALGORITHME GENETIQUE

Le fonctionnement des AGs se rapproche beaucoup de celui de la génétique dans la nature. On définit au départ une population d'individus sélectionnés aléatoirement et uniformément distribuée sur tout l'espace si cela est possible. Chaque individu de cette population est appelé *"Chromosome"*. Le chromosome est un ensemble de symboles, il est généralement, mais pas nécessairement, binaire. La population change à travers des itérations

…s. Durant ces générations, chaque individu est évalué …elée *"Fitness"*.

Pour créer une nouvelle génération, les meilleurs individus sont sélectionnés (ceux qui passent une épreuve de sélection choisie) et soumis aux opérateurs génétiques (croisement, mutation). Ce processus se poursuit, génération après génération, jusquà ce que le critère dàarrêt soit atteint, comme par exemple le nombre maximal de générations.

Algorithme :

1) Initialiser la population initiale P.
2) Evaluer P.
3) TantQue (Pas Convergence) faire :
    a) P ' = Sélection des Parents dans P
    b) P ' = Appliquer Opérateur de Croisement sur P '
    c) P ' = Appliquer Opérateur de Mutation sur P '
    d) P = Remplacer les Anciens de P par leurs Descendants de P '
    e) Evaluer P

Fin TantQue

9. PRINCIPE DE FONCTIONNEMENT

9.1 Le codage des données :

Làétape du codage est fondamentale dans les AGs, elle associe à chacun des points de làespace considéré une structure de données. Cette structure conditionne le succès des AG. Historiquement le codage binaire a été le premier à être utilisé [GOLD 89] car :

- Il peut facilement coder toutes sortes dàobjets : des réels, des entiers, des valeurs booléennes, des chaînes de caractèresé
- Il permet la création dàopérateurs de croisement et de mutation simples.

Cela nécessite simplement làusage de fonctions de codage et de décodage pour passer dàune représentation à làautre. Càest également en utilisant ce codage que les premiers résultats de convergence théorique ont été obtenus.

s comme le codage par permutation des entiers ou le
s le codage des réels et évite la phase coûteuse du décodage [LABE 07].

9.2 Génération de la population initiale

La population initiale conditionne fortement la rapidité et la convergence de l'algorithme génétique. On distingue deux cas :

a) Si la position de l'optimum dans l'espace d'état est totalement inconnue, il est essentiel que la population initiale soit répartie sur tout le domaine de recherche.

b) Si on a des informations à priori sur le problème, on génère les individus dans un sous domaine particulier (relativement aux informations disponibles) afin d'accélérer la convergence.

9.3 La fonction à optimiser

La fonction d'adaptation, ou ***fitness***, associe une valeur pour chaque individu. Cette valeur a pour but d'évaluer si un individu est mieux adapté qu'un autre à son environnement. Ce qui signifie qu'elle quantifie la réponse fournie au problème pour une solution potentielle donnée. Ainsi les individus peuvent être comparés entre eux. Cette fonction, propre au problème, est souvent simple à formuler lorsqu'il existe peu de paramètres. Au contraire, lorsqu'il y a beaucoup de paramètres ou lorsqu'ils sont corrélés, elle est plus difficile à définir.

9.4 Les opérateurs génétiques

Afin de diversifier les individus d'une génération à l'autre, il est nécessaire d'introduire divers opérateurs selon la complexité de l'application. On peut citer [GOLD 94] :

9.4.1 Sélection (Reproduction)

La sélection a pour objectif d'identifier les individus qui doivent se reproduire. Cet opérateur ne crée pas de nouveaux individus mais identifie les individus sur la base de leur

mieux adaptés sont sélectionnés alors que les moins

Notons que les étapes de sélection et de remplacement sont indépendantes de l'espace de recherche. On distingue deux types de sélection [Net 1]:

- **Sélection déterministe:** On sélectionne toujours les meilleurs individus et on écarte totalement les plus mauvais. Cela suppose un tri de l'ensemble de la population. On parle alors d'élitisme.
- **Sélection stochastique :** On favorise toujours les meilleurs individus mais de manière stochastique, ce qui laisse une chance aux individus moins performants. Il se peut que le meilleur individu ne soit pas sélectionné au profit du plus faible et qu'aucun des enfants ne soit au niveau du meilleur parent.

En général, la sélection doit favoriser les meilleurs éléments selon le critère à optimiser (minimiser ou maximiser). Ceci permet de donner aux individus dont la valeur est plus grande une probabilité plus élevée de contribuer à la génération suivante. Il existe plusieurs méthodes de sélection, les plus connues étant la « roulette » et la « sélection par tournoi » [SPAL 99].

a) Sélection par roue de loterie

Le principe de sélection le plus simple est celui de la sélection par roue de loterie (Roulette Wheel Sélection) conçu par J.Holland (1975). Il consiste à associer à chaque individu un segment dont la longueur est proportionnelle à sa fitness; ces segments sont ensuite concaténés sur un disque (roue de loterie).
La sélection d'un segment se déroulera par le tirage d'un nombre aléatoire de distributions uniformes entre 0 et 1.

A l'aide de ce système, les grands segments, c'est-à-dire les bons individus auront plus de chance d'être tirés au sort lors du déroulement du jeu et vont être reproduits et soumis à d'autres opérateurs génétiques. Parfois, on peut avoir un bon individu qui se reproduit trop souvent en provoquant l'élimination de ses congénères (Convergence prématurée) et convergera vers un optimum local (un bon individu qui prend le contrôle de la population) [GOLD 94].
La probabilité de sélection d'un individu au sein d'une population de taille *j* s'écrit :

Figure 3.1 La probabilité de sélection

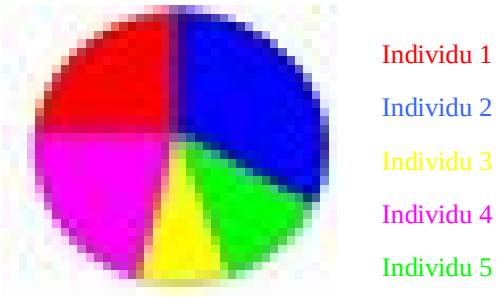

Figure 3.2 La sélection par roue de loterie

b) Sélection de Holland (Par restes stochastiques)

Chaque individu de la population est caractérisé par une note, représentant le rapport de son adaptation (sa fitness) à lôadaptation moyenne de la population, dont la partie entière représente le nombre de copies de lôindividu dans la population à créer. Si la population est incomplète (nombre dôindividus insuffisant), on sélectionne les meilleurs individus de la population après lôavoir triée.

c) Sélection par tournoi

Elle consiste à choisir aléatoirement un nombre dôindividus (participants au tournoi) et à reproduire le meilleur dôentre eux (celui qui a la plus grande adaptation).

Les individus sont de nouveau disponibles pour les tournois restants (il y a autant de tournois que dôindividus à remplacer).

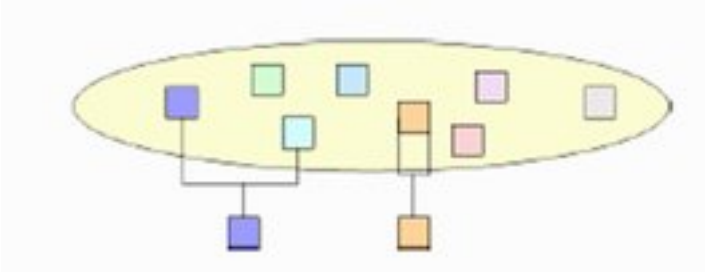

Figure 3.3 Le tournoi entre individus

Cette méthode de sélection permet de mettre en avant les meilleurs individus de la population. On trie l'ensemble de la population suivant leur adaptation et on sélectionne les premiers (les faibles n'ont aucune chance au contraire des forts qui sont toujours sélectionnés).

9.4.2 L'opérateur de croisement (Crossover)

Cet opérateur a pour but d'enrichir la diversité de la population et exploiter l'espace de recherche en manipulant la structure du chromosome. Les croisements sont envisagés avec deux parents et génèrent deux enfants. Les descendants doivent hériter quelques caractères de chaque parent. C'est donc un essai d'amélioration des meilleurs individus (parents) produits jusqu'au moment du croisement. Pour cela, plusieurs techniques sont utilisées selon le codage adopté.

Croisement en un point :

Le croisement en un point est le croisement le plus simple. On sélectionne aléatoirement un point de coupure *k* compris entre 1 et L-1 (L est la longueur du chromosome) puis on subdivise le génotype de chacun des parents en deux parties de part et d'autre part de ce point. On échange ensuite les deux sous chaînes terminales de chacun des deux chromosomes, ce qui produit deux enfants.

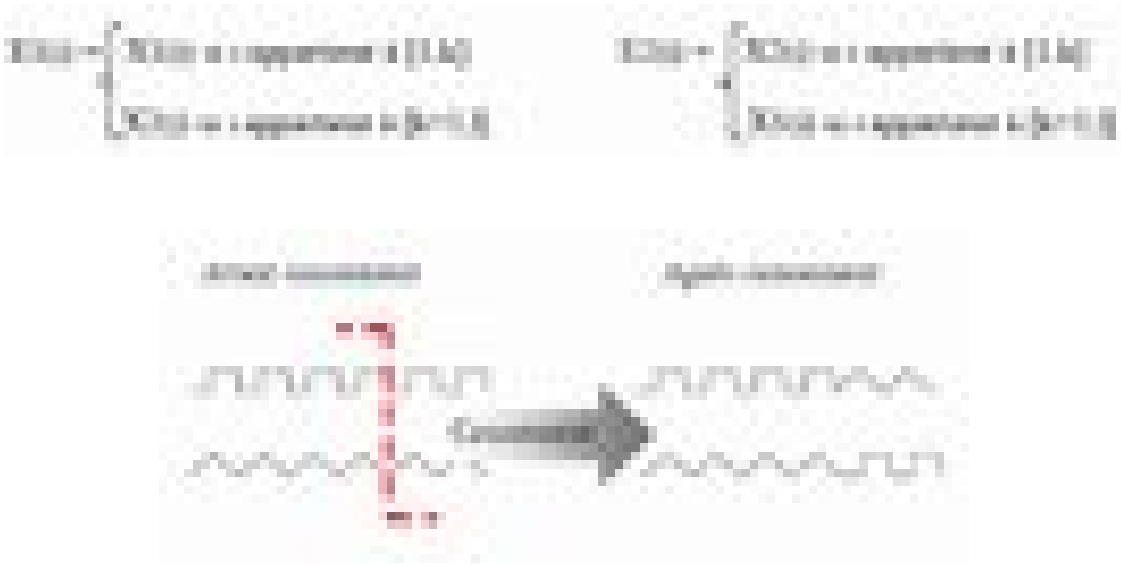

Figure 3.4 L'opérateur de croisement en un point de coupure

Ce type de croisement peut être considéré comme une généralisation du croisement en un point, en découpant le chromosome non pas en deux sous chaînes mais en *k* sous chaînes.

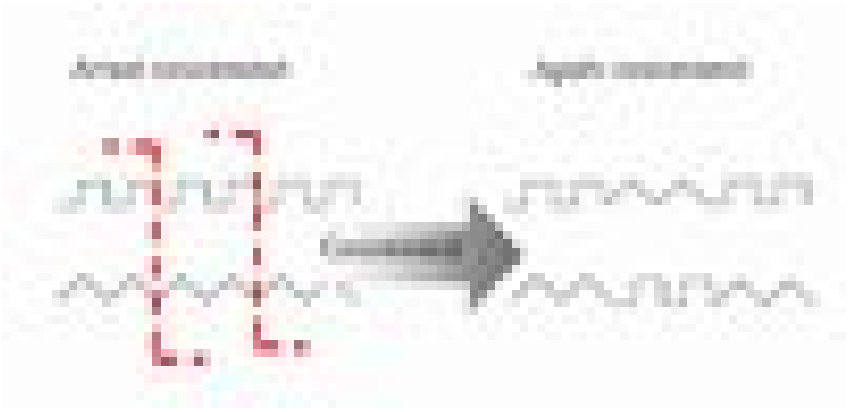

Figure 3.5 L'opérateur de croisement en deux points de coupure

Croisement uniforme :

Le croisement uniforme peut être vu comme un croisement multi points dont le nombre de coupures est indéterminé à priori. Pratiquement, on utilise « un masque de croisement », engendré aléatoirement pour chaque couple d'individus, qui est un mot binaire de même longueur que les chromosomes. Un « 0 » à la n ^{ième} position du masque laisse inchangé les symboles à la n ^{ième} position des deux génotypes, un «1 » déclenche un échange des symboles correspondants.

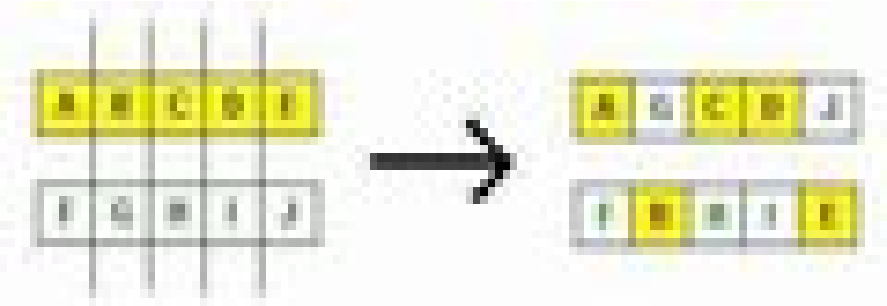

Figure 3.6 L'opérateur de croisement uniforme

Plus particulièrement, il existe trois autres types de croisements utilisés dans les problèmes d'ordonnancement. Nous les citons dans ce qui suit [VACH 00] :

crossover)

Le PMX fut proposé par Goldberg et Lingle [GOLD 85]. Le but était de construire un enfant par choix de sous séquences d'un ordonnancement d'un des parents et de préserver autant que possible l'ordre et la position des lots des autres parents. La sous- séquence de l'ordonnancement est sélectionnée par choix de deux points de coupure aléatoire, lesquels servent de frontière pour l'opération de remplacement.

Le crossover OX (order crossover)

Le crossover OX, proposé par Davis [DAVI 85], construit un enfant par choix de sous séquences d'un ordonnancement d'un des parents et préserve l'ordre relatif des lots des autres parents.

Le crossover CX (cycle crossover)

Le crossover CX, proposé par Oliver [OLIV 87], engendre un enfant dans tous les cas tel que chaque lot provient d'un des parents.

9.4.3 L'opérateur de mutation

Cet opérateur permet aux algorithmes génétiques d'explorer tout l'espace de recherche en modifiant aléatoirement une partie de la population. Il a été conçu pour renforcer les deux opérateurs: reproduction et croisement, qui par leurs caractères pseudo- aléatoires peuvent avoir le désavantage d'oublier des solutions. Sur le plan théorique, les propriétés de convergence des AGs sont fortement dépendantes de cet opérateur, et un algorithme peut même converger rien qu'en utilisant des mutations.

Comme pour le croisement, plusieurs techniques de mutation peuvent être utilisées. Pour les problèmes discrets, l'opérateur de mutation consiste généralement à tirer aléatoirement un gène dans le chromosome et à le remplacer par une valeur aléatoire. Si la notion de voisinage existe (utilisation de distance) dans le modèle retenu, il pourra être judicieux de choisir à chaque fois des valeurs mutées dans le voisinage des valeurs originales [GOLD 94].

un exemple d'application de lôopérateur de mutation sur
e binaire) [GOLD 94].

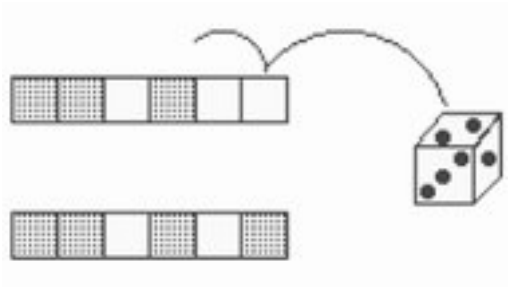

Figure 3.7 Lôopérateur de mutation

Remarques:

Les opérateurs de croisement et de mutation sont appelés opérateurs génétiques car ils simulent le processus d'héritage des gènes, pour créer une nouvelle génération. Par contre l'opérateur de reproduction (sélection) est un opérateur d'évolution qui simule le processus d'évolution darwinien, pour produire de nouvelles populations qui serviront à la création des générations.

10. LES PARAMETRES DE DIMENSIONNEMENT

Parmi les différents paramètres qui caractérisent les AGs, nous avons les paramètres suivants:

ÅLa taille de la population, cêst-à-dire le nombre dôindividus dans la population. Si la taille est trop petite, lôAG peut ne pas converger, par contre si elle est trop grande, lôévaluation des individus peut être très longue [LABE 07].

ÅLe critère dôarrêt de lôalgorithme: Pour un AG, ce critère n'est pas fixe, il peut changer d'une application à une autre. Il peut s'agir de:

- Nombre total de générations atteint.
- Limites sur utilisation des ressources CPU.
- Optimum atteint.

fonction fitness.

- Plusieurs générations sans améliorations.
- Combinaison entre deux ou plusieurs critères cités ci-dessus.

Å La probabilité dôapplication des opérateurs génétiques: détermine le choix du taux de mutation et de croisement. Le taux de croisement est généralement assez fort et se situe entre 70% et 95% de la population totale, par contre celui de la mutation est généralement faible, et se situe entre 0.5% et 1% de la population totale.

Ce paramétrage n'est pas universel, car il n'est pas adapté à la résolution de tous les problèmes, mais peut être pris comme point de départ pour démarrer une recherche de solutions à un problème donné.

11. DOMAINES D'APPLICATION:

Les algorithmes génétiques ont prouvé leur puissance dans plusieurs domaines, ils sont appliqués pour les problèmes d'optimisation n'ayant pas de méthodes de résolution décrites précisément, ou dont la solution dans le cas où elle est connue est très compliquée pour être calculée. Dans ce cadre, on peut citer quelques problèmes complexes [LABE 07]:

ÅLa constitution des équipes de travail.
ÅLôoptimisation dans les réseaux.
ÅLe problème des huit rênes.
ÅLa mise au point d'un emploi du temps.
ÅLe problème du voyageur de commerce.
ÅEtc.

Les AGs sont riches en terme d'applications, et largement utilisés. On peut les appliquer pour résoudre des problèmes divers: biologiques, informatiques; on les trouve aussi dans les sciences de l'ingénieur, dans la programmation des jeux et la reconnaissance des formes. Ils

ent des images médicales et dans divers problèmes

12. AVANTAGES ET INCONVENIENTS DES ALGORITHMES GENETIQUES

12.1 Les Avantages:

Å Parce qu'ils sont basés sur les techniques d'évolution naturelle, les AGs ne nécessitent pas beaucoup de connaissances mathématiques sur le problème à optimiser (n'ont pas besoin de dérivabilité, continuité, etc.).

ÅLargement applicables: on peut résoudre avec les AG plusieurs types de problèmes: linéaires ou non linéaires, discrets ou continus,é .

ÅCe sont des algorithmes de recherche globale, contrairement aux méthodes classiques d'optimisation qui n'effectuent ce type de recherche qu'en satisfaisant à des propriétés garantissant la globalité de chaque optimum.

ÅLes AGs offrent la flexibilité d'hybridation avec d'autres méthodes, ce qui permet dôapprocher beaucoup plus la solution exacte et d'avoir de meilleurs résultats.

12.2 Les Inconvénients :

ÅPas de garantie d'obtenir une solution en un temps fini.

ÅCoûteux en temps de calcul (codage, évaluation, sélection, recombinaison, mutation, évaluation, etc.).

ÅPas de paramétrage universel.

ÅLes opérateurs de base sont parfois insuffisants pour résoudre un problème donné, ce qui peut conduire à une mauvaise conclusion (Algorithme Génétique non applicable pour le problème), ou à plus de temps dans les essais avec d'autres opérateurs génétiques.

HMES GENETIQUES POUR LA RESOLUTION DES PROBLEMES DE PLANIFICATION ET D'ORDONNANCEMENT MULTI SITES

Dans le but dôoptimiser la fonction de planification et dôordonnancement de la production dans les entreprises multi sites, nous avons adopté les algorithmes génétiques. Ces derniers ont été utilisés pour pouvoir répondre en particulier au besoin de minimisation du temps de production par rapport à lôensemble des sites constituant lôentreprise.
Ceci se traduit par la fonction objectif de lôalgorithme génétique que nous avons mis en place, en dôautres termes, nous devons obtenir ceci :

$$F(t) = \min C_{max}$$

Dans ce but, une codification du problème a été faite. Ainsi, chaque chromosome représente les tâches à effectuer.

Les tâches sont codifiées à leur tour, car une tâche ne peut pas être représentée par un simple chiffre. Elle est alors représentée par: son numéro, le lot auquel elle appartient, sa priorité, le site, lôatelier, et la machine sur lesquels elle doit sôexécuter, ainsi que sa durée. Ce dernier paramètre nous conduit au calcul de la fonction objectif que nous nous sommes fixée. En effet, la durée de chaque tâche sera utilisée au niveau de chaque chromosome pour pouvoir lôévaluer et faire évoluer les populations constituant lôalgorithme génétique pour pouvoir déterminer la meilleure solution ou sôen rapprocher.

14. CONCLUSION

Dans ce chapitre, nous avons présenté le moyen que nous avons utilisé pour satisfaire la fonction dôordonnancement statique (dans la production multi sites), ce moyen étant les AGs, pour lesquels nous avons pu dégager les atouts et les points forts qui nous ont amenés à les utiliser dans notre approche.

Nous aborderons ci-après le thème des systèmes multi agents en passant par les notions élémentaires de lôIA, de lôagent, des interactions entre agents ainsi que les systèmes multi agents et les agents que nous avons développés pour lôaspect dynamique de notre approche.

CHAPITRE 4 :

LES SYSTEMES MULTI AGENTS POUR LA PLANIFICATION ET L'ORDONNANCEMENT DISTRIBUES

Au cours de ces dernières années, un effet de convergence est apparu vers une conception comportementaliste des objets informatiques. Cette tendance répond au concept d'agent dit autonome, doté de buts propres par opposition aux concepts objets. Cette orientation est traduite plus globalement par une nouvelle conception de la programmation plus modulaire, dite Programmation Orientée Agent (POA), dôoù *l'approche multi- agents* pour pallier les limites de la programmation Orienté Objets afin de faire face à de nouvelles exigences: complexité croissante des applications informatiques, plus ouvertes, plus hétérogènes et plus dynamiques.

Aussi, les agents logiciels représentent un paradigme de développement de logiciel approprié pour la résolution de problèmes distribués. Le concept dôagents autonomes et intelligents est ainsi souhaité dans le domaine de la planification et de lôordonnancement de la production [SAUE 00]. Lôautonomie concerne la capacité de décision dans la détermination du planning de production, des approvisionnements, des allocations de ressources et dôamélioration de la production [TARO 01].

Une étude intéressante a été citée dans [MULL 02] où lôauteur propose de faire le point sur la nécessaire apparition des systèmes multi agents. Le tableau suivant exprimant cette idée démontre bien le passage de la notion dôobjet vers la notion dôagent :

Paradigme\responsabilité	Comment	Quand	Pourquoi
Objet	X		
Processus	X	X	
Agent	X	X	X

Tableau 4.1 Lôobjet, le processus et lôagent

Un des apports fondamentaux de la programmation orientée- objet est lôencapsulation du « comment ». Le mécanisme proposé est de demander à un objet de rendre un service (le « quoi »). Comment ce service est réalisé est de la responsabilité de lôobjet à qui on le demande.

ce service est rendu sans remettre en cause celui qui a
tilisabilité des composants logiciels.

Néanmoins, si lôobjet a la responsabilité du « comment », il nôa aucun contrôle sur le « quand » le service est exécuté ni sur la possibilité de refuser de remplir le service demandé. Le processus, pensé comme un objet actif, ajoute la responsabilité du « quand » dans la mesure où il sôexécute en parallèle et donc décide souverainement quand il est opportun de remplir un service donné. La programmation dôun tel composant demande non seulement une analyse des services à rendre (« quoi ») et de la meilleure façon de les accomplir (« comment ») mais aussi de la temporalité des composants avec lesquels il est en interaction de façon à être sûr dôaccomplir les bonnes actions au bon moment.

Dans la méthodologie de conception, il sôagit donc dôune part dôanalyser à quoi sert ce composant (« pourquoi ») et dôautre part dôanalyser dans quel contexte il va servir afin de dériver une réalisation correcte. Si le contexte est dynamique et partiellement imprédictible, la programmation du composant en devient délicate. Le concept dôagent (et donc de programmation orientée- agent) permet de fournir le niveau dôabstraction nécessaire pour résoudre ce problème en proposant de rendre non seulement explicite le « pourquoi » du composant mais en plus en le codant explicitement dans le composant en même temps que les mécanismes qui permettront au composant de calculer dynamiquement quand et comment il est plus adéquat de rendre ses services. On obtient ainsi la notion dôagent autonome.
La figure suivante montre la différence entre un agent et un objet :

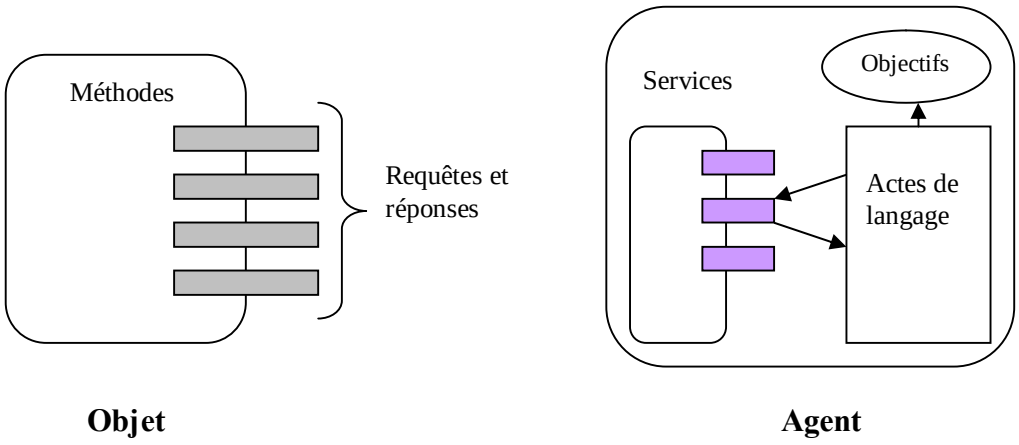

Figure 4.1 De lôobjet vers lôagent

tion de production nécessite en première analyse les

- Les commandes clients qui décrivent les produits à fabriquer, leur quantité, leur date de livraison, le client auquel ils sont destinés, etc. ;
- Les centres de fabrication qui prennent compte et intègrent les commandes clients sur lesquels ils doivent intervenir, quand ils doivent intervenir, etc;
- Les stocks qui contiennent les pièces en attente de manipulation, montage ou livraison.

Dans une perspective orientée- agent, on peut essayer d'assigner à chaque objet une finalité:

- Une commande de livraison a pour but de se faire fabriquer dans les délais ;
- Un centre de charge doit maximiser son taux d'utilisation, minimiser les changements de réglage, etc. ;
- Un stock doit assurer la présence d'un minimum de pièces (ou aucune en flux tendu).

Ainsi, chaque objet devient actif et va interagir avec les autres pour satisfaire son objectif. Ce système se stabilise lorsque tous les buts sont satisfaits et se remet en activité dès qu'un but cesse de l'être suite à une perturbation (indisponibilité d'un centre de charge, retard de livraison, etc.). De ce fait, on obtient un système dynamique qui va réagir spontanément à toute perturbation en essayant de rétablir par exemple: son délai de fabrication, son taux d'utilisation et la disponibilité des pièces nécessaires. Finalement, on souhaite que le système dans son ensemble respecte certaines propriétés, par exemple d'optimalité (minimisation des retards moyens, etc.) [MULL 02].

2. LA NOTION D'AGENT

2.1 L'intelligence artificielle distribuée:

L'un des objectifs de l'Intelligence Artificielle (IA) est la définition de systèmes capables de représenter des connaissances, de raisonner, de planifier des actions afin de résoudre des problèmes pouvant être très complexes. Elle cherche à obtenir des résultats comparables à ceux que feraient des êtres humains dans des cas similaires, mais sans forcément utiliser les mêmes moyens.

mples, il est possible de considérer un système dit vironnement. Cependant, en dehors de ces problèmes simplifiés, on se retrouve souvent dans les cas réels avec un système qui n'est plus seul à agir sur l'environnement. Les effets de ses actions peuvent alors se trouver combinées à celles d'autres entités avec une certaine influence. Alors, il faut étendre cette notion de système intelligent unique et construire un système à base d'entités plus simples où chacune s'occupe d'une partie du problème, tout en ayant au final une complexité globale moindre [AISS 06].

Nous pouvons ainsi dire que, contrairement à l'approche centralisée de l'IA, l'intelligence artificielle distribuée (IAD) vise la répartition de l'expertise sur un ensemble de composants qui communiquent pour atteindre un objectif global [VACH 00]. Cette approche nécessite la division du problème en sous- problèmes.

C'est donc avec ces motivations que sont apparues au début des années 1970, les approches visant à distribuer l'intelligence artificielle avec le premier système à tableau noir HEARSAY. Ensuie, trois axes sont apparus parmi lesquels on retrouve lôaxe concernant les ***Systèmes Multi- Agents*** (SMA) [FERB 95] qui considère que de petites entités interagissant fortement peuvent produire un tout intelligent amenant finalement à une vue sociale de l'IA. C'est alors qu'est apparu le terme agent, afin de qualifier les entités considérées dans cette approche.
Depuis, le terme agent a connu de multiples définitions et les travaux utilisant une approche dite agent pour résoudre des problèmes divers sont nombreux.

2.2 Définition de l'agent :

Le concept d'agent est utilisé dans divers domaines comme l'ingénierie, la gestion des réseaux, les systèmes distribués, la robotique, les interfaces Homme/Machine et l'Intelligence Artificielle [VACH 00]. On trouve alors plusieurs définitions différentes d'un agent variant selon le domaine dôapplication et le système utilisé.
Nous présenterons deux définitions parmi les plus connues :

Définition 1 : de Ferber [FERB 95] :
"On appelle agent une entité réelle ou abstraite qui est capable d'agir sur elle-même et son environnement, qui dispose d'une représentation partielle de cet environnement, qui, dans un

er avec d'autres agents, et dont le comportement est la
sa connaissance et des interactions avec les autres agents".

Définition 2 : de M. Wooldridge et N. Jennings [WOOL 95]:

Le terme "agent" caractérise un système informatique matériel ou (plus souvent) logiciel qui comporte les caractéristiques suivantes :

- ***Autonomie* :** *l'agent agit sans l'intervention d'humains ou d'autres intervenants, et a un certain contrôle sur ses actions et ses états internes.*
- ***Habilité sociale* :** *l'agent interagit avec d'autres agents (pouvant être des êtres humains) à l'aide d'un langage de communication d'agents.*
- ***Réactivité* :** *l'agent perçoit son environnement (qui peut être un monde physique, un utilisateur via une interface graphique, un ensemble d'autres agents, Internet, ou encore tous ces éléments combinés), et répond de manière opportuniste aux changements qui y surviennent.*
- ***Pro- activité* :** *l'agent n'agit pas simplement aux stimuli de son environnement, il est aussi capable de démontrer des comportements dirigés par des buts en prenant des initiatives.*

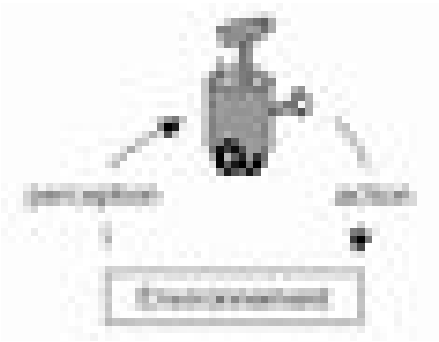

Figure 4.2 L'agent

D'un autre côté, nous pouvons dire qu'un agent est un objet logiciel autonome, capable de communiquer. Un système d'agents est conçu pour atteindre les objectifs programmés d'avance par son concepteur [TARO 01].

Un grand nombre d'architectures d'agents sont organisées de manière modulaire. Ces modules se retrouvent sous des formes proches dans la plupart des architectures d'agents. Typiquement, toutes les architectures d'agents comportent les trois volets : perception, délibération et action, comme le montre la figure 4.3 :

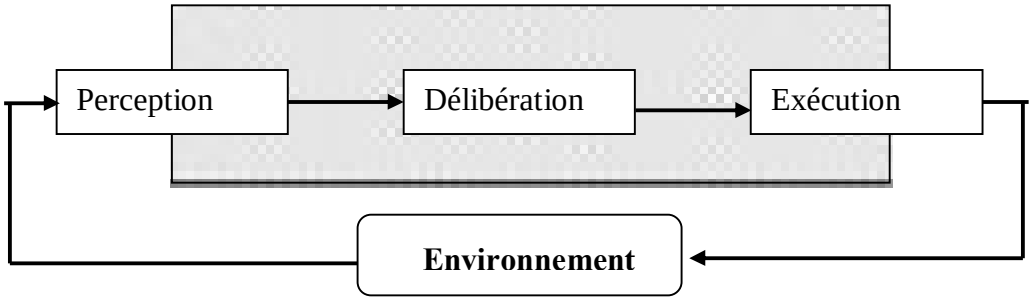

Figure 4.3 Structure générale d'un agent

Notons que l'organisation, la communication et l'interdépendance entre modules sont différentes d'une architecture à une autre.

2.4 Typologie des agents :

2.4.1 Les agents réactifs

Un agent est réactif s'il répond de manière opportune aux changements de son environnement. Ceux-ci peuvent être constitués de stimuli externes et asynchrones. La structure des agents purement réactifs tend à la simplicité, mais ces derniers peuvent être capables d'actions de groupe complexes et coordonnées.

C'est le cas d'une société de fourmis, dont la somme des membres est capable d'actions évoluées mais dont chaque individu pris séparément possède une représentation faible de l'environnement et n'a pas de buts globaux [AISS 06].

itifs

Un agent est intentionnel, dirigé par des buts ou encore cognitif, s'il ne se contente pas d'agir en fonction des conditions de son environnement mais en fonction de ses buts propres et s'il possède une intentionnalité (une volonté consciente d'effectuer un acte). Un tel agent doit posséder un état mental qui l'incite à agir dans un sens particulier et donc une représentation symbolique, de plus ou moins haut niveau en fonction de la complexité de la tâche à accomplir, de son environnement et de ses buts.

Cette représentation symbolique lui permet de prendre conscience de l'état d'accomplissement de ses buts et des contraintes restant à soulever, mais aussi de ses motivations propres.

Un agent intentionnel est opposé à un agent purement réactif, car la perception de son environnement est symbolique et son comportement est conditionné par ses motivations. Notons qu'un agent intentionnel peut réagir à un stimuli de son environnement tout en conservant une haute représentation de celui-ci et en agissant en fonction de ses intentions. Les agents cognitifs ont des architectures complexes (ex : BDI) [AISS 06].

Notons que cette notion de réactif / cognitif ne constitue qu'une classification de travail. Elle doit pouvoir être dépassée par une synthèse des deux termes chaque fois que cela s'avère nécessaire. En effet, réactif et cognitif sont deux extrêmes entre lesquels se situent des types hybrides d'agents [FERB 95].

2.4.3 Les agents autonomes

Un agent autonome est un agent qui agit dans et sur son environnement et qui ne se contente pas d'être dirigé par des commandes venant d'un autre agent ou de l'opérateur. Un tel agent doit posséder des effecteurs et être situé dans son environnement.

L'autonomie d'un agent ne porte pas seulement sur son comportement mais aussi sur ses ressources internes qui sont elles aussi autonomes. Un agent autonome doit savoir comment accomplir ses buts en fonction de l'état de son environnement ou tout au moins, doit être capable de trouver un moyen pour accomplir ses buts.

Un agent adaptatif est capable d'apprendre en fonction de son expérience passée et de son évolution. Les techniques d'apprentissage classique de l'IA s'appliquent particulièrement bien à l'apprentissage d'un agent (qu'il s'agisse de faire évoluer ses capacités d'action, de décision ou d'analyse).

2.4.5 Les agents mobiles

Un agent mobile est un agent capable de se déplacer dans son environnement, qui peut être physique (réel ou simulé) ou structurel (niveaux d'exécution par exemple).

Un agent mobile dispose donc de dispositifs assurant sa mobilité. Dans le cas d'un agent logiciel, la mobilité implique le déplacement dans le réseau ou dans l'architecture du système, c'est à dire la mise en place d'un mécanisme de migration de processus pour tirer profit de la topologie du réseau informatique.

A titre d'exemple, les agents d'information sont la plupart du temps des agents mobiles, leur rôle est de parcourir des sources d'information souvent distribuées et donc d'être amenés à se déplacer dans un réseau. L'agent mobile assure alors un rôle de passerelle et est pourvu d'une interface avec ces sources d'information.

2.4.6 Les agents flexibles

Les agents flexibles sont des agents capables de modifier leur structure fonctionnelle (code) en cours d'utilisation. Par exemple, les agents flexibles peuvent être capables de charger dynamiquement un script de contrôle envoyé par l'utilisateur.

L'exécution de tels scripts nécessite une interface de chargement (typiquement, un service réseau) ainsi qu'un mécanisme d'exécution des scripts reçus: interface avec les données, opérateurs d'exécution, etc.

2.4.7 Les agents sociaux

Un agent social est un agent qui opère des interactions avec d'autres agents au sein de son environnement. Ces interactions peuvent faciliter la tâche d'un agent (coopération) ou au contraire le gêner dans l'accomplissement de ses buts (encombrement, compétition).

Les agents avec lesquels un agent social est en mesure d'avoir des interactions peuvent être hétérogènes, c'est-à-dire qu'ils peuvent avoir des formes, des interfaces, des buts ou des représentations de l'environnement différentes.

3.1 Définition :

Les systèmes multi- agents constituent une nouvelle technique de modélisation qui place lôobjet dôétude au centre de sa démarche. Ces modèles représentent les actions individuelles, les interactions entre les acteurs et les conséquences de ces interactions sur la dynamique du système.

Les systèmes multi- agents empruntent à lôIAD les modes de communication et de concertation entre agents. Ils reprennent les idées dôautonomie et dôémergence du résultat final à partir des interactions individuelles, à la vie artificielle.

La figure 4.4 montre les sources des SMA :

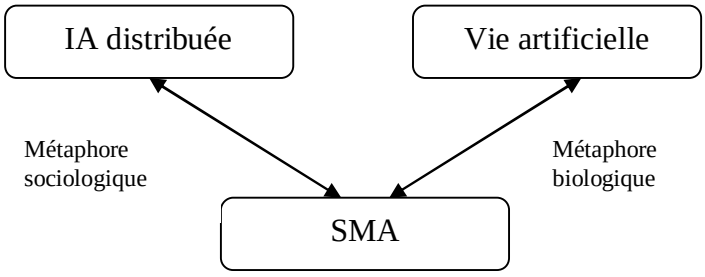

Figure 4.4 Sources des SMA

Ferber [FERB 95] donne la définition suivante :

« Un système multi- agents est un système composé des éléments suivants:

- *un environnement, c'est à dire un espace disposant généralement d'une métrique,*
- *un ensemble d'objets situés dans l'espace, passifs, ils peuvent être perçus, détruits, créés et modifiés par les agents,*
- *un ensemble d'agents qui sont les entités actives du système,*
- *un ensemble de relations qui unissent les objets entre eux,*
- *un ensemble d'opérations permettant aux agents de percevoir, de détruire, de créer, de transformer et de manipuler les objets*
- *un ensemble d'opérateurs chargés de représenter l'application de ces opérations et la réaction du monde à cette tentative de modification (les lois de l'univers)* **».**

stème multi agents :

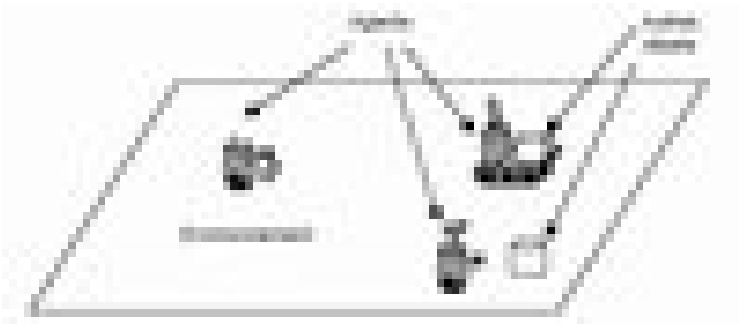

Figure 4.5 Un exemple de système multi- agents

Ainsi, un système multi- agents est un ensemble dentités autonomes et actives (les agents). Les comportements sont distribués au niveau individuel des agents. Chacun est alors spécialisé et agit de façon autonome.

De ces actions individuelles, émerge la solution ou le comportement général du système, soit par une interaction due à la modification de lenvironnement par les agents où ils évoluent, soit par une communication directe entre agents par le biais dun langage symbolique, par exemple.

A partir de cette définition, nous déduisons que linteraction et la communication sont un phénomène important dans les SMAs. Ce qui nous amène à présenter et décrire les phénomènes d'interaction et la notion d'organisation qui rendent possible la résolution collective de tâches dans les systèmes multi- agents.

3.2 Interactions et coopération entre agents

Un système multi- agents (SMA) se distingue dune collection dagents indépendants par le fait que les agents interagissent en vue de réaliser conjointement une tâche ou datteindre conjointement un but particulier [AISS 06]. Les agents peuvent interagir en communiquant directement entre eux ou par lintermédiaire dun autre agent ou même en agissant sur leur environnement.

Le fait d'avoir plusieurs agents actifs au même moment dans le système implique de nouveaux phénomènes :

e agent et les lois d'évolution de l'environnement se
lifier, s'annuler, se perturber...

2) Les agents peuvent percevoir les effets provoqués par les actions dôautres agents.

Ainsi, un agent peut avoir connaissance du fait qu'il n'est pas seul dans son environnement. Ces deux remarques amènent à la notion d'interaction qui peut être définie de la façon suivante [FERB 95]: *« Une **interaction** est une mise en relation dynamique de deux ou plusieurs agents par le biais d'actions réciproques.* »

Cette interaction peut être coopérative ou non. Nous pouvons caractériser un système par le type de coopération mis en î uvre qui peut aller de la coopération totale à lôantagonisme total.

Aussi, des agents coopératifs peuvent changer leurs buts pour répondre aux besoins des autres agents afin dôassurer une meilleure coordination entre eux. Cela peut entraîner des coûts de communication élevés.

Les agents antagonistes par contre, ne vont pas coopérer et dans ce cas, leurs buts respectifs vont se trouver bloqués. Dans de tels systèmes, les coûts de communication sont minimaux.

Les interactions des agents dôun SMA sont motivées par lôinterdépendance des agents selon les trois dimensions suivantes:

- leurs buts peuvent être compatibles ou non;
- les agents peuvent désirer des ressources que les autres possèdent;
- un agent peut disposer dôune capacité nécessaire à un autre agent pour lôaccomplissement dôun des plans dôaction de ce dernier.

La façon dont se réalisent les interactions permet d'obtenir diverses situations au niveau du SMA. La figure 4.6 est une représentation dôun agent en interaction avec son environnement et les autres agents [VACH 00]:

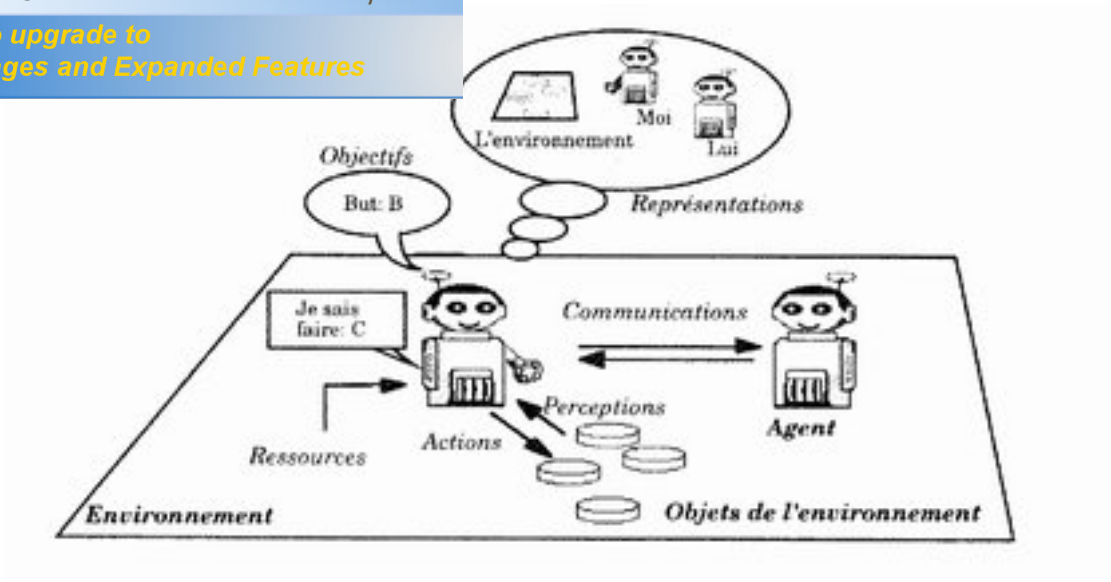

Figure 4.6 Représentation imagée dûn agent en interaction avec son environnement et les autres agents.

De façon concrète, l'ensemble de ces situations est géré au sein d'un SMA par des mécanismes de ***coordination***. Nous retrouvons la définition suivante dans la littérature :

« *La **coordination** est le processus de construction de programmes en assemblant les parties actives. Un modèle de coordination est la colle qui lie des activités séparées en un ensemble, c'est aussi la gestion des dépendances entre les activités.* »

La coordination est fortement liée aux relations qui peuvent s'établir entre les agents et plus généralement à la façon dont les agents sont organisés.

Pour établir plus clairement ce qu'est une organisation, nous proposons la définition utilisée par [FERB 95] pour les SMA:

« *Une **organisation** peut être définie comme un agencement de relations entre composants ou individus qui produit une unité, ou système, dotée de qualités inconnues au niveau des composants ou individus. L'organisation lie de façon interrelationnelle des éléments ou évènements 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.* »

de ***structure organisationnelle*** pour désigner la partie anisation donnée (dite ***organisation concrète***) devient alors une instance dôune structure organisationnelle. Nous citons par exemple le modèle organisationnel Aalaadin que nous présentons en annexe B.

4. COMMUNICATION ENTRE AGENTS

Lôidée dôinteraction est parfois liée à celle de communication. La communication la plus évoluée nécessite un langage commun manipulant des symboles. Toutefois, lôaction sur lôenvironnement et la modification de celui-ci par un agent influence les actions des autres agents. Dans ce cas, il y a interaction et non- communication.

Un système peut évoluer vers la résolution dôun problème sans quôil y ait communication directe entre les agents. Lôexemple du « tableau noir » commun peut être considéré comme un mode de communication si lôagent le modifie intentionnellement pour passer lôinformation aux autres agents. Sôil nôy a pas dôintention, il ne sôagit pas alors de « communication ».

Cependant, la transformation des données du tableau et de lôenvironnement peut constituer un signal dôaction pour certains agents. Lôaction de ces agents dépend alors de lôaction dôagents voisins; cette interaction ne peut pas alors être assimilée à une communication.

Traiter le problème de lôinteraction veut dire se donner les moyens de décrire les mécanismes élémentaires permettant aux agents dôinteragir dôun côté, et dôanalyser et de concevoir les différentes formes dôinteraction que des agents peuvent pratiquer pour accomplir leurs tâches et satisfaire leurs buts dôun autre côté.

En premier, les agents doivent être capables, par le biais de la communication, de transmettre des informations, mais surtout dôinduire chez lôautre un comportement spécifique. Communiquer est donc une forme dôaction particulière qui, au lieu de sôappliquer à la transformation de lôenvironnement, tend à une modification de lôétat mental du destinataire.

œexécuter une tâche, tend à provoquer chez lœautre une
stitue donc une manière de satisfaire un objectif sans réaliser la tâche soi- même.

Il existe deux formes de communication :

- Les communications intentionnelles mettent en contact des agents cognitifs par le biais dœenvois de message. Cette forme de communication est surtout utilisée par des agents organisés en réseaux,
- Les communications réactives qui prennent la forme de signaux transmis entre agents réactifs. On rencontre ces formes de communication surtout dans la robotique collective mobile et la simulation de sociétés animales mais beaucoup moins dans les réseaux.

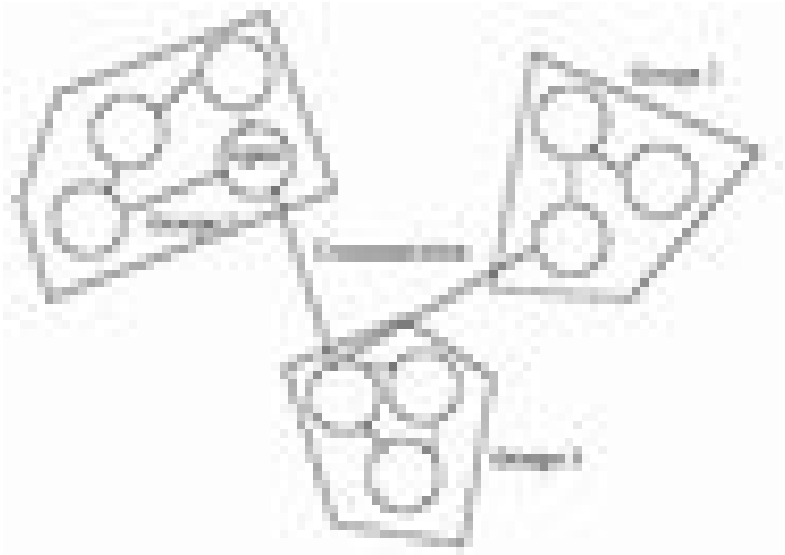

Figure 4.7 Exemple de communication entre des groupes dœagents

En ce qui concerne notre travail, nous avons utilisé le modèle Aalaadin cité plus haut (voir annexe B) présenté dans [FERB 98] et [FERB 02], qui nous a aussi servi comme source d'inspiration pour la définition de notre SMA [AITS 08 a] [AITS 08 b] et qui nous a poussé ainsi par la suite à utiliser la plate-forme MADKIT [FERB 02].

5. LES SYSTEMES MULTI AGENTS ET LE TEMPS REEL

Un agent est dit temps- réel sœil doit atteindre ses objectifs avec, en plus, des contraintes de temps qui sont susceptibles d'altérer la qualité de ses résultats. Un système temps- réel doit

agents temps réel de communiquer, de coopérer et de leur objectif et les contraintes temps réel (qualité de service) spécifiées.

Un tel système a l'avantage de présenter une manière particulière d'aborder des problèmes compliqués qui sont difficiles à résoudre de façon classique en temps réel tel que la gestion et la planification des ressources. Cependant, le développement de ces systèmes nécessite une véritable méthode devant être utilisée afin de prendre en compte les critères de qualité de service des systèmes temps réel; nous pouvons citer par exemple le langage UML (les diagrammes de séquences et les diagrammes d'état/transition).

En général, nous pouvons dire que les agents sont surtout utilisés dans des systèmes temps réel dits " soft ", où le non respect des délais ne met pas en péril toute l'application.

6. DOMAINES D'APPLICATIONS DES SYSTEMES MULTI AGENTS

Les systèmes multi agents étant issus du domaine de l'IAD, ils permettent de modéliser des applications où la modélisation classique s'avère inadéquate et où le système est naturellement distribué. Si un problème n'est pas naturellement distribué, on peut choisir de le distribuer pour des raisons de simplification.
Les applications qui sont naturellement distribuées sont par exemple celles concernant la simulation d'écosystèmes où un agent est associé à chaque individu [DUVA 01].

Nous pouvons citer aussi les systèmes complexes qui sont des systèmes où les techniques de modélisation classiques sont difficilement utilisables. En effet, dans ces systèmes, les paramètres sont nombreux ou contradictoires pour pouvoir être pris en compte, d'où la nécessité de l'autonomie et de l'adaptation du système à des facteurs incertains ou imprévisibles. Nous citons par exemple les outils implémentant la distribution et la mobilité.

Nous avons aussi les systèmes d'aide à la décision, qui sont présents dans de nombreux domaines et ont pour objectif d'aider le décideur dans sa tâche en lui fournissant tous les éléments pertinents pour la prise de décision. Cela consiste généralement à extraire de l'information depuis de multiples sources et à la traiter. Les SMA apparaissent comme étant bien adaptés pour traiter de l'information qui peut revêtir diverses formes et provenir de

x de [TAGH 08] qui a utilisé les SMA pour l'aide à la de la production dynamique.

N'oublions pas le domaine du commerce électronique et les agents du WEB, car le domaine du commerce électronique est un domaine en plein essor qui permet de favoriser les transactions commerciales. Ce domaine utilise l'outil informatique et plus particulièrement les ressources mises à disposition par Internet pour rapprocher les acteurs commerciaux dans certains domaines. L'utilisation des SMA est une bonne solution ici aussi car elle permet, comme pour le domaine d'aide à la décision, de faire de la fusion d'informations. La dénomination « agent du WEB » désigne les agents qui circulent sur le réseau Internet pour extraire de l'information.

D'un autre côté, les recherches en SMA sont très actives et des systèmes opérationnels ont déjà été développés dans de nombreux domaines comme le diagnostic, l'enseignement, la conception, le contrôle de réseaux de communication, etc. [BURG 94], [RAMO 94], [ROY 97], [BALA 97], [SHEN 98], [GOUA 99], [COUD 99], [ARCH 99], [TRAN 99], etc. Ces recherches concernent l'utilisation de l'approche multi- agents pour structurer des architectures pour le contrôle des processus [HAFR 01].

Plus particulièrement, les SMA sont utilisés dans le domaine industriel. Nous pouvons citer quelques unes de leurs applications [MOYA 00]:

- *La conception et l'ingénierie simultanée* : la production de meilleurs produits en un temps court. Dans ce cas, toutes les étapes du cycle de vie du produit (définition du cahier des charges, conception, élaboration des méthodes, mises en fabricationé) sont considérées dés que possible.

- *Le système d'information et le CIM (Computer Integrated Manufacturing)* : il est de plus en plus prouvé que la productivité des entreprises est plus limitée par l'information que par le travail ou le capital. Un exemple de gêne à la circulation de l'information provient des logiciels des différentes parties de l'entreprise qui ne peuvent pas "discuter" entre eux. Dans l'approche SMA, chacune de ces parties peut être vue comme un agent.

uction (multi- sites) : chaque site de production est décisions des autres sites.

- *La gestion de la production* : le problème consiste à faire cohabiter des objectifs qui ont des termes plus ou moins longs.

- *L'ordonnancement et le pilotage d'un système de production* : comment ré- ordonnancer dynamiquement un atelier qui subit des événements perturbateurs. Ce ré- ordonnancement est une résolution coopérative et distribuée des problèmes entre les différentes entités de l'entreprise. La résolution part de la plus petite entité (le poste de travail), et tente de résoudre le problème en mettant en jeu progressivement de plus en plus d'entités (machines identiques, puis atelier, etcé).

- *les chaînes logistiques* : ce sont des réseaux de fournisseurs, d'usines, d'entrepôts, de centres de distribution et de détaillants à travers lesquels des matières premières sont acquises, transformées et livrées au consommateur. L'objectif est d'optimiser les performances de ces chaînes logistiques. Pour cela, les niveaux de décision stratégique, tactique et opérationnelle sont mis en jeu pour que chaque maillon opère de manière intégrée par rapport à l'ensemble de la chaîne.

Nous examinons ci- dessous l'apport des SMA dans le domaine de la production.

7. LES SMA ET LES SYSTEMES DE PRODUCTION

Deux remarques peuvent être faites au sujet des SMA dans leur rapport avec l'entreprise [NEUB 97]:

- L'entreprise est un système complexe qui dispose d'une organisation, c'est-à-dire d'une structure qui assure la répartition et la coordination des tâches dans le but d'atteindre certains objectifs globaux. L'entreprise est un système naturellement distribué, son savoir faire, ses compétences, ses connaissances ne sont pas centralisées ni détenus par une entité unique. Chaque individu ou groupe d'individus dans l'entreprise possède une partie de cette connaissance nécessaire à la réalisation de ses activités, et d'autre part, la réalisation

xes dans lôentreprise repose sur lôinteraction de ces
ange dôinformation, é).

- Lôamélioration des performances du système global repose tant sur les capacités internes des individus à apprendre et à améliorer leurs connaissances, que sur les capacités dôinteraction [CAMP 94].

On voit donc apparaître deux axes de recherche pour lôétude des systèmes de production en utilisant les modèles agents [GLEI 94]:

- La recherche liée au domaine : cet axe concerne la modélisation des agents du système de production, leur identification et la définition de modèles génériques de comportement. [VERN 97] distingue par exemple dans la modélisation des entreprises, trois classes dôagents: les machines, les applications et les hommes.

- La recherche liée à lôorganisation : cet axe concerne les processus dôinteraction, coordination, coopération, collaboration, compétition, direction, é nécessaires à lôamélioration du système global. Il conditionne en grande partie la cohérence globale du système. Côest ce à quoi nous nous intéressons plus précisément dans notre travail.

Aussi, le fait que des SMA permettent de distribuer les problèmes sur des ensembles auto organisés vers une gestion locale favorisant réactivité et comportements émergents, peut faciliter la mise en oeuvre du pilotage en temps réel. En effet, le but principal dôun SMA est de faire collaborer un certain nombre dôagents afin de résoudre un problème. Un problème est défini par la donnée dôun état initial connu et dôun état final souhaité; la résolution du problème consiste alors à définir les étapes intermédiaires pour passer ainsi de lôétat initial à lôétat final [HAFR 01].

Nôoublions pas que le principe des SMA est de fédérer lôensemble des connaissances et la capacité de raisonnement détenu par les agents, que ceux-ci soient intelligents ou non. Notons que chacun des agents peut être spécialisé dans un sous- domaine du domaine global et cêst lôunification de leurs compétences qui permet de résoudre la totalité du problème posé [HAFR 01].

E

Lôapprentissage constitue un des éléments importants des systèmes multi agents. Cependant, il faut bien faire la distinction entre les deux formes dôapprentissage que lôon peut trouver au sein de ces systèmes. Lôune se situe directement à lôéchelle de lôagent, et lôautre au niveau supérieur, celui des systèmes multi agents [NEUB 97].

Il y a ainsi un besoin de techniques et de méthodes d'apprentissage en SMA à cause de la complexité de leur conception.

Le passage du cadre de l'agent à celui de la société d'agents enrichit la problématique de l'apprentissage. Les agents doivent apprendre la façon dôinteragir pour résoudre un problème. Des questions se posent alors: Quand et avec Qui communiquer ? Comment se coordonner ? é [AISS 06].

Parmi les méthodes dôapprentissage, nous citons les algorithmes génétiques: procédures robustes d'exploration d'espaces complexes [TEMP 01]. En effet, les algorithmes génétiques offrent une simplicité de calcul, une performance de recherche d'amélioration et ne sont pas limités par des hypothèses contraignantes sur le domaine d'exploitation. Ceci constitue les raisons pour lesquelles nous avons opté pour leur utilisation dans la résolution de notre problème.

Dans notre présent travail, lôidée est de faire un apprentissage au niveau de lôagent superviseur pour la prise de décision en parallèle à la réception des flux dôinformations des autres agents: le niveau de stock des sites, les priorités des opérations, les capacités des ressources, les lots de production, ... Nous avons ainsi recours à lôutilisation des SMAs et des AGs.

Notre problématique se présente en termes de planification et dôordonnancement multi sites. Ainsi nous avons adopté la notion dôagent et dôapprentissage (figure 4.10):

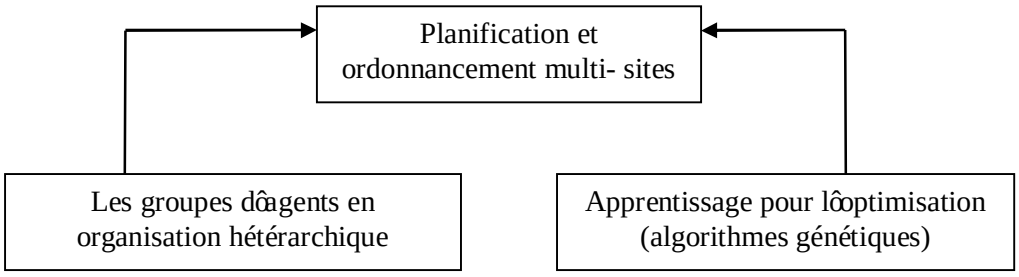

Figure 4.8 Planification et ordonnancement multi sites selon notre approche

ente notre modèle du système de production incluant un SMA, utilisant un AG à travers une architecture hétérarchique:

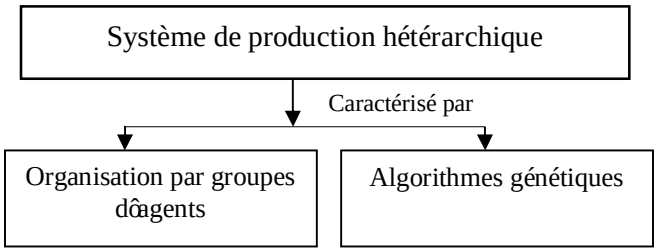

Figure 4.9 Notre modèle de résolution et dâoptimisation

9. PLATES-FORMES DE DEVELOPEMENT DE SYSTEMES MULTI AGENTS

Il existe des outils ou plates-formes de développement rapide pour la conception de systèmes reposant sur lâutilisation du paradigme agent. Après avoir examiné plusieurs de ces plates-formes, très peu apparaissent comme pouvant servir réellement à la conception de tout type dâapplication.

En effet, la plupart de ces plates-formes ne sâadaptent quâà la conception dâun seul type dâapplications. Il apparaît, de plus, que la définition dâun agent est beaucoup trop vague pour quâun modèle universel puisse exister.

Parmi les plates-formes les plus connues, nous pouvons citer SWARM, JADE, voyager, DIMA, Zeus, Madkit, etc.

SWARM est une plate-forme de développement qui possède une forte orientation vers le développement dâapplications de simulation à évènements discrets de systèmes complexes.

JADE (Java Agent DEvlopment Framework) est une plate-forme où les agents sont des coquilles auxquelles il faut ajouter des comportements implémentant des services/ fonctionnalités; elle offre une possibilité de communication entre plusieurs plates-formes JADE. Les communications utilisent le standard ACL.

Voyager est une plate-forme de développement commerciale. Elle fonctionne en environnement distribué et repose sur le standard CORBA. Son domaine dâapplication est orienté vers le commerce électronique et les applications WEB.

ion de systèmes multi agents) est une plate-forme dans laquelle l'architecture des agents choisie pour le développement de celle-ci est une architecture modulaire (module de supervision, module de communication, module de perception, module de raisonnement).

Zeus est une plate-forme de développement de SMA générique, personnalisable et peut être augmentée par l'adjonction de nouveaux composants. De par sa généricité, elle peut être utilisée dans la conception d'un grand nombre d'applications mais en retour nécessitera des augmentations assez nombreuses et spécifiques à chaque type d'application.

Madkit est une plate-forme de développement et d'exécution de SMA fondés sur des critères organisationnels.

Pour notre travail, nous avons opté pour l'utilisation de cette plate-forme (www.Madkit.org) qui se base sur le modèle organisationnel Aalaadin (voir annexe B).

Le choix de Madkit a été dicté par ses propriétés de flexibilité, qui permet son enrichissement dans le cadre de nos travaux; de légèreté, car elle n'impose pas de fonction superflue ce qui facilite son utilisation pour le développement de SMA, et son langage de développent étant Java qui dispose de tous les outils nécessaires pour le développement de SMA, en plus du fait qu'elle est particulièrement adaptée à des applications distribuées.

Cette plate-forme a été utilisée par de nombreuses équipes de recherche, notamment pour des simulations d'architectures hybrides de pilotage de robots sous- marins, pour l'évaluation de réseaux sociaux ou bien encore pour l'étude de pilotage par SMA de production on-line.

Nous citons par exemple, Wex, développé par Euriware S.A., qui est une application Madkit assez complexe pour les applications de gestion de connaissances. Cet outil engendre de l'information à partir de sources de données (bases de données, outils de support, moteurs de recherche sur le web, pages courantes explorées par des utilisateurs et analysées, é) et présente des vues unifiées de ces sources de connaissance hautement hétérogènes. Les utilisateurs peuvent maintenir des ontologies partagées dans leur domaine. Les agents sont implémentés pour encapsuler les différents mécanismes afin de récupérer et transformer l'information.

traite a été définie, et les différents agents peuvent se
e aux besoins spécifiques du client [FERB 00].

10. UTILISATION DU MODELE AALAADIN POUR L'ORGANISATION HETERARCHIQUE :

Nous utilisons une architecture hétérarchique sur laquelle est implémenté un SMA contenant plusieurs types d'agents. Ce mécanisme peut prendre en charge l'aspect distribué du problème tout en assurant la propriété de communication entre les sites.

Le siège directeur contient **l'agent superviseur**, les autres sites contiennent chacun, un **agent stock**, un **agent ressource** et un **agent livraison** qui sont contenus dans des groupes comme le montre la figure 4.10 (exemple de deux sites de production).

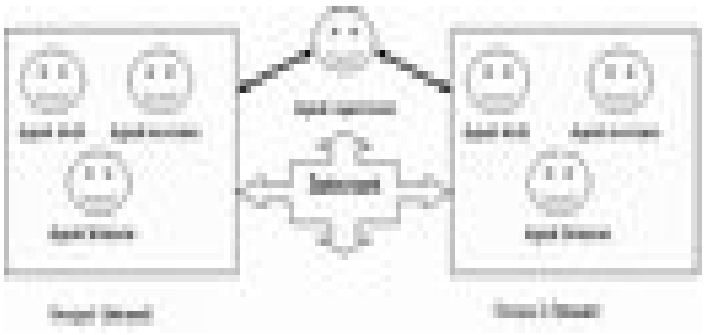

Figure 4.10 Répartition des agents dans des groupes

Nous rappelons que les agents que nous avons crées sont inspirés du modèle Aalaadin élaboré par J.Ferber.

En suivant le principe du modèle Aalaadin, nous dégageons plusieurs notions telles que:

- Le ***Rôle*** qui représente la fonction d'agent dans le système.
- Les ***Propriétés*** de l'agent: les agents doivent avoir au préalable des connaissances sur leurs capacités : le nom de la ressource, les opérations réalisables par cette ressource et leurs différentes propriétés, etc.
- Le ***Groupe*** auquel il appartient et dans le quel il joue le rôle ***Rôle*** (selon l'agencement), car les agents doivent avoir aussi des connaissances sur le voisinage qui représente les ressources qui lui sont directement connectées : les groupes auxquels ils appartiennent.

La figure 4.11 montre les différents groupes d'agents du modèle que nous avons proposé, ainsi que le mode de communication entre ces agents.

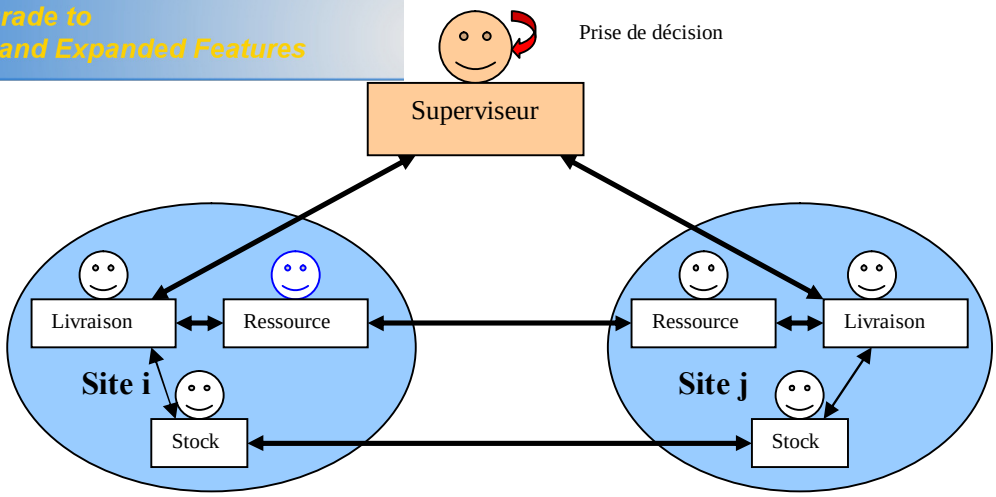

Figure 4.11 Le modèle de système multi agents mis en place

Nous présentons dans ce qui suit, lôarchitecture de chacun des agents utilisés dans le système.

L'agent superviseur :

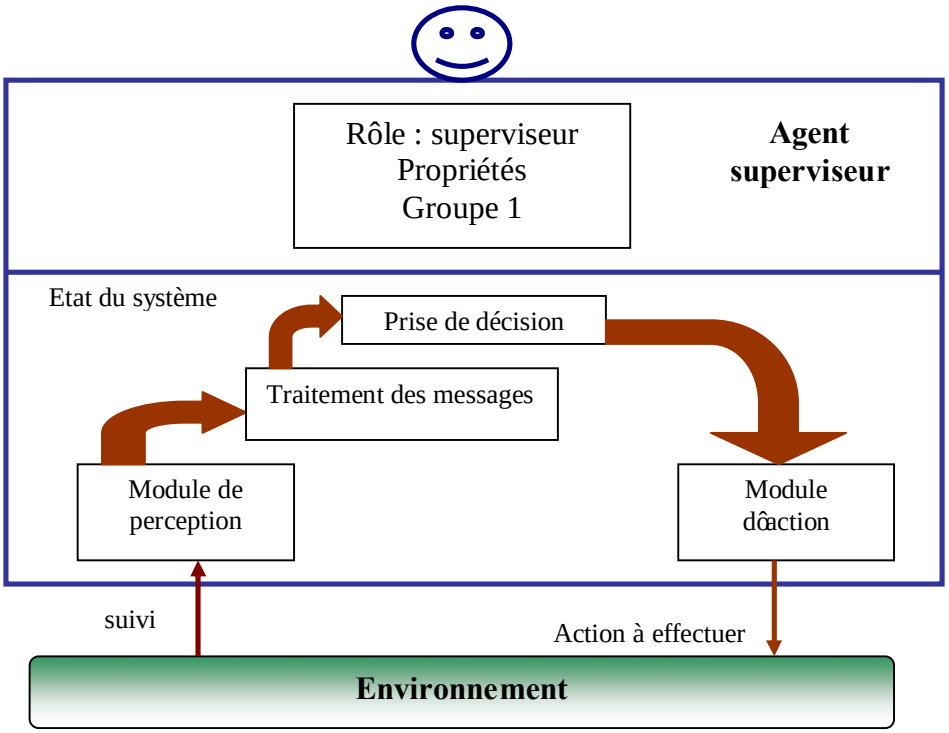

Figure 4.12 Architecture de lôagent superviseur

Lôagent superviseur est lôagent doté de la capacité de traitement de messages reçus de son environnement en plus de sa capacité de prise de décision à travers le lancement de lôAG.

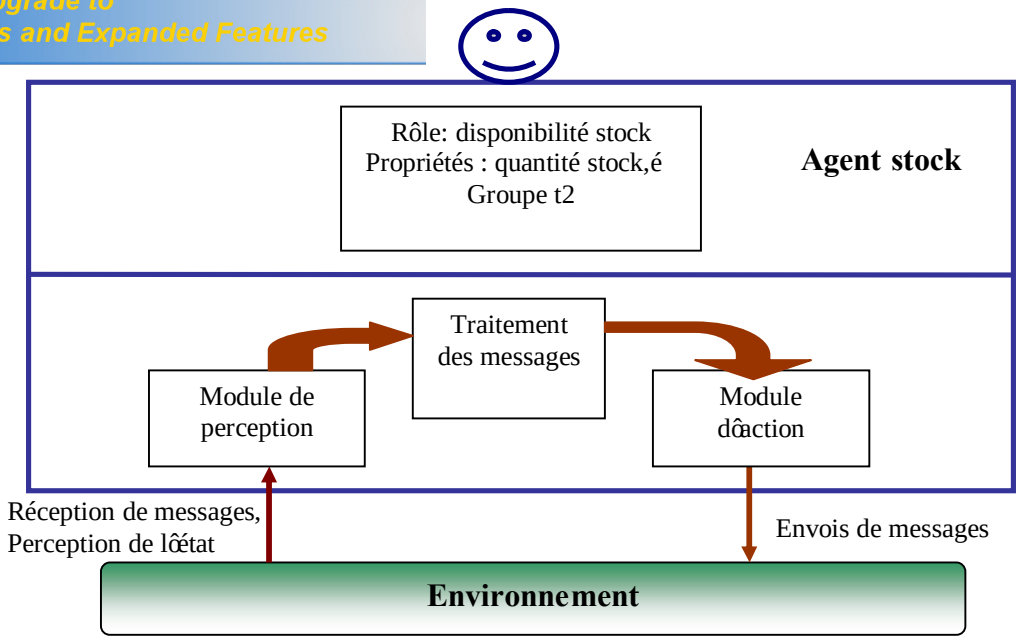

Figure 4.13 Architecture de lôagent stock

Lôagent stock ne prend aucune décision, il ne fait que traiter, recevoir et envoyer des messages au niveau de son environnement.

L'agent ressource :

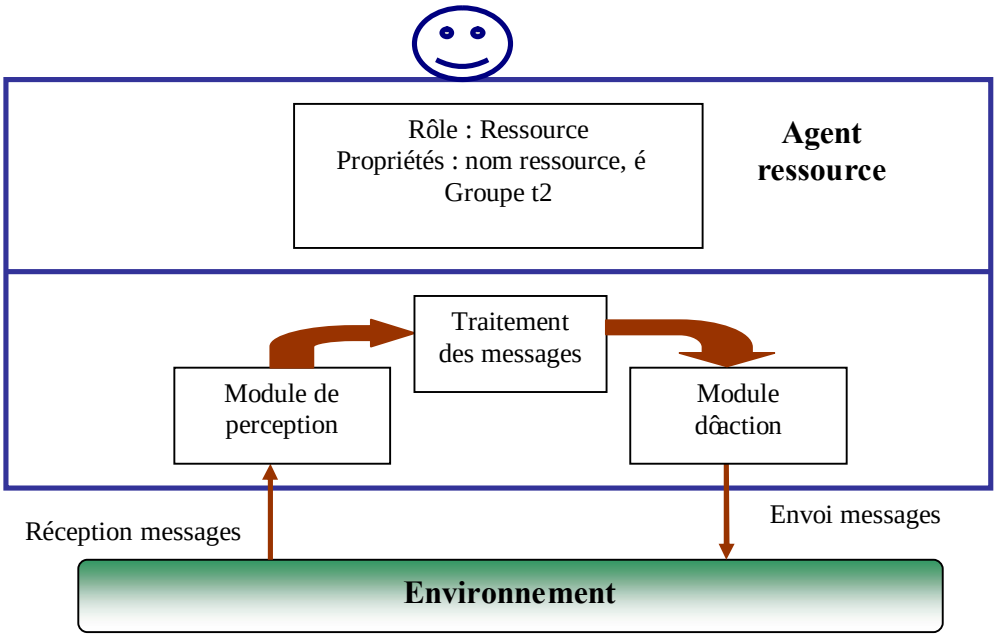

Figure 4.14 Architecture de lôagent ressource

Lôagent ressource nôest pas doté dôun module de prise de décision, sa participation ne se limite que dans le fait dôenvoyer, de recevoir et de traiter les messages.

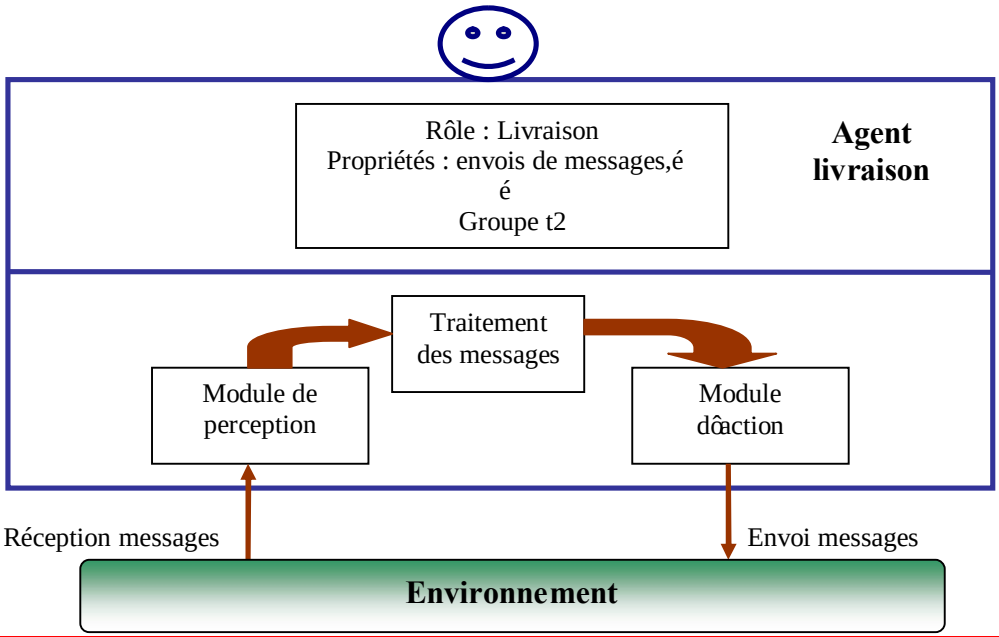

Figure 4.15 Architecture de lôagent livraison

A travers ce schéma, nous remarquons que cet agent à son tour nôest pas doté de capacité de décision; il reçoit, envoie et traite les messages.

11. CONCLUSION

Dans ce chapitre, nous avons essayé de définir le plus précisément possible, ce quôest un agent et ce quôest un système multi- agents. Nous avons aussi présenté quelques notions élémentaires qui sôy rapportent. Lôidée de base des systèmes multi agents est de faire cohabiter des entités autonomes, capables de percevoir et dôagir sur lôenvironnement en les faisant interagir et collaborer pour résoudre des problèmes complexes et distribués. A cet effet, il est apparu judicieux dôexpliquer lôinteraction entre les agents, la nature de la communication et surtout lôorganisation dans la quelle se fait cette coordination.

Nous avons aussi énuméré quelques plates-formes existantes pour le développement dôapplications se basant sur les SMA et avons explicité notre choix dôutilisation de la plate-

nel Aalaadin [FERB 02] qui nous a inspiré dans la
nts a aussi été présenté.

Il faut noter que notre motivation pour l'utilisation des SMAs dans ce travail est due au caractère distribué de l'intelligence des SMAs qui permet de résoudre des problèmes, de manière globale, par des entités autonomes.

Nous avons aussi montré l'architecture globale de notre SMA et identifié les rôles et les propriétés des différents agents le constituant en présentant leur architecture. Le comportement de ces agents sera explicité au niveau du chapitre 6 à travers différents diagrammes UML.

Notons que le SMA mis en place a été adapté à une organisation hétérarchique utilisée au sein de la plate-forme J2EE. Ces notions seront présentées dans le chapitre suivant.

CHAPITRE 5 :

ARCHITECTURE MULTI- TIERS POUR UNE ORGANISATION HETERARCHIQUE

Les technologies de calcul informatique et dôapplications sous réseaux ont connu un long chemin durant ces vingt dernières années et se sont de plus en plus détachées des plates-formes matérielles sur lesquelles elles sôexécutent. Aussi, les technologies ont évolué de monolithiques à ouvertes et ensuite vers des systèmes distribués [GOEL 06].

Notons aussi que les besoins des entreprises se font ressentir en plusieurs points, par exemple dans la nécessité de se doter dôune capacité de réaction très rapide face aux nouvelles orientations du marché. La fiabilité et la disponibilité sont aussi nécessaires afin de faire face à la croissance du monde de lôéconomie actuel, il est donc indispensable que les opérations des entreprises basées sur le web par exemple fonctionnent correctement et que les transactions commerciales soient fiables.

Dôun autre côté, lôentreprise doit aussi assurer la confidentialité des informations détenues tout en garantissant et en gérant la montée en charge du nombre dôutilisateurs et en utilisant de manière optimale les ressources du système.

Enfin, les applications de lôentreprise doivent être capables dôintégrer les systèmes dôinformation existants. En effet, côest dans la capacité dôune entreprise à combiner les anciennes et les nouvelles technologies que réside la clé de son succès.

Dans ce qui suit, nous présentons les différentes architectures dôapplications proposées et mises en place pour répondre à certains de ces besoins, pour arriver à la plate-forme J2EE et ensuite passer à lôarchitecture dôagencement que nous avons utilisée, c'est-à-dire lôarchitecture hétérarchique.

2. LES ARCHITECTURES MULTI TIERS [BEAU 99] :

Une architecture multi- tiers est un modèle dôarchitecture dôapplications dans lequel on sépare la présentation, les traitements et les données. Lôobjectif à atteindre est de permettre une évolution de lôun de ces trois tiers de façon plus ou moins indépendante des deux autres. Lôimplémentation physique de ces architectures est souvent soumise aux contraintes de lôexistant. Ainsi, elles sont parfois mises en oeuvre au travers de plates-formes et de systèmes

plique leur conception, leur mise au point et leur

Les enjeux liés à cette séparation logique sont considérables :

ÅAméliorer la réactivité de mise à disposition des applications par la réutilisation de ñbriquesò existantes,

ÅAutoriser lôaccès au système par le biais de canaux variés tels que lôInternet,...

ÅCapitaliser sur lôexistant.

Les architectures multi- tiers prennent en compte la montée en charge par la séparation des tiers et la possibilité de déploiement sur plusieurs serveurs physiques.

Le problème de lôinteropérabilité nôest cependant pas définitivement réglé par les architectures 3-tiers. Faire coopérer deux applications de manière synchrone, en mode objet, tisse aussi des adhérences fortes au niveau du composant entre ces applications (lôIDL des objets manipulés). Si on ne fait pas attention, on risque de déplacer le problème des adhérences de la donnée vers les objets. La mise en place dôun serveur dôapplications doit sôinsérer dans un projet dôarchitecture global intégrant cette problématique.

Dans ce qui suit, nous allons présenter les différentes architectures multi niveaux existantes.

2.1 Architecture à deux niveaux :

Dans une application classique à deux niveaux, la charge de traitement est attribuée au poste client tandis que le serveur se contente de contrôler le trafic entre lôapplication et la base de données.

Par conséquent, les performances de lôapplication baissent du fait des ressources limitées du PC, celle-ci doit formuler de nombreuses demandes de données avant de pouvoir satisfaire les requêtes de lôutilisateur. Ces requêtes à la base de données peuvent fortement pénaliser le trafic sur le réseau.

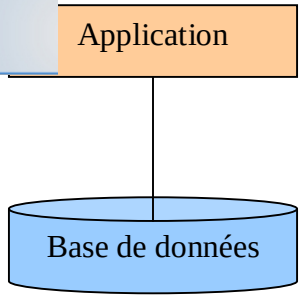

Figure 5.1 Architecture à deux niveaux

Lôapproche à deux niveaux pose un problème de maintenance. En effet, la moindre modification apportée à lôapplication peut avoir des répercussions sur la totalité de la base utilisateur. Même sôil est possible dôautomatiser ce processus, il demeure nécessaire de remettre à niveau chaque installation cliente. Aussi, certains utilisateurs peuvent ne pas se sentir prêts à affronter une réorganisation complète et ignorer les modifications, alors quôun autre groupe dôutilisateurs insistera pour appliquer les changements immédiatement. Le résultat peut ainsi être des installations clientes utilisant des versions différentes dôune même application.

2.2 Architecture à trois niveaux :

Pour éliminer ce risque, la communauté informatique a créé la notion dôarchitecture à trois niveaux. Ainsi, une application est divisée en trois niveaux logiques, distincts, chacun dôentre eux est pourvu dôun ensemble dôinterfaces bien défini.
Le premier, appelé **niveau présentation**, est généralement constitué dôune interface utilisateur graphique. Le niveau intermédiaire, ou **niveau métier**, recouvre la logique métier ou logique applicative. Le troisième niveau, ou **niveau données**, contient les données nécessaires à lôapplication.

Le niveau intermédiaire (logique applicative) est composé du code appelé par lôutilisateur (via le niveau présentation) pour extraire les données utiles. Le niveau présentation reçoit alors les données et les formate en vue de leur affichage.

applicative de lôinterface accentue dôune manière otion de lôapplication. Il devient alors possible de construire et de déployer de nombreuses interfaces utilisateurs sans avoir à modifier la logique applicative, à la condition toutefois, que cette dernière soit dotée dôune interface parfaitement définie avec le niveau présentation.

Le troisième niveau abrite les données nécessaires à lôapplication. Il peut sôagir de nôimporte quelle source dôinformation: base de données dôentreprise sous Oracle ou Sybase, jeu de documents XML(côest à dire des données ayant été enregistrées dans les documents élaborés selon les spécifications XML), ou encore service dôannuaire tel quôun serveur LDAP.

Outre les bases de données relationnelles classiques, une application peut aussi avoir accès à de nombreuses sources de données différentes relatives à lôentreprise.

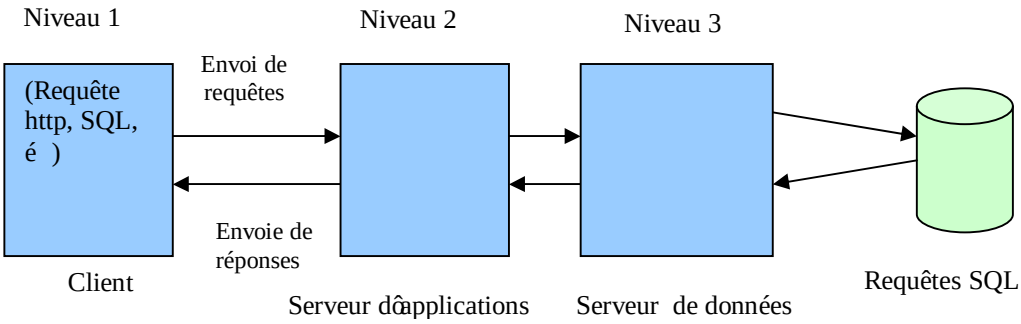

Figure 5.2 : Architecture à trois niveaux

2.3 Architecture multi- niveaux

Ainsi, il nôexiste pas de solution prête pour définir les niveaux applicatifs dôun système multi niveaux. Un système multi niveaux peut supporter des configurations différentes. Au sein dôune architecture multi niveaux, la logique applicative se décompose selon les fonctions. Cette architecture se subdivise de la façon suivante :

1- Interface utilisateur, chargée de gérer les interactions entre les utilisateurs et lôapplication; il peut sôagir dôun navigateur web, dôune application de bureau plus puissante ou encore dôun mobile WAP;

nt de définir ce que doit afficher lôinterface utilisateur et la manière dont les requêtes de lôutilisateur seront traitées. Selon les interfaces utilisateurs prises en charge, il y aura un besoin de disposer de versions légèrement différentes de la logique de présentation afin de traiter la requête cliente de façon appropriée;

3- Logique métier, elle modélise les règles métier de lôapplication, souvent via lôinteraction avec les données de lôapplication;

4- Services d'infrastructure, qui fournissent des fonctionnalités supplémentaires nécessaires aux composants de lôapplication, tels quôun service de messagerie, de support transactionnel, etc.;

5- Niveau données, abritant les données de lôentreprise. Les applications reposant sur ce type dôarchitecture utilisent un motif de conception MVC (Model- View- Controller ou Modèle - Vue ï Contrôleur). En dôautres termes, les données (le modèle) sont séparées de la partie présentation de lôinformation (la vue).

Cêest la logique applicative /métier (le contrôleur) qui, dans lôintervalle, surveille le flux dôinformations. Ainsi, la conception dôune application est fondée sur lôinteraction entre ces trois composants fonctionnels : le modèle, la vue et le contrôleur.

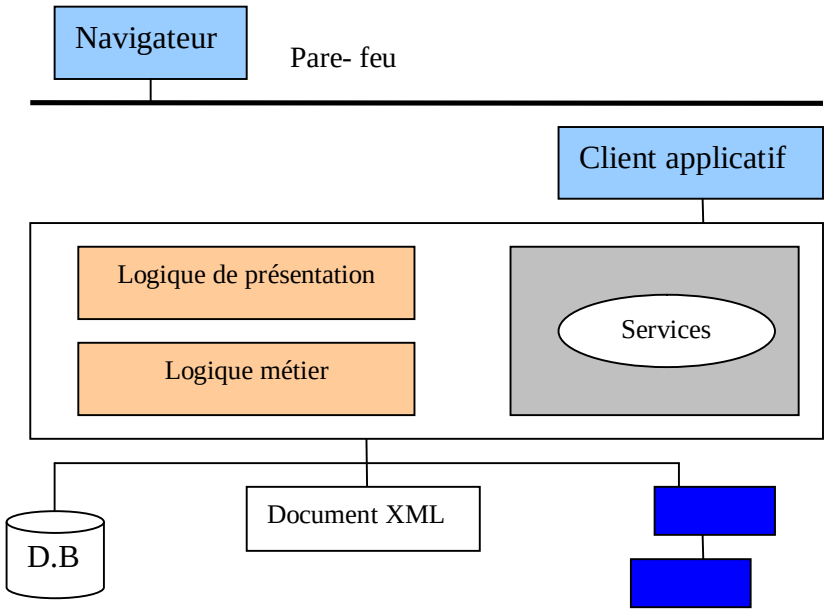

Figure 5.3 Architecture multi niveaux

Jusquà présent, nous nâavons vu que lâarchitecture dâun seul type dâapplications. Il existe donc le risque de se retrouver avec de nombreuses applications différentes les unes des autres, présentant même des architectures dissemblables et incapables de communiquer entre elles. Au sein dâune entreprise, lâobjectif est de créer un ensemble plus cohérent pour pouvoir répondre aux besoins cités ci dessus.

Pour transformer un système multi niveaux en un système dâentreprise, il suffit dâétendre le niveau intermédiaire de sorte quâil accueille plusieurs objets applicatifs plutôt quâune application unique. Ces derniers doivent être pourvus dâune interface leur permettant de coopérer les uns avec les autres.

Une interface peut être considérée comme un contrat. Chaque objet stipule, via son interface, les paramètres quâil accepte et les résultats quâil refusera. Les objets applicatifs peuvent ainsi communiquer les uns avec les autres à travers leur interface :

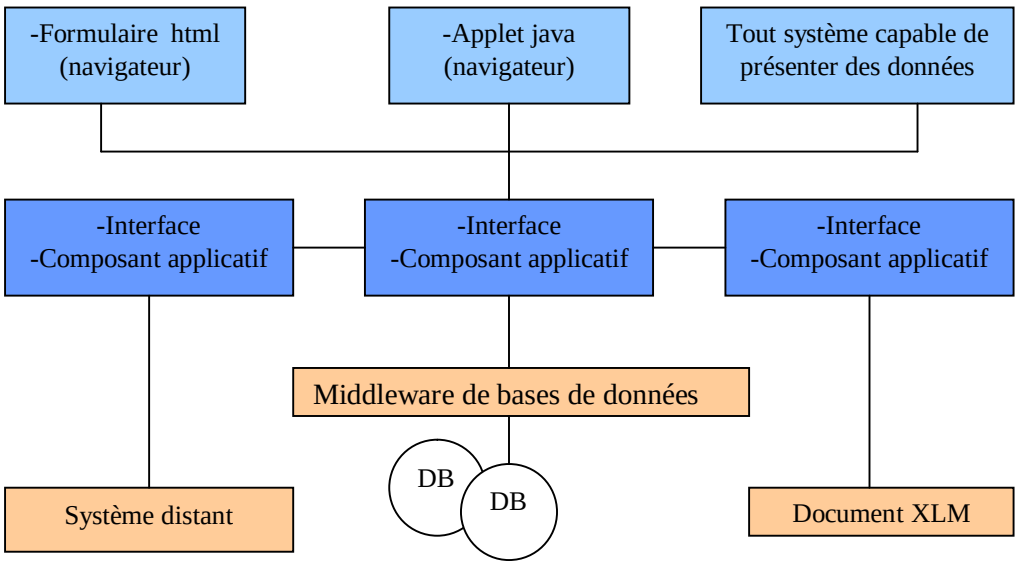

Figure 5.4 Architecture dâentreprise

Avec lâarchitecture dâentreprise, il est possible de disposer de plusieurs applications et dâun ensemble de composants communs à toute lâentreprise. Ainsi, en créant un ensemble

s par la totalité de lôentreprise, la normalisation des
éliorée dôoù la force de cette architecture.

Si une règle métier change, alors ces changements ne doivent être répercutés quôau niveau de lôobjet métier et, si besoin est, à son interface et tout objet ayant accès à cette interface.

2.5 Les architectures transactionnelles :

Le transactionnel est issu des technologies Mainframe dôIBM des années 70. Il sôagissait alors de mettre massivement à disposition de terminaux des applications en mode texte. Le terme de transaction vient du fait que plusieurs écrans peuvent sôenchaîner avant quôune réelle modification dans le système ne soit réalisée. Ces transactions sôentendent en revanche mono-moniteur (CICS ou IMS) et mono- base.

Au niveau de cette architecture, nous remarquons que:

-La gestion du transactionnel est faite pendant le développement: elle nécessite lôécriture dôun code spécifique utilisant des implémentations de TxAPI dépendantes du moniteur utilisé (aucune norme nôest pratiquement imposée),
-Bien que lôX/Open ait défini une norme dôinteropérabilité entre les TM (XA+), celle-ci est très rarement implémentée dans les moniteurs transactionnels, et ne répond pas à un besoin réel. Les communications sôeffectuent par lôintermédiaire de passerelles.

2.6 Les architectures à base de composants :

On définit le composant comme un objet qui adhère à un modèle. Il implémente un ensemble de méthodes normées lui permettant :

ÅDôêtre exécuté dans un serveur dôapplications en supportant le transactionnel, la sécurité, le multi- accès, lôindépendance à la localisation, etc. sans ajout de code particulier,

ÅDôêtre intégré aisément dans un AGL dans la phase de développement, grâce aux méthodes auto- descriptives dôintrospection.

Cette notion permet un réel assemblage de composants avec des outils de haut niveau, sans se soucier des détails de lôimplémentation.

également un ensemble dôobjets et de méthodes, et qui permettent de gérer plus finement les aspects transactionnels et la sécurité.

Ces deux facteurs permettent de diminuer considérablement la complexité de mise en oeuvre dôune architecture 3-tiers.

2.6.1 Les modèles

Deux modèles de composants se partagent aujourdôhui le marché :

ÅLe modèle COM, issu de Microsoft et du monde OLE, désormais renommé ActiveX et dont la première implémentation côté serveur date de 1996 (MTS 1.0),

ÅLe modèle Enterprise JavaBean (EJB), spécification multi- éditeurs dont Sun et IBM sont à lôorigine, et dont les premières implémentations apparaissent actuellement sur le marché. Les EJBs sont implémentés sur une plate-forme J2EE.

Nous présentons maintenant la plate-forme J2EE, une plate-forme qui a su répondre aux besoins des entreprises en leur offrant de nombreuses possibilités.

3. LA TECHNOLOGIE J2EE :

3.1 Introduction :

J2EE est l'acronyme de Java 2 Entreprise Edition. Cette édition est dédiée à la réalisation d'applications pour entreprises. [Net 4]

Vu lôusage fréquent des réseaux dans les entreprises et que ces applications sont amenées à se développer, les plus grands fournisseurs de logiciels ont proposé leur propre solution. J2EE est la réponse de SUN qui a donné naissance aux architectures multi- tiers afin de couvrir les inconvénients de lôarchitecture classique client- serveur.

2EE est un environnement serveur dôapplications ecture multi- tiers, côest a dire que ce sont des applications distribuées qui interagissent par lôintermédiaire dôun réseau.

J2EE est basée sur J2SE (Java 2 Standard Edition) qui contient les API de base de Java. [Net 4] et est composée de deux parties essentielles :

- Un ensemble de spécifications pour une infrastructure dans laquelle sôexécutent les composants écrits en Java: un tel environnement se nomme serveur d'application.
- Un ensemble d'API qui peuvent être obtenues et utilisées séparément. Pour être utilisées, certaines nécessitent une implémentation de la part d'un fournisseur tiers.

Les objectifs de J2EE incluent une meilleure qualité et la maintenabilité, mais surtout ce qui faisait défaut jusquôici, la portabilité entre systèmes. En plus de cela, l'utilisation de J2EE pour développer et exécuter une application présente plusieurs avantages:

- une architecture d'application basée sur les composants et permettant un découpage de l'application et donc une séparation des rôles lors du développement;
- la possibilité de s'interfacer avec le système d'information existant grâce à de nombreuses API : JDBC, JNDI, JMS, JCA ...

- la possibilité de choisir les outils de développement et le ou les serveurs d'applications utilisés, qu'ils soient commerciaux ou libres;

- J2EE permet une grande flexibilité dans le choix de l'architecture de l'application en combinant les différents composants. Ce choix dépend des besoins auxquels doit répondre l'application mais aussi des compétences dans les différentes API de J2EE. L'architecture d'une application se découpe idéalement en au moins trois tiers. La partie cliente est la partie qui permet le dialogue avec l'utilisateur. Elle peut être composée d'une application *stand alone*, d'une application web ou d'applets;

- J2EE offre la partie métier : c'est la partie qui encapsule les traitements (dans des EJB ou des JavaBeans) [Net 4].

basé composant simplifient ainsi le développement et le déploiement dôapplications d'entreprise. Elle contrôle l'infrastructure et soutient les services Web, définit la norme pour développer des applications à plusieurs niveaux d'entreprise (multi- tiers), réduit le coût et la complexité de développement et donne comme résultat des services pouvant être rapidement déployés et facilement augmentés.

Lôévolution de nouveaux services de logiciel a donné une impulsion additionnelle à cette nouvelle architecture et l'utilisation croissante de l'Internet et les intranets pour des applications d'entreprise ont contribué à rendre la technologie J2EE indispensable.

Le but principal est dôarriver à appliquer la notion : "Write Once, Run Anywhere" et donc de bénéficier des composants déjà écrits se trouvant nôimporte où dans le réseau mondial pour gagner en termes de temps et dôargent. Pour cette raison, la compagnie SUN, depuis lôinvention du langage JAVA, a pensé à produire une plate-forme qui supporte les applications distribuées sans perdre lôacquis, c'est-à-dire préserver les systèmes dôinformation déjà existants. Ceci dit, nous pouvons ajouter que cêest une orientation Web basée sur la modularisation (notion de composants ou tout simplement Beans) et la réutilisation.

Figure 5.5 Vue générale de lôarchitecture J2EE

2EE:

En plus des avantages cités plus haut, la technologie J2EE est à la recherche dûne indépendance maximale du code applicatif car elle est caractérisée par [DECH 03] :

-Lôabsence de liaison statique entre modules de code applicatif (composants).
-Lôabsence de liaison statique entre modules de code applicatif et services plate-forme
-Le fait dôéviter lôutilisation de code «technique» dans le code applicatif.

Nous pouvons ainsi dire que J2EE représente un pas important vers du code applicatif plus réutilisable.

3.3 L'environnement d'exécution des applications J2EE :

J2EE propose des spécifications pour une infrastructure dans laquelle s'exécutent les composants. Ces spécifications décrivent les rôles de chaque élément et précisent un ensemble d'interfaces pour permettre à chacun de ces éléments de communiquer. Ceci permet de séparer les applications et l'environnement dans lequel il s'exécute. Les spécifications précisent à l'aide des API un certain nombre de fonctionnalités que doit implémenter l'environnement d'exécution. Ces fonctionnalités sont de bas niveau ce qui permet aux développeurs de se concentrer sur la logique métier.

Pour exécuter ces composants de natures différentes, J2EE défini des conteneurs pour chacun de ces composants. Il définit pour chaque composant des interfaces qui leur permettront de dialoguer avec dôautres composants lors de leur exécution. Les conteneurs permettent aux applications d'accéder aux ressources et aux services en utilisant les API. Les appels aux composants se font par des clients via les conteneurs. Les clients n'accèdent pas directement aux composants mais sollicitent le conteneur pour les utiliser [Net 4].

Ces notions sont détaillées dans ce qui suit :

La technologie J2EE offre plusieurs fonctions à travers [DECH 03] :

- **Java Servlets :** lôAPI Java Servlet fournit une couche dôabstraction orientée objet dans le cadre de lôélaboration dôapplications Web dynamiques;

- **JavaServerPages (JSP) :** Cette extension permet de valoriser davantage les applications Web J2EE en permettant le développement dôapplications Web basées sur les modèles;

Lôutilisation des JSP est particulièrement intéressante pour produire des pages HTML dynamiques, comparée à lôimplémentation directe dôune servlet. Il faut toutefois bien en saisir lôapport. Lorsquôun client invoque, via une requête, un serveur dôapplications, il ne se contente généralement pas de demander la simple production dôune page HTML. Lors de la réception de cette demande, le serveur peut être amené à insérer des données, à créer ou à récupérer un objet pour modifier certaines valeurs ou exécuter des fonctions plus complexes. Côest seulement après avoir réalisé ces opérations que le serveur construit la réponse HTML faite au client. Le serveur a donc la charge dôexécuter la logique applicative et de produire la couche graphique.

Dans la version initiale des JSP (JSP modèle 1, basée sur lôinvocation directe de la JSP par le client), cette séparation nôétait pas clairement définie. Si lôutilisation dôobjets métier sous forme de JavaBeans permet dôencapsuler lôaccès aux données et les traitements métier, elle laisse néanmoins aux JSP le soin de gérer les Beans en question (création, stockage dans la session, etc) et contraint à embarquer dans la JSP une partie de la logique métier [BERQ 01].

p

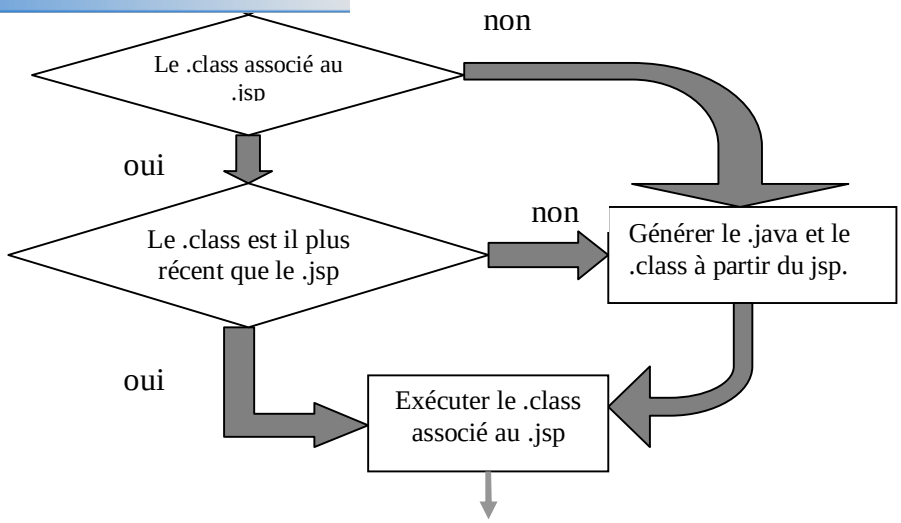

Figure 5.6 Représentation de lœexécution dœune JSP [BERQ 01]

- **Enterprise JavaBeans (EJB):** Il sœagit de composants Java pour les applications distribuées multi- niveaux. Cette extension fournit un moyen standard pour définir les composants côté serveur et une vaste infrastructure dœexécution pour lœhébergement des composants coté serveur;

- **Java Message Service (JMS) :** JMS fournit une API Java pour accéder à un service de messages et permet de publier et de souscrire à divers types de services middleware orientés message;

- **JDBC :** Egalement appelée paquetage facultatif **JDBC,** cette API contribue à améliorer lœAPI JDBC standard en facilitant lœobtention de connexion, les transactions distribuées, etc. ;

- Transactions: Nous citons: JTA et JTS;

- J2EE ConnectorArchitecture (JCA)

- **Corba**

e indépendant des plates- formes et des protocoles
ons de courrier électronique basées sur Java.

- **XML/SOAP:**

-Java API for XML Processing (JAXP)

-Java API for XML Registries (JAXR)

-Java API for XML-Based Remote ProcedureCall (JAX-RPC)

-SOAP with Attachments API for Java (SAAJ)

- **J2EE Deployment Specification (JSR-88)**
- **J2EE Management Specification (JSR-77)**

3.5 Différents types de composants J2EE :

Nous citons ici quelques composants J2EE [DECH 03]:

- **Clients :** Nous avons deux types de clients : les clients Web et les clients EJB

Les clients Web :

Sôexécutent normalement au sein des navigateurs Web. Pour ces clients, lôinterface utilisateur est générée du côté serveur, par exemple en HTML ou en XML, elle est ensuite téléchargée, puis restituée par les navigateurs. Ces clients utilisent http pour communiquer avec les conteneurs Web. Les composants applicatifs au sein des conteneurs Web contiennent des servlets Java et des pages JSP. Ces composants mettent en oeuvre les fonctionnalités requises pour les clients Web. Les conteneurs Web sont chargés dôaccepter les requêtes émises par les clients Web et de générer des réponses en utilisant les composants applicatifs;

Les clients EJB :

Sont des applications ayant accès aux composants EJB au sein des conteneurs EJB. Il existe deux types possibles de clients EJB:

- La première catégorie recouvre les applications clientes. Il sôagit dôapplications autonomes ayant accès aux composants EJB au moyen du protocole RMI-IIOP.

osants se trouvant à lôintérieur du conteneur WEB. En pages JSP peuvent également accéder aux composants EJB via le protocole RMI-IIOP, à lôinstar des applications clientes.

- **EJBs :** Un module EJB est une unité qui peut être déployée et est composée dôEJB, dôarchives JAR, etc. Les modules EJB sont empaquetés sous forme de fichiers JAR, et un descripteur de déploiement (ejb- jar.xml) est inclus dans le registre META-INF du fichier JAR.

- **Connecteurs**

- **Servlets (container Servlet) :** Ce sont des Composants protocolaires de J2EE (principalement utilisés pour la présentation Web), ils permettent le traitement dôinteractions réseau de type «requête/réponse» et lôencapsulation des flux entrant/sortant par un appel procédural (service). Leur hypothèse de base est lôexistence dôun réseau IP.

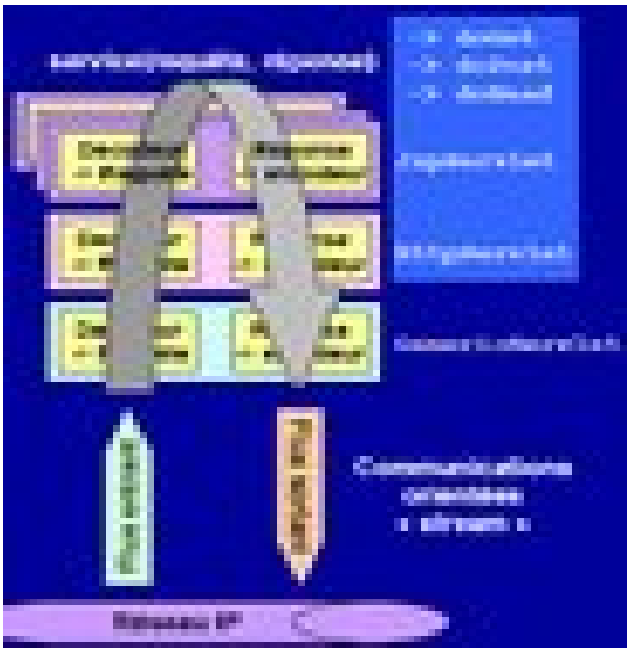

Figure 5.7 Composants servlets

Nous allons maintenant aborder lœensemble des technologies fournissant les mécanismes nécessaires à la conception dœapplications dœentreprise distribuées de grande taille.
Cet ensemble de technologies existantes se divise de la façon suivante :

Les technologies composants:

Ces technologies servent à contenir la partie la plus importante de lœapplication : la logique métier. Il existe trois types de composants : Les JSP, les servlets et les EJB.

Les technologies de services :

Ces technologies fournissent aux composants de lœapplication des services connexes leur permettant de fonctionner efficacement;

Les technologies de communication :

Ces technologies sont transparentes, en général implémentées selon une architecture multi- tiers.

Nous présentons dans ce qui suit la modélisation dœune architecture multi tiers dœaprès une organisation hétérarchique (voir chapitre 1).

4. MODELISATION D'UNE ORGANISATION HETERARCHIQUE D'UNE ENTREPRISE MULTISITES SELON UNE ARCHITECTURE MULTI TIERS

La figure 5.8 montre les différentes couches constituant les entreprises multi sites (modèle macroscopique):

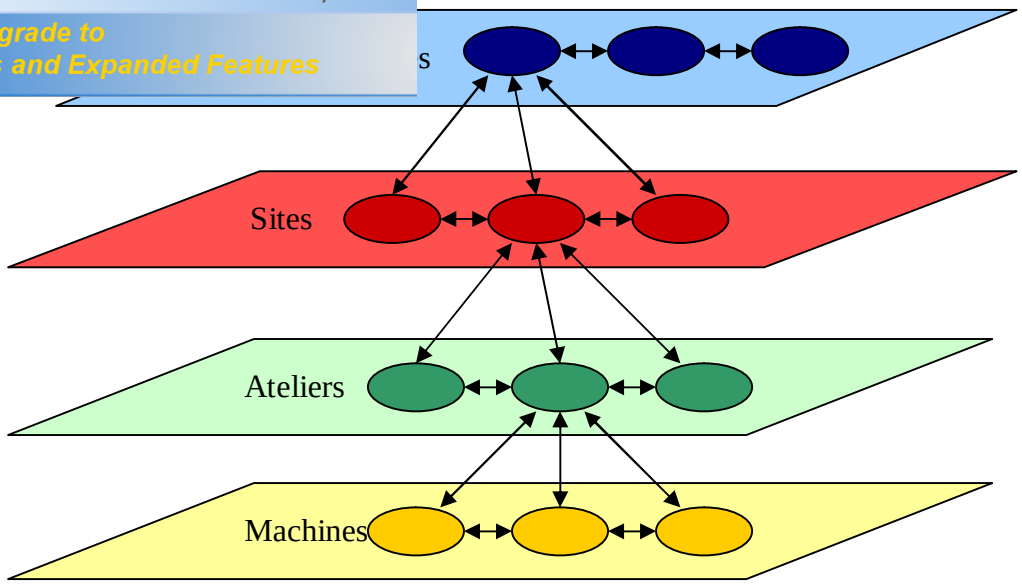

Figure 5.8 Modèle macroscopique dôentreprises multi sites

La figure 5.9 montre un exemple dôune entreprise multi sites selon organisation hétérarchique au sein dôune architecture multi tiers:

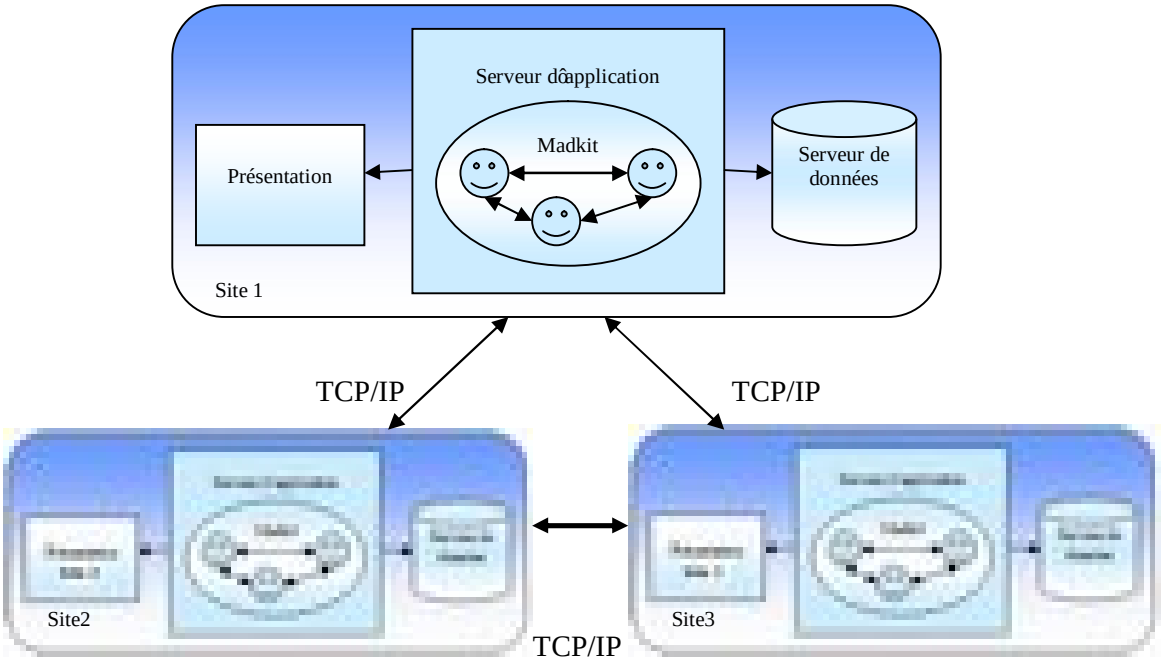

Figure 5.9 Modèle détaillé de lôentreprise multi sites selon notre approche

rganisation hétérarchique, par exemple, sur ce schéma hiérarchiques, le premier contient le site 1, le second contient les sites 2 et 3.

Le serveur dôapplication contient la plate-forme Madkit pour la gestion des différents agents, le serveur de données contient les données relatives aux sites de production de lôentreprise, et enfin, le poste client contient lôinterface via laquelle seront affichées et récoltées les données concernant chaque site, comme nous pouvons visualiser les différents diagrammes de Gantt locaux et globaux concernant la production multi sites.

En effet, lôagent superviseur est chargé du lancement de lôalgorithme génétique qui donnera un résultat représenté sous forme de diagramme de Gantt, Ce diagramme peut être affiché au niveau de chaque poste client.

Dôun autre côté, la plate-forme Madkit, se trouvant au niveau applicatif, a pour rôle la gestion des différents groupes dôagents.

Lôapport de lôorganisation hétérarchique est de permettre la communication entre les groupes dôagents se trouvant au même niveau hiérarchique, par exemple, ici entre les groupes contenus respectivement dans les sites 2 et 3, dôoù le gain de temps pour arriver à une solution satisfaisante pour le système. Dans le cas contraire, si une solution nôest pas trouvée, les agents contenus dans les groupes communiqueront avec les agents des autres niveaux afin dôobtenir une solution.

5. CONCLUSION

Nous pouvons dire que lôarchitecture hétérarchique résout les problèmes de lôarchitecture hiérarchique tout en utilisant ses points forts et en réduisant ses inconvénients.

Nous pouvons dire aussi que les architectures multi- tiers à base de composants peuvent être aujourdôhui le meilleur choix pour réaliser des applications de qualité car elles obligent à adopter une démarche de conception et séparent dans la réalisation les différents éléments IHM, logique et données.

Nous avons aussi présenté notre modèle multi tiers selon une organisation hétérarchique.

Dans le prochain chapitre nous présenterons la conception, la modélisation de notre approche, ainsi que son implémentation.

CHAPITRE 6

CONCEPTION, RÉALISATION ET DISCUSSION DES RESULTATS

Lôobjectif de ce projet est dôélaborer une méthode qui prend en charge les évolutions dôun système de production multi sites qui inclut le court, le moyen et le long terme.

Pour ce faire, nous avons opté pour une modélisation multi agents. Les algorithmes génétiques ont été utilisés en vue dôoptimisation.

2. MODELISATION

Notons que nous utilisons une architecture hétérarchique pour toutes les possibilités de communication inter- entités quôelle peut offrir.

Nous rappelons aussi que nous faisons face à un environnement distribué, ce qui nous fait penser à lôutilisation dôun SMA pour pouvoir répondre aux besoins locaux et globaux de chaque groupe ou site de lôentreprise.

2.1 Agentification du problème :

Comme nous lôavons précisé dans le chapitre 4, nous utilisons une architecture hétérarchique sur laquelle est implémenté le SMA.

Après avoir présenté ces notions fondamentales dans le chapitre 4, nous présentons dans ce qui suit, chacun des agents de notre système en donnant des précisions sur leurs comportements et les différentes actions quôils peuvent exécuter ...

2.1.1 L'agent superviseur :

Côest lôélément clé de notre système, car au niveau du module décisionnel de cet agent, un algorithme génétique est implémenté. Il a pour rôle de définir la planification de la production par rapport à toutes les tâches qui doivent être exécutées au niveau de tout le système. Côest lôentité du niveau le plus haut dans lôagencement hétérarchique (Chapitre 4) de notre système; il veille à ce que lôobjectif global du système soit atteint.

sera explicitée dans la section qui suit sous forme de

Les états possibles de cet agent:

- En cours dœexécution ;
- En attente;

Les différentes actions que peut effectuer un agent superviseur :

- Demande de lancement dœune tâche ;
- Demande de transfert ;
- Traitement dœune requête (que nous allons voir dans la prochaine section)

2.1.2 L'agent stock :

Cet agent possède une seule et unique exigence à satisfaire *" il veille à ce qu'un lot soit fabriqué sans être retardé et à gérer au mieux le niveau du stock "*.

Les états dœun agent Stock :

- Plein : Capacité maximale atteinte ;
- Capable de recevoir encore: capacité maximale non atteinte ;
- Réception dœordre de stockage ;
- Réception dœordre de transfert.

Les différentes actions que peut effectuer un agent stock:

- Accepter une requête de stockage ;
- Refuser une requête de stockage ;
- Accepter une requête de transfert ;
- Refuser une requête de transfert ;
- Envoyer un message de rupture de stock.

2.1.3 L'agent ressource :

Cet agent représente lœétat et lœévolution dœune machine qui lui est dédiée. Il possède ainsi des connaissances sur son état, ses capacités, é

e en considération aussi, les requêtes quôil reçoit des autres agents. Il peut prendre plusieurs états parmi lesquels nous pouvons citer:

- A lôarrêt (en attente) ;
- En panne ;
- En marche ;
- Réception dôordre de lancement dôune tâche.

Les différentes actions quôil peut effectuer :

- Mise en marche,
- Arrêter machine,
- Répondre.

2.1.4 L'agent livraison :

Nous citons parmi les états dôun agent livraison :

- A lôarrêt;
- En marche;
- Réception dôordre de transfert;
- Réception dôalerte de panne;
- Réception dôalerte de rupture de stock.

Les différentes actions que peut effectuer un agent livraison :

- Mise en marche;
- Mise à lôarrêt;
- Envoi dôordres (de transfert, é).
- Envoi de requête de prise en charge de production.

2.2 Architecture globale du système

Les SMAs sont un mécanisme pouvant prendre en charge lôaspect distribué du problème en assurant la propriété de communication entre les sites.

ient un agent superviseur, les autres sites contiennent,
rce et un agent livraison.

Par exemple, la figure 6.1 est une illustration dôune entreprise ayant un siège directeur et cinq sites de production distribués, dont trois font partie du niveau un et deux font partie du niveau deux.

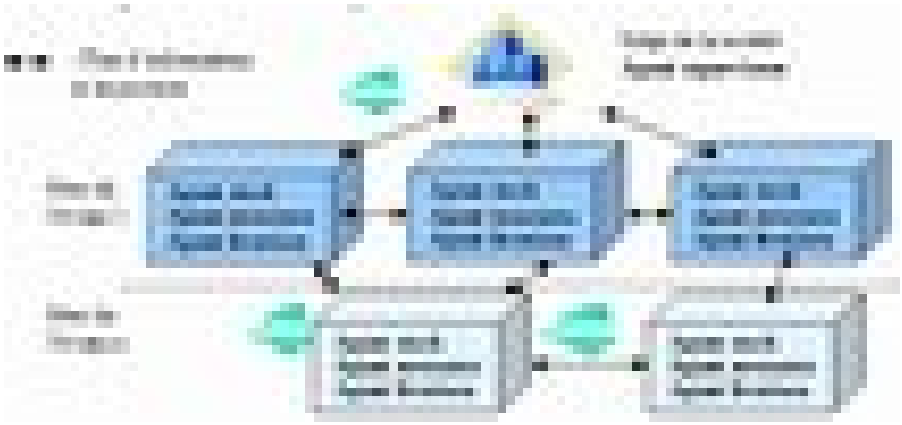

Figure 6.1 Représentation du modèle de lôentreprise

Le siège directeur contient **l'agent superviseur**, les autres sites (sites de production) contiennent, chacun, un **agent stock**, un **agent ressource** et un **agent livraison** appartenant à dans des groupes.

La communication est faite entre les groupes dôagents en suivant le principe de lôarchitecture Aalaadin (chapitre 3). Ce qui permet une grande flexibilité de communication entre ces groupes.

2.3 Modélisation des composants du système et leur interactions :

Afin de mieux expliciter notre modèle, nous présentons dans ce qui suit les diagrammes UML. Nous avons choisi dôutiliser le langage UML car il offre un ensemble de modèles permettant de représenter un système informatique du point de vue statique et dynamique et son utilisation prévue dans lôentreprise.

En plus de cela, lôusage de ces modèles par les informaticiens vise à améliorer la qualité des applications informatiques quôils développent.

le notre système ainsi que le comportement des agents et les interactions entre eux, nous avons utilisé les diagrammes de cas dûutilisation, de classes et de séquence :

Le diagramme de classes décrit les classes que le système utilise, ainsi que les liens qui existent entre elles.

Le diagramme de cas dûutilisation décrit la succession des opérations réalisées par un acteur. C'est le diagramme principal du modèle UML, celui où s'assure la relation entre l'utilisateur et les objets que le système met en oeuvre.

Le diagramme de séquence représente la succession chronologique des opérations réalisées par un acteur.

Ces diagrammes ont été développés en utilisant le modeleur BOUML présenté en annexe A.

Nous passons à la représentation de nos diagrammes UML :

Les figures 6.2, 6.3, 6.4, 6.5 représentent les diagrammes de cas dûutilisation associés à notre système :

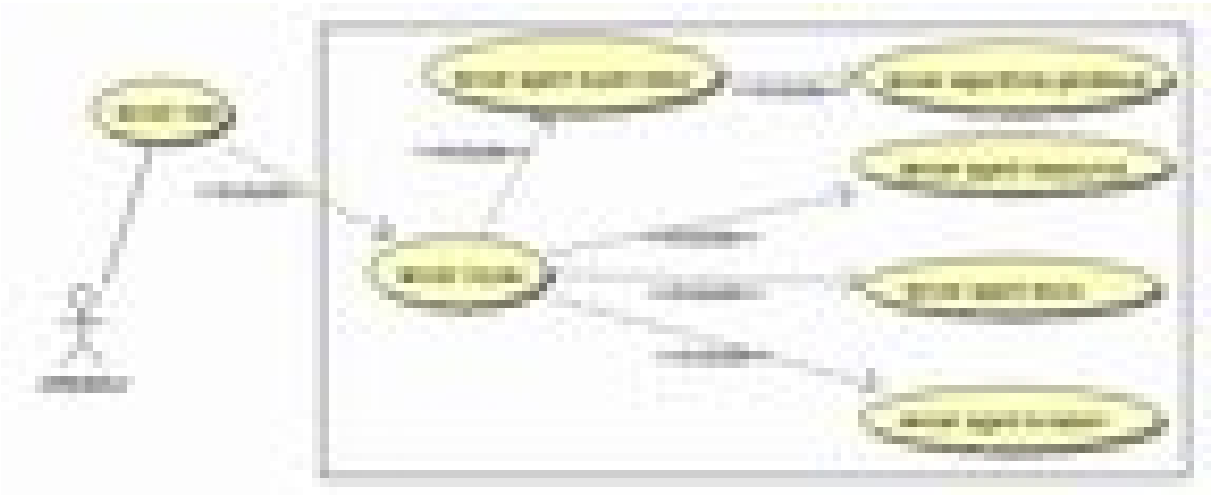

Figure 6.2 Diagramme de cas dûutilisations de notre système

Après lancement du noyau, les agents ressource, stock et livraison de tous les sites sont lancés en plus de lôagent superviseur, qui à son tour, lance lôexécution de lôalgorithme génétique.

Aussi, comme premier scénario pouvant apparaître au sein du système, nous avons la rupture de stock au niveau d'un site.

La figure 6.3 montre le cas d'utilisation représentant la rupture de stock au niveau d'un site :

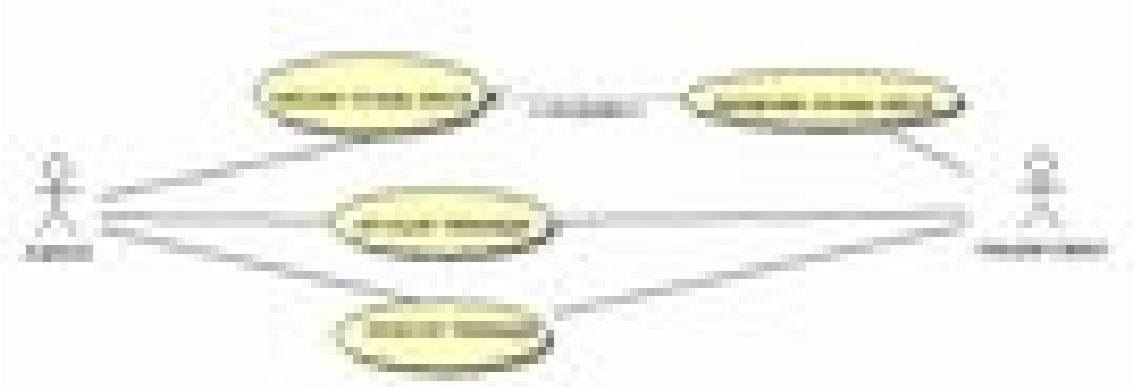

Figure 6.3 Diagramme de cas d'utilisation représentant la rupture de stock

En effet, lorsqu'une rupture de stock se produit dans un site, l'agent superviseur envoie un message aux agents stock des autres sites. Ceci se produit après que l'agent stock le demande aux agents stock se situant au même niveau hiérarchique que lui.

2.3.2 Scénario 2 : Panne d'une ressource

La figure 6.4 montre le cas d'utilisation représentant la panne d'une ressource au niveau d'un site :

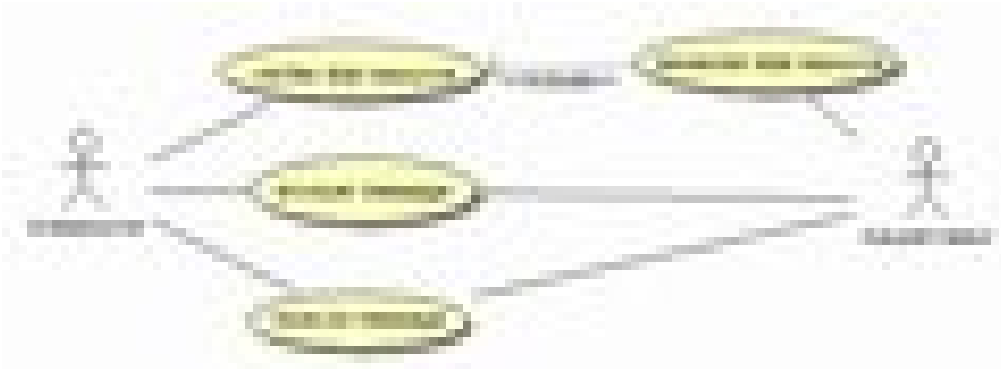

Figure 6.4 Diagramme de cas d'utilisation représentant la panne d'une ressource

L'agent superviseur envoie un message aux agents ressources des autres sites de niveaux différents pour pouvoir s'informer sur leurs états et prendre une décision.

e de cas dôutilisation concernant la communication entre les agents :

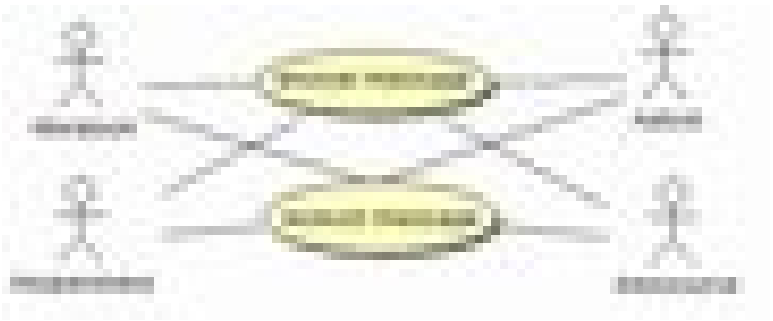

Figure 6.5 Diagramme de cas dôutilisation de lôagent livraison

En effet, nous pouvons remarquer lôinteraction entre tous les agents du système pour pouvoir arriver à une solution.

2.3.3 Diagramme de classe

La figure 6.6 montre le diagramme de classes associé à notre application :

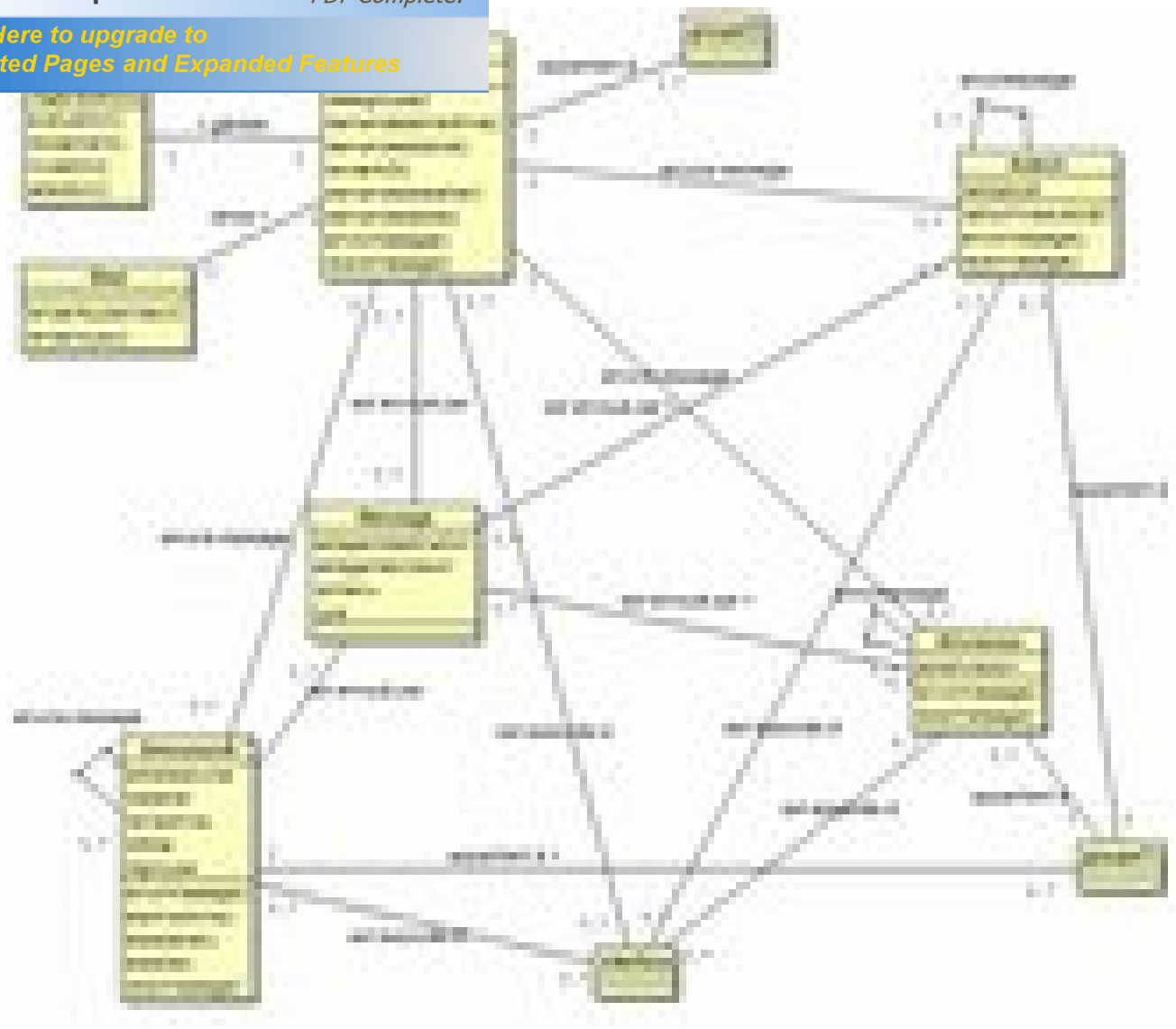

Figure 6.6 Diagramme de classes de notre modèle

Ce diagramme de classes montre les différentes entités qui constituent notre système et les liens qui existent entre elles. Nous avons comme classes principales:

-Les classes associées aux agents telles que : la classe ***Asuperviseur*** qui contient la méthode de lancement de lôalgorithme génétique, la classe ***Astock*** qui correspond aux agent stocks des différents groupes, la classe ***Aressource*** qui correspond aux agents ressources des sites et la classe ***Alivraison*** correspondant aux agents livraison,

- La classe ***message*** qui concerne tous les messages échangés entre les agents du système,

-La classe ***interface*** liée aux agents pour pouvoir visualiser les interactions entre eux et la classe ***population*** qui concerne les différentes générations ou populations de lôalgorithme génétique.

ramme de séquence associé à notre application:

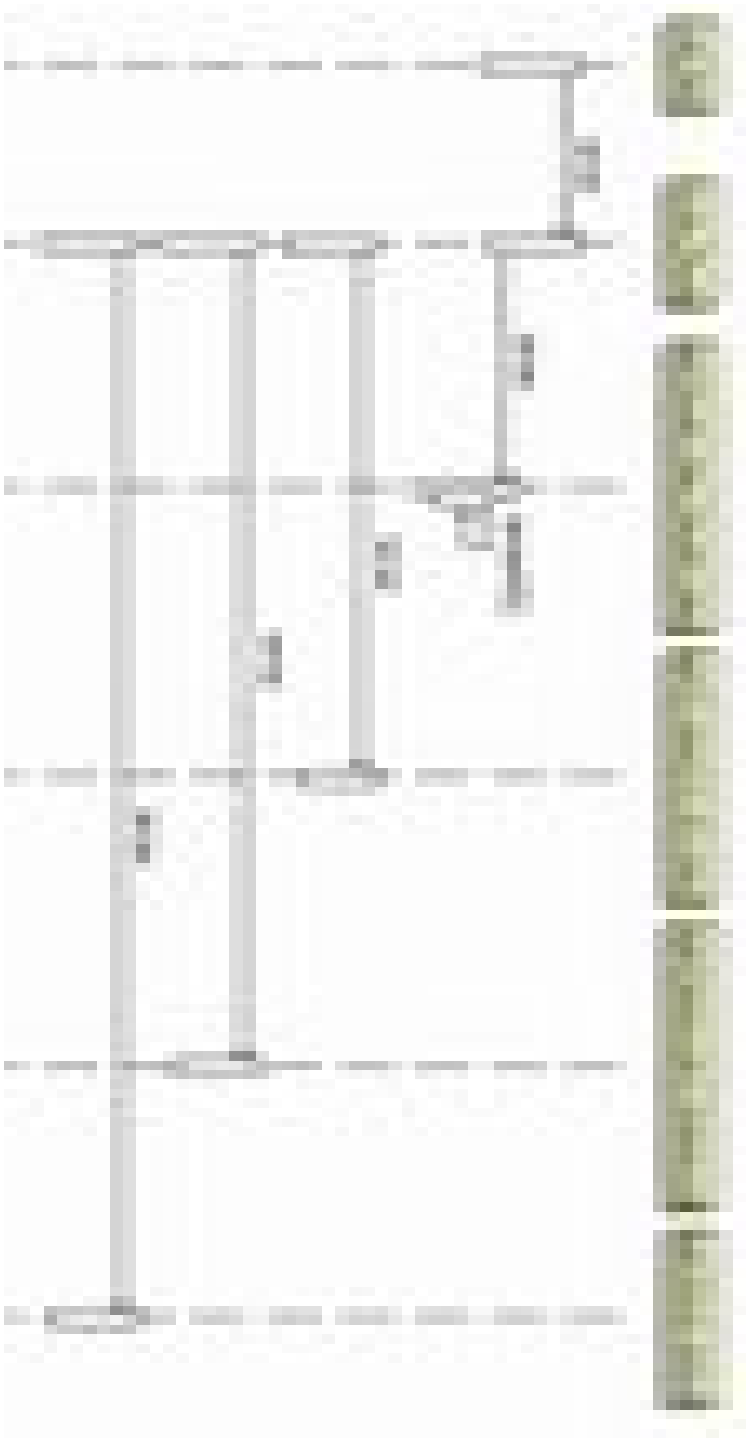

Figure 6.7 Diagramme de séquence du lancement de lôagent superviseur

Ce diagramme représente lôaspect initial de notre système, c'est-à-dire, le lancement des agents dont lôagent superviseur qui, à son tour, lance lôalgorithme génétique.

iagrammes de séquences pour les différents scénarios

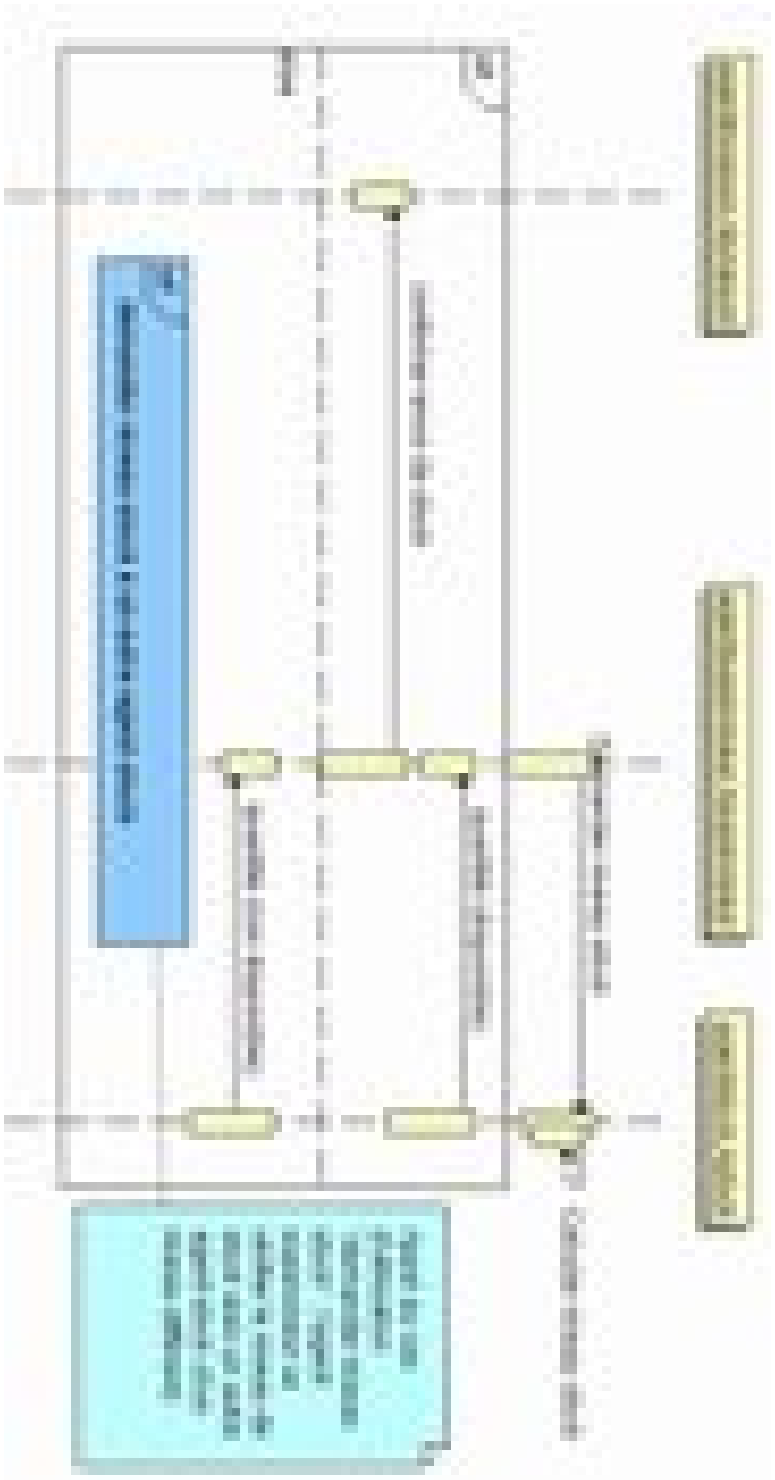

Figure 6.8 Diagramme de séquence du scénario 1 (rupture de stock à un niveau *n* de lôarchitecture hétérarchique)

Dans ce diagramme, nous remarquons lôinteraction entre un agent livraison avec lôagent superviseur pour pouvoir obtenir la quantité de stock manquante dôun site de niveau différent.

cié au scénario 2 (panne dôune ressource) :

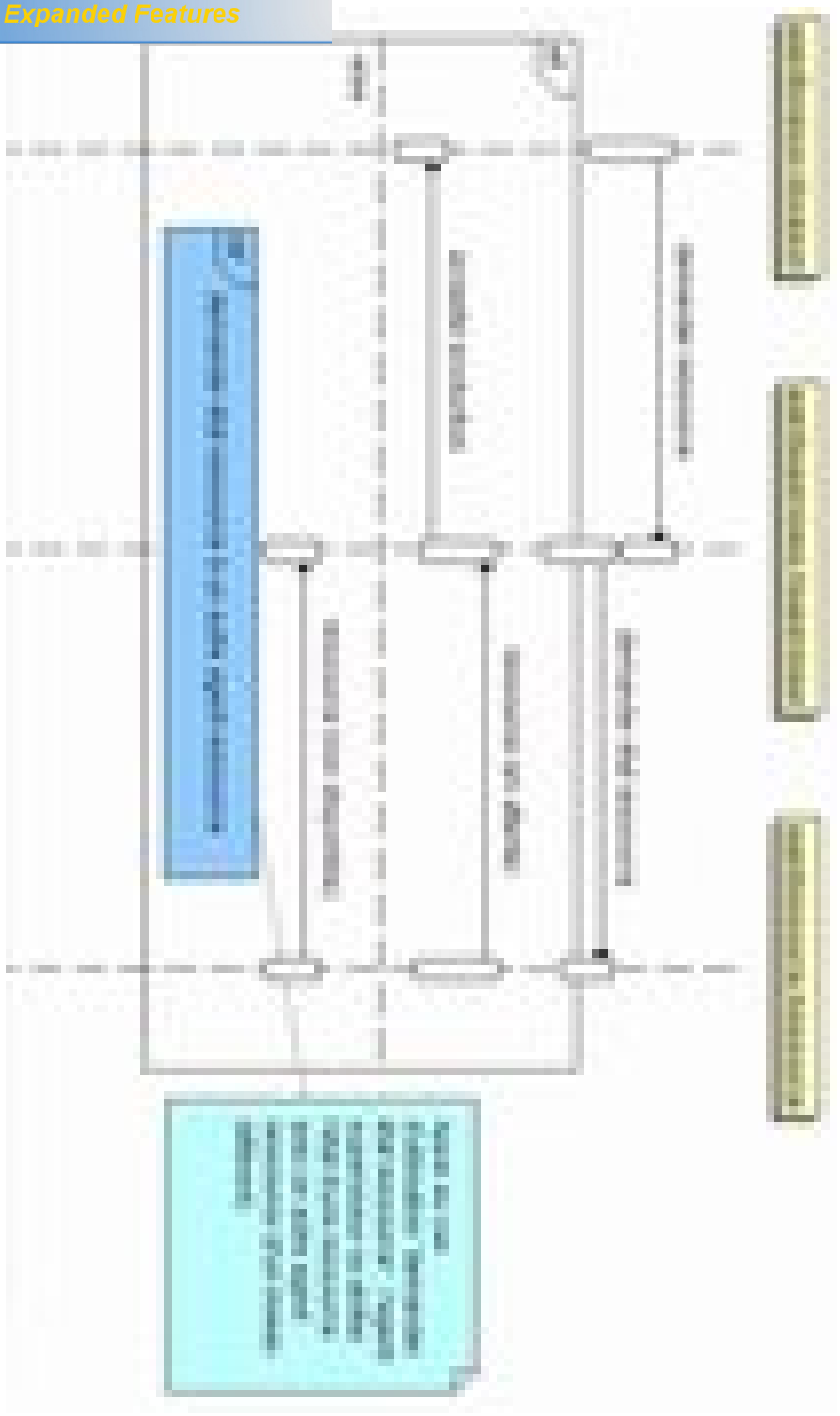

Figure 6.9 Diagramme de séquence du scénario 2 (panne dôune machine à un niveau *n* de lôarchitecture hétérarchique)

Au niveau de ce diagramme, nous remarquons lôinteraction entre lôagent livraison et lôagent superviseur pour pouvoir transférer la production qui a été interrompue à une autre ressource de niveau hiérarchique différent.

Enfin, pour conclure, nous présentons lôarchitecture globale de notre système avec la figure 6.10 :

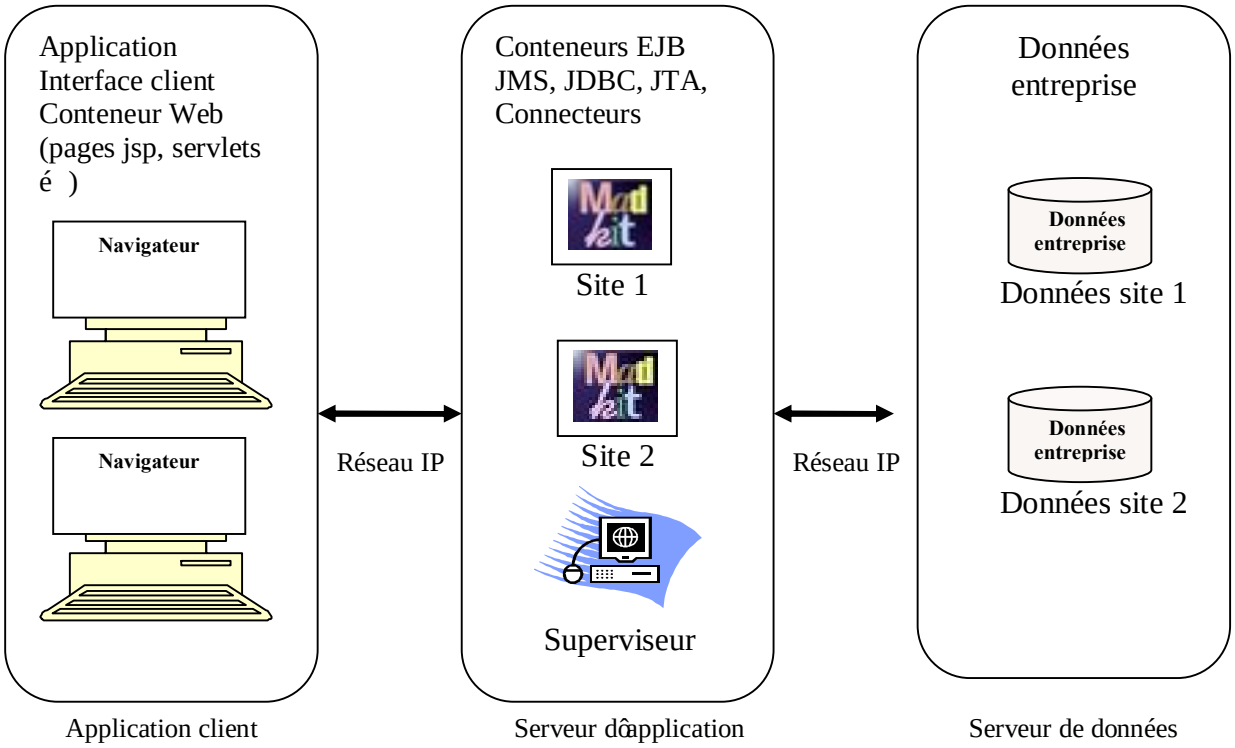

Figure 6.10 Le déploiement de notre application

Le modèle présenté va être déployé sur trois composants matériels :

- Serveur de données, pour contenir les données concernant lôentreprise telles que le nombre de sites, le nombres dôateliers dans chaque site, le nombre de machines, les lots à produire ainsi que les tâches à exécuter pour chaque lot;
- Serveur dôapplication contenant la plate forme Madkit;
- Poste client : au niveau des différents sites pour lôaffichage des pages web, à savoir les jsp et les pages html.

MIS EN PLACE

3.1 Introduction

Nous rappelons que lôalgorithme génétique que nous avons développé est implémenté au niveau de lôagent superviseur.

Généralement, un algorithme génétique simple, comporte quatre opérations fondamentales (voir chapitre 2) Lôorganigramme qui suit (figure 6.11) stipule les étapes suivies au sein de notre algorithme génétique.

Le but de lôalgorithme génétique dans notre système est de produire un plan de production optimal par rapport aux délais. Pour ce faire, nous formulons la fonction fitness en intégrant principalement le C_{max}. Côest par rapport à cette valeur que nous évaluons les individus constituant les différentes populations.

La valeur du fitness permet de faire évoluer la population (par la sélection é); nous cherchons ainsi à minimiser le retard de livraison.

Nous devons optimiser le temps de production au niveau de tous les sites pour avoir une livraison dans les meilleurs délais.
En dôautres termes, nous devons obtenir ceci:

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

La figure 6.11 montre lôalgorithme génétique mis au point afin dôoptimiser le temps de production au niveau de tous les sites de production (C_{max}).

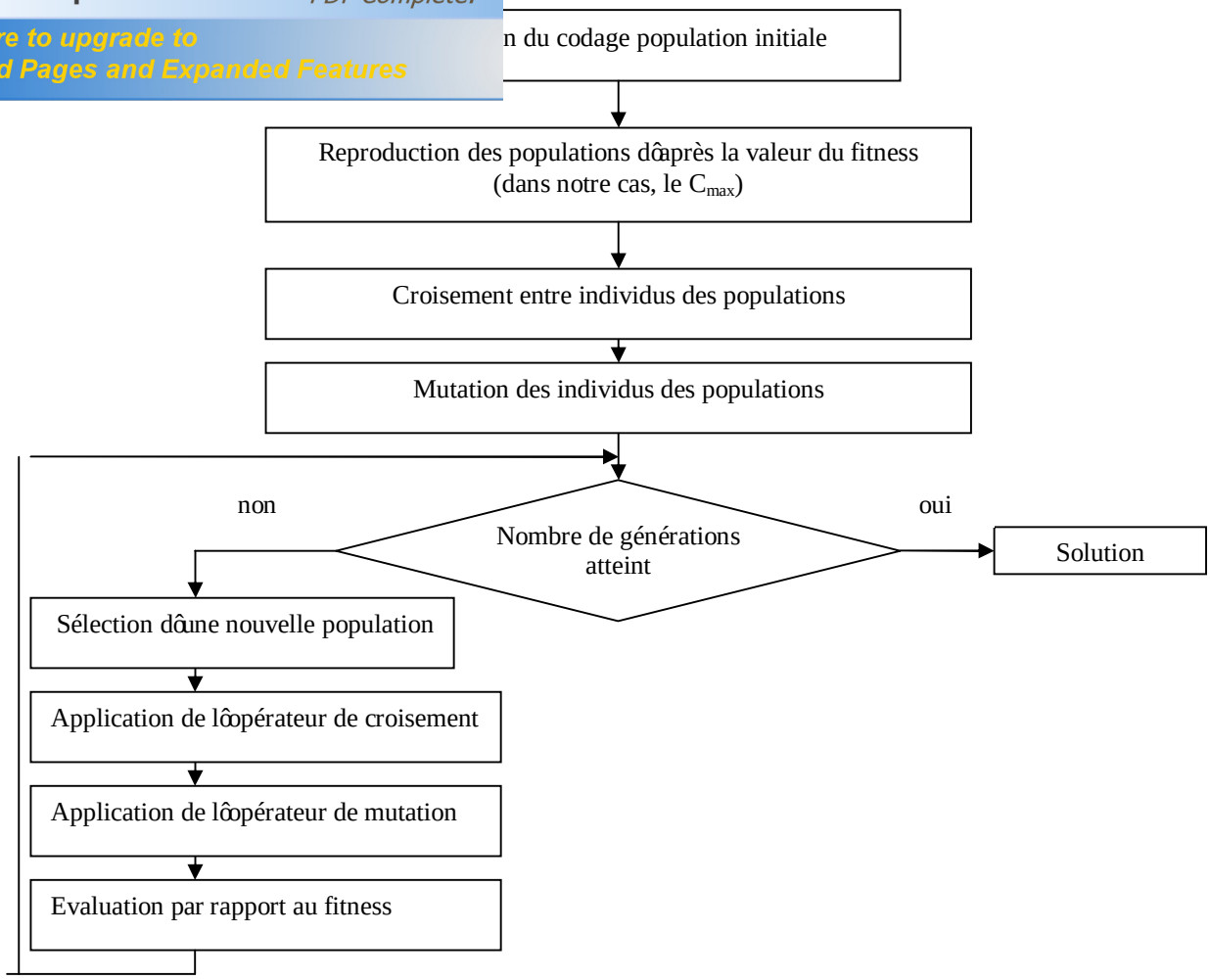

Figure 6.11 Représentation de notre algorithme génétique

3.2 Mise en place de l'algorithme génétique :

3.2.1 Le codage:

Nous passons à la modification de l'algorithme standard au niveau du codage : par rapport au génome humain, les caractéristiques d'un individu dans un algorithme génétique sont codées par une chaîne de caractères, appelée chromosome.

Traditionnellement, un individu est représenté par un chromosome dont les gènes sont des bits. Cette représentation n'est pas vraiment adaptée aux problèmes d'ordonnancement, car nous sommes en face dâune notion particulière : la tâche.

e modélisée simplement par un numéro [VACH 00]. Cette codification est inadéquate et insuffisante pour résoudre un problème tel que celui auquel nous faisons face. Elle sera ainsi caractérisée dans notre cas par plusieurs éléments : le numéro de la tâche, sa durée, la machine où elle peut s'exécuter, l'atelier et le site de son exécution ...

Un chromosome représente ainsi la succession des tâches correspondantes aux différents lots à traiter au niveau des sites (voir figure 6.16).

Nous avons mis en place une population initiale en utilisant des probabilités uniformément distribuées pour pouvoir profiter au maximum de la diversité des individus à traiter.

3.2.2 La reproduction :

La sélection est faite en utilisant le ***tournoi*** entre individus et l'élitisme, pour pouvoir obtenir le maximum de possibilités tout en gardant les individus les plus intéressants.

L'opérateur de ***croisement*** qui a été utilisé est le ***croisement multi points*** (voir chapitre 2) 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. Aussi, cet opérateur a été implémenté comme suit : Prendre deux gènes d'un certain chromosome de la population et les modifier ce qui nous permet d'obtenir un nouvel individu prêt à participer aux prochaines étapes de l'algorithme génétique (sélection, croisement, é).

Notons que la probabilité de mutation que nous avons utilisée est assez faible par rapport à celle utilisée pour le croisement. En effet, il faut avoir un nombre d'individus ou de chromosomes (parents) assez élevé pour pouvoir les croiser et en obtenir d'autres (enfants), alors que la mutation n'est appliquée que pour diversifier encore plus la population.

3.2.3 La sélection :

A la génération **N**, l'opérateur de sélection doit calculer la valeur du Fitness de chaque chromosome pour déterminer les meilleurs éléments. Les meilleurs individus dans notre cas

valeur du C_{max}. Comme a été indiqué plus haut, nous ise la fonction :

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

Notons que la condition terminale à laquelle nous nous sommes tenus après plusieurs expérimentations sur plusieurs critères (la taille du problème : nombre de lots à traiter, é) est le nombre de générations à produire au sein de notre algorithme.

Nous passons dans la section suivante à la réalisation de cette maquette modélisée.

4. REALISATION ET IMPLEMENTATION

Notre modèle a été simulé dans lœnvironnement *Borland JBuilder* vu son potentiel à faciliter la communication et la programmation de threads; il offre aussi la plate forme J2EE (Java 2 Enterprise Edition) (**http://java.sun.com/j2ee/overview.html**) qui permet le développement dœapplications multi tiers. Aussi, java est compatible avec lœarchitecture de la plate forme choisie MADKIT pour le développement de SMA (visiter http:// www.madkit.org/ downloads). Nous avons aussi utilisé la plate forme J2EE qui regroupe plusieurs bibliothèques (servlets/ EJB/ JSP) et une plate-forme logicielle pour les applications multi- tiers (voir chapitre 4). Nous détaillerons dans les annexes ces différents outils.

4.1.4 Développement de l'outil de simulation:

La réalisation du modèle précédemment présenté comporte trois volets:

1- La réalisation des JSPs (Java Server Page) qui sont des interfaces web contenant du html et du code java, et peuvent lancer et collecter les données de chaque site. (Figures 6.12 et 6.13).

Les trois prochaines figures montrent les premières pages jsp de notre maquette dont le rôle est la configuration des sites de production de lœentreprise:

Figure 6.12 Configuration des sites

A partir de cette page jsp, nous obtenons la deuxième page jsp représentée dans la figure suivante:

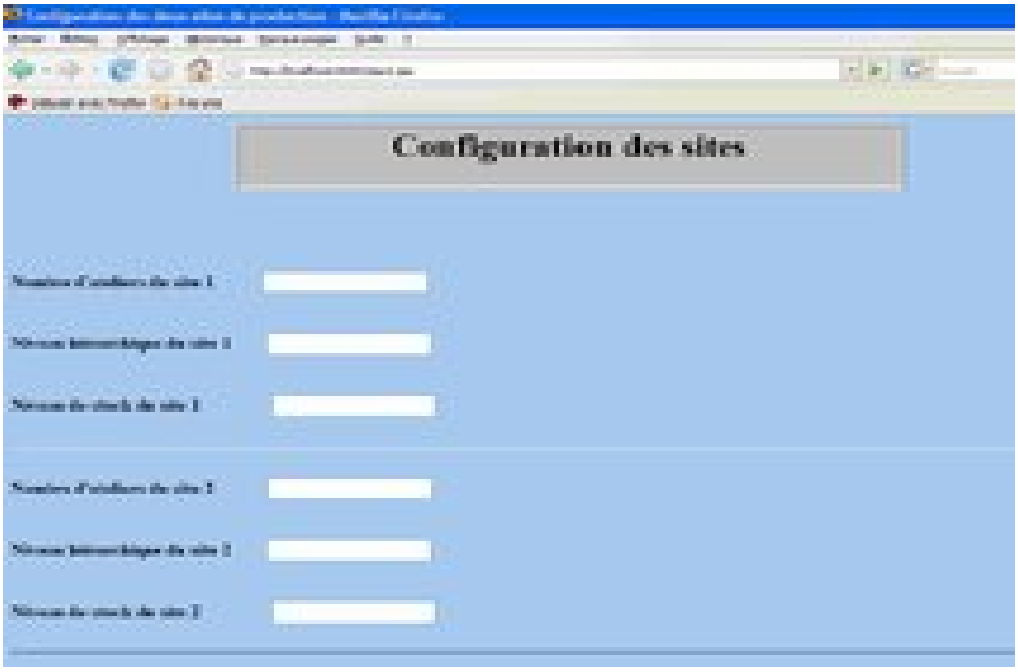

Figure 6.13 Configuration des sites de production

Figure 6.14 Configuration des ateliers de production de chaque site

2- La réalisation des EJBs (Enterprise Java Bean), qui constituent un module déployé au niveau du serveur d'application ; leur rôle est de codifier l'information, lancer le noyau Madkit ainsi que les agents correspondants aux sites.

Dans notre cas, les EJB permettent de passer d'une page jsp à une autre, ainsi que de lancer le noyau de madkit et permettre la creation et le lancement des agents de notre système.

Après la configuration de tous les ateliers appartenant aux sites de production, le lancement de l'algorithme génétique se fait par l'agent superviseur.

3- Le développement des agents, leurs modules de communication, leur module de prise de décision (decision making) et le module d'apprentissage de l'agent superviseur.

4.2 Expérimentation et investigation :

Dans notre architecture (figures 6.13 et 6.14), le **serveur de données** contient les données de l'entreprise: le nombre de sites, d'ateliers, de machines, le niveau hiérarchique de chaque site et les quantités dans les stocks des sites.

L'algorithme génétique implémenté demande aussi comme entrées d'autres informations supplémentaires et indispensables à son fonctionnement. La figure 6.15 illustre quelques unes de ces informations concernant les tâches de fabrication en particulier.

Figure 6.15 Configuration des différents lots

Nous supposons que la production est en continu, nous saisissons le nombre de lots, le nombre de tâches associées à chaque lot respectivement, en saisissant bien sûr les données relatives aux tâches à savoir, les sites où une tâche peut sœexécuter, les ateliers, les machines, les durées dœexécution des tâches ....

La figure suivante montre une partie de la configuration des lots à produire, ces différentes lignes vont être codifiées pour pouvoir être utilisées dans lœAG qui donnera un résultat traduit en diagramme de Gantt.

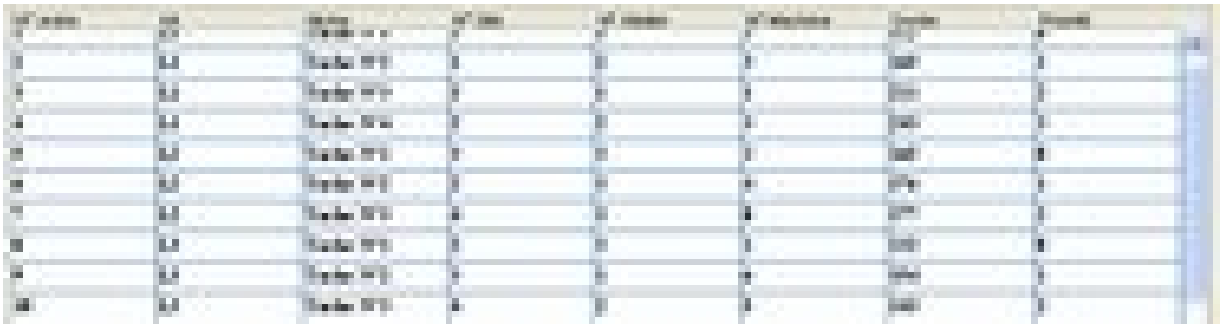

Figure 6.16 Tableau récapitulatif des tâches

4.2.1. L'ordonnancement préventif (Planification)

Après avoir obtenu toutes ces informations, notre algorithme génétique est lancé par l'agent superviseur et donne un résultat de planification correspondant à une valeur de fitness la plus optimale possible, c'est-à-dire la durée d'exécution totale des différentes tâches des lots.

La figure suivante montre une partie d'une population obtenue après plusieurs générations, chaque ligne correspond à un individu (ou solution); un des individus (ligne) sera sélectionné pour être représenté par le diagramme de Gantt (figure 6.18).

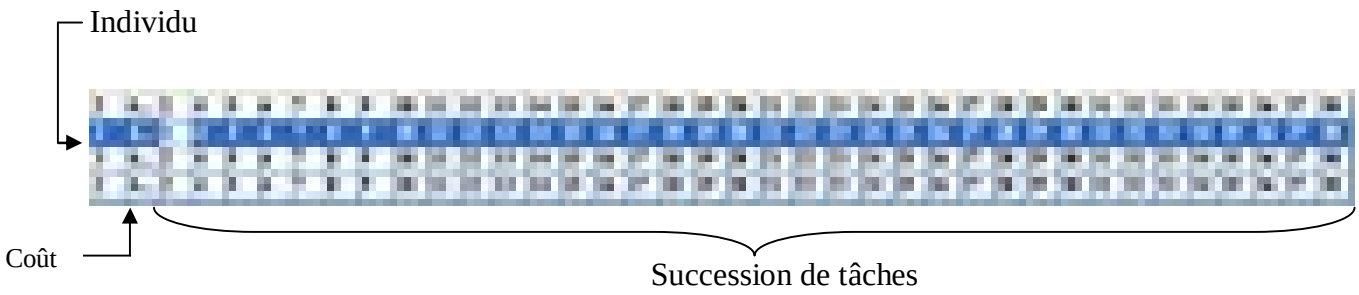

Figure 6.17 Population et individus

Le résultat est représenté sous forme de diagramme de Gantt comme le montre la figure suivante.

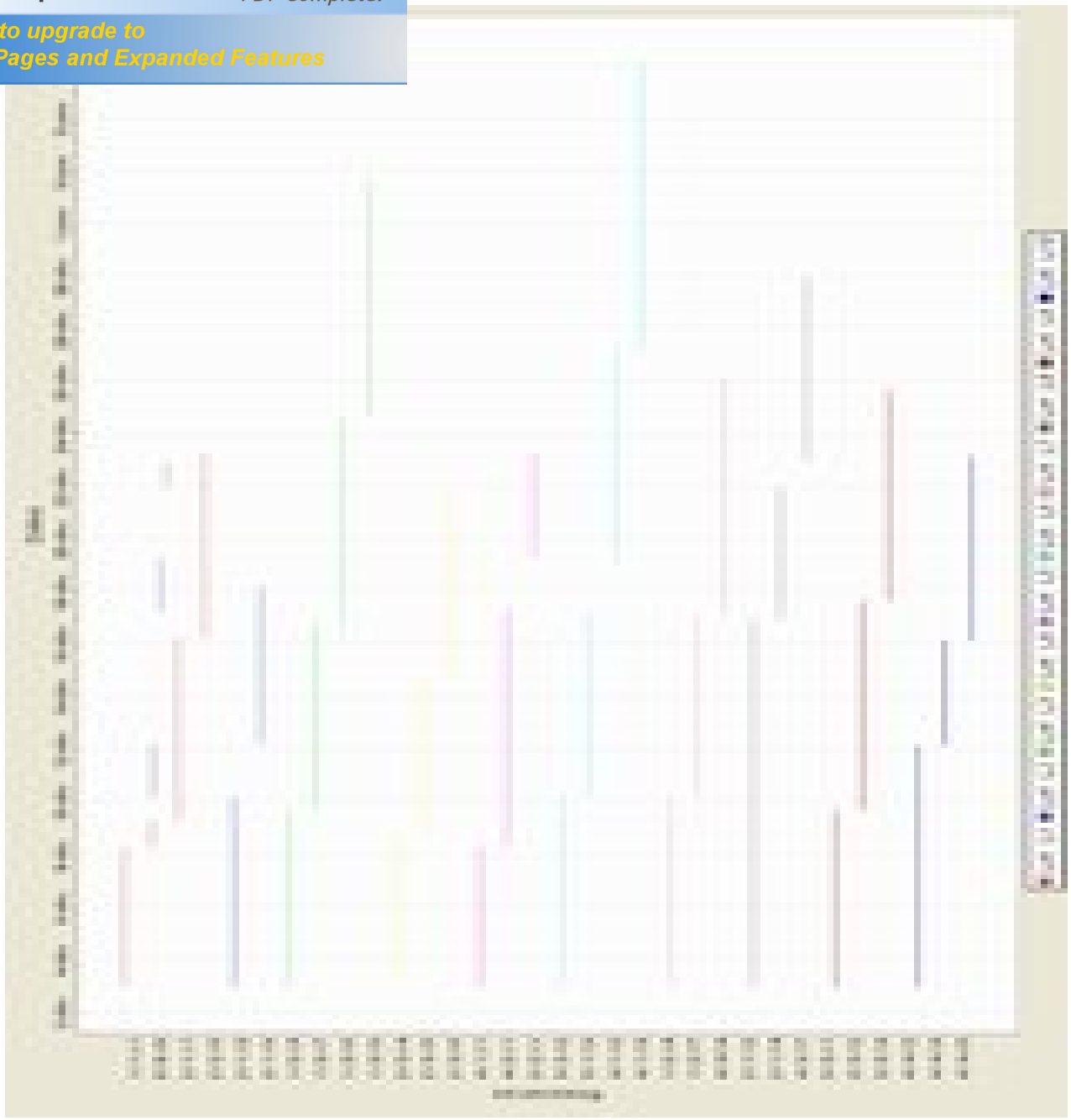

Figure 6.18 Diagramme de Gantt obtenu après application de notre approche

Notons que la notation au niveau vertical: 0/0/0 correspond au transport, en effet, lôalgorithme génétique prend en compte le fait de transporter des produits entre les différents sites pour pouvoir y effectuer différentes tâches.

Le logiciel permet aussi de visualiser les différents diagrammes de Gantt partiels correspondants aux sites de production de lôentreprise. Pour notre exemple, nous obtenons les quatre diagrammes de Gantt:

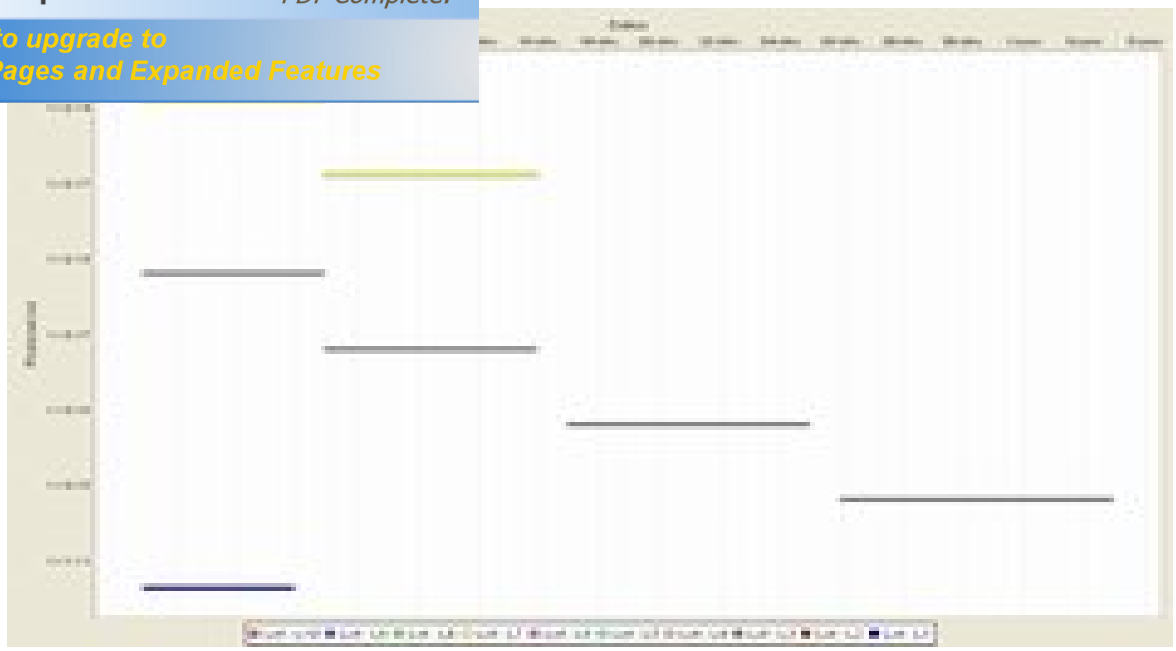

Figure 6.19 Diagramme de Gantt associé au site 1

Figure 6.20 Diagramme de Gantt associé au site 2

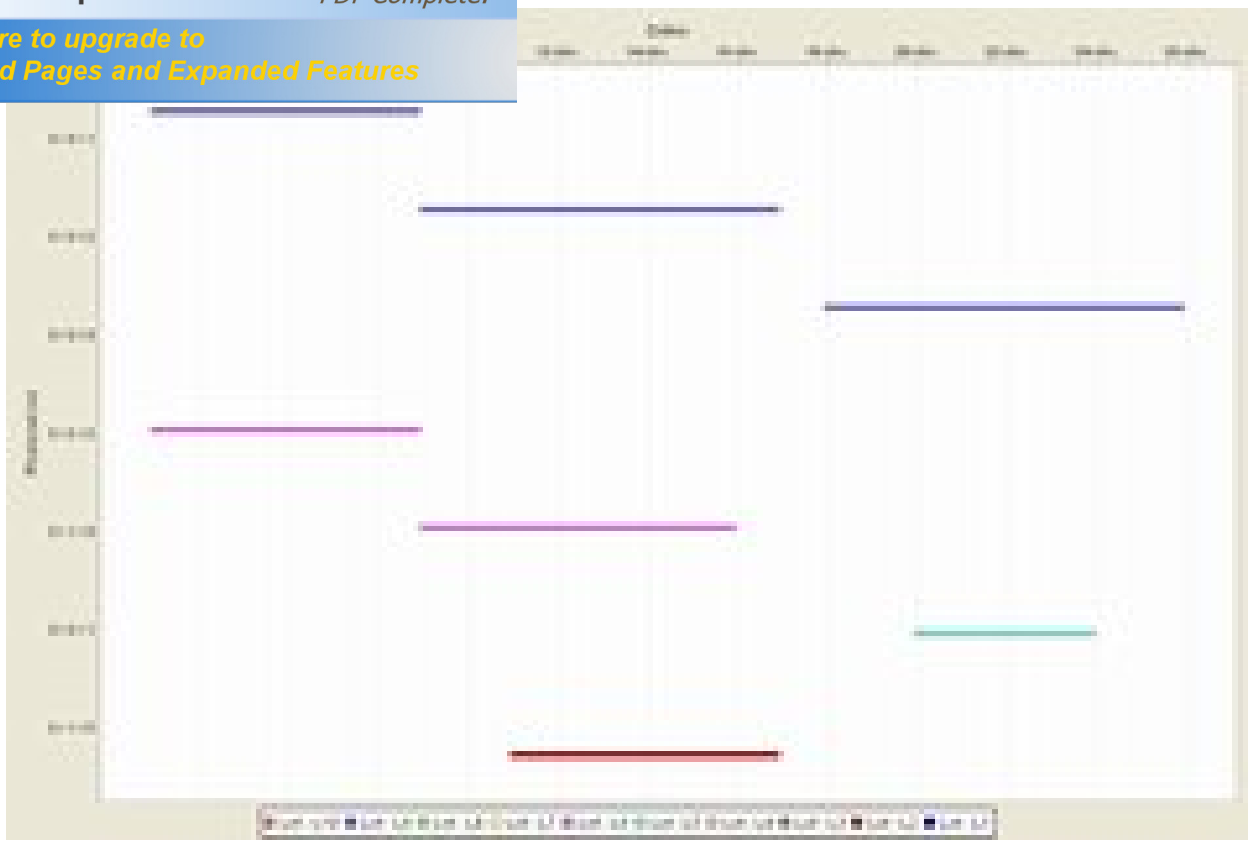

Figure 6.21 Diagramme de Gantt associé au site 3

Figure 6.22 Diagramme de Gantt associé au site 4

énétique :

En ce qui concerne lôalgorithme génétique implémenté, nous avons choisi comme critère dôévaluation, son temps dôexécution. En moyenne, les résultats sont bons pour les petits fichiers, cela dépend des critères dôarrêt choisis, car, nous le rappelons, les algorithmes génétiques sont des méthodes pseudo aléatoires. Dans notre cas, nous avons choisi le nombre de générations.

Les algorithmes génétiques permettent d'explorer une surface de l'espace des solutions (plusieurs individus par génération) alors que les autres méthodes n'explorent qu'un point à la fois (un individu par itération).

Cela permet alors de déterminer une bonne solution (qui peut être optimale si nous avons un moyen de la connaître par une méthode exhaustive; ce point peut constituer une perspective de recherche pouvant être la suite de notre travail) et ce, plus rapidement et avec plus de sûreté que par les autres méthodes.

LôAG a été lancé à partir dôun poste avec les capacités suivantes : un processeur de capacité de 1.73 GHz, et 512 Mo de RAM.

Le tableau suivant montre les résultats obtenus par application de lôAG dans le cas de cinq sites de production:

Nombre de Sites	Ateliers	Nombre de lots	Nombres de tâches	Temps dôexécution
5	3	1	1	15
5	3	1	2	10
5	3	1	3	12
5	3	2	5	12
5	3	2	6	13
5	3	3	5	8
5	3	3	6	6
5	3	3	7	8

Tableau 6.1 Temps dôexécution de lôalgorithme génétique pour 1,2 et 3 lots selon différents nombres de tâches par lot

A partir de ce tableau, nous remarquons que la valeur du temps dôexécution de lôAG est plus petite en ce qui concerne la production de 3 lots, ce qui encourage à utiliser cet AG pour un plus grand nombre de lots.

tats concernant lœexécution de lœalgorithme génétique
ons ont été faites par rapport au nombre de tâches par lot:

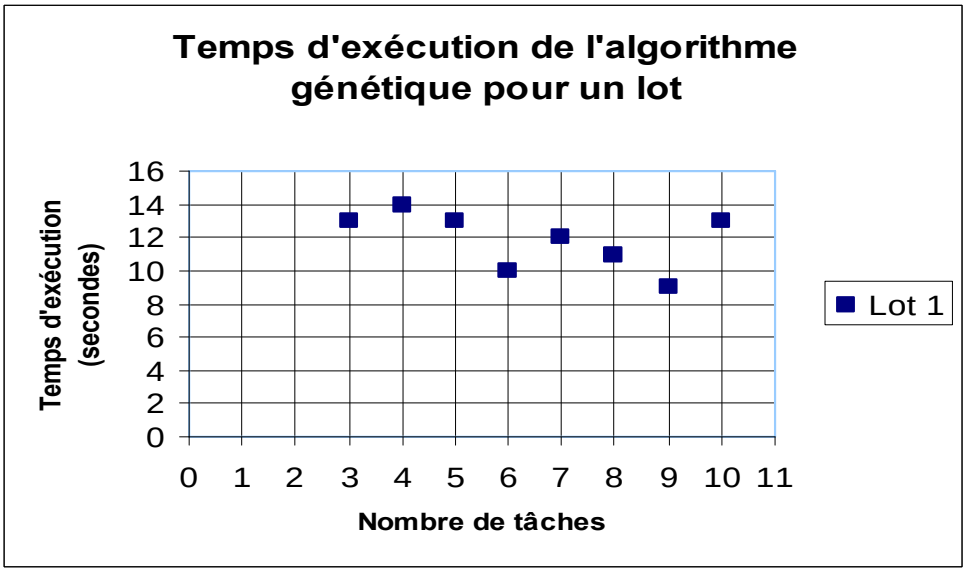

Figure 6.23 Temps dœexécution de lœAG pour un seul lot

La figure suivante montre la durée dœexécution de lœalgorithme génétique pour lœordonnancement de 2 lots à produire suivant 3 configurations différentes (variation du nombre de tâches par lot):

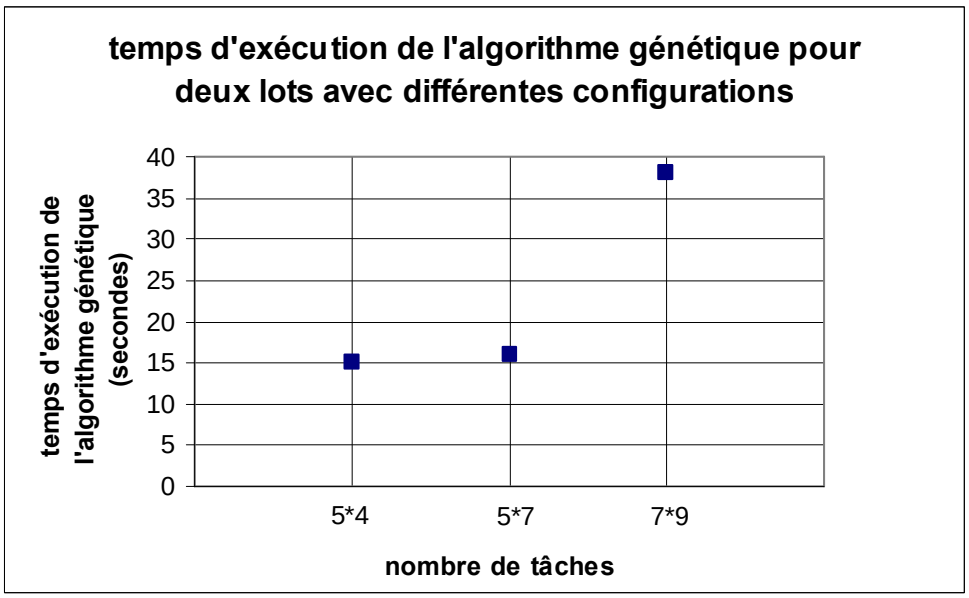

Figure 6.24 Temps dœexécution de lœAG pour différentes configurations de 2 lots

écution de lôAG pour trois, cinq et dix lots avec différents nombres de tâches par lot.

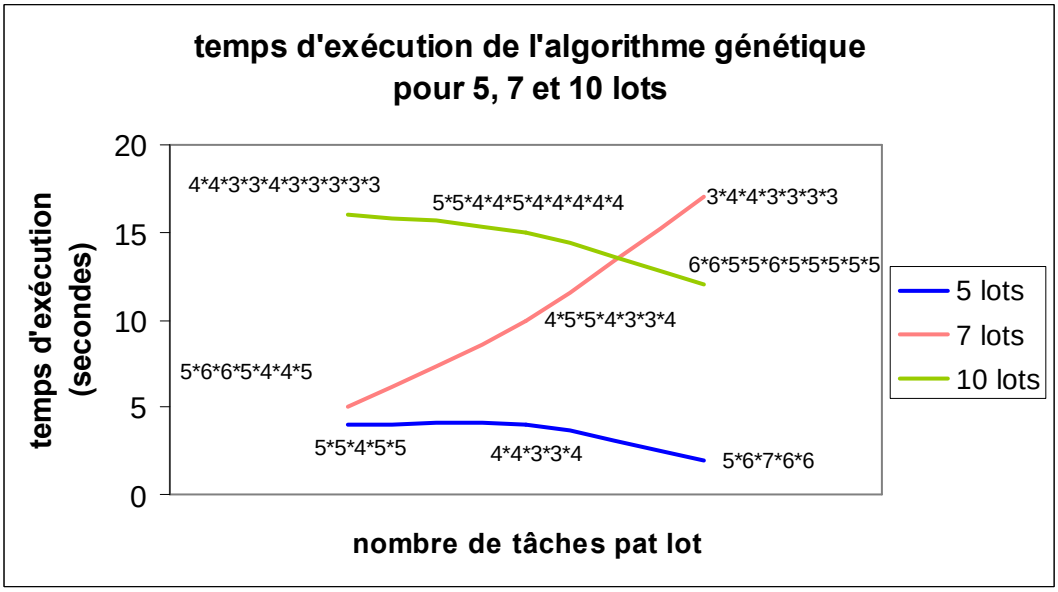

Figure 6.25 Temps dôexécution de lôAG pour 5,7 et 10 lots à produire

Au niveau de cette figure, nous remarquons que pour lôexemple de 10 lots et pour celui de 5 lots, malgré la variation du nombre de tâches de chaque lot, le temps dôexécution de lôalgorithme génétique est presque stable, alors que pour lôexemple de 7 lots, le temps dôexécution de lôAG augmente. Ceci peut sôexpliquer par la présence de lôaspect pseudo aléatoire de lôalgorithme. Nous pouvons déduire de cette représentation que lôAG développé peut être utilisé pour des fichiers de grande envergure comme pour des fichiers de petite taille sans affecter son efficacité et en mettant un temps dôexécution assez satisfaisant ce qui représente sa force.

4.2.3. La réactivité du système

Pour la gestion des aléas, nous avons mis en place un ***module perturbation*** utilisant le SMA développé.

Nous précision que nous avons pris en charge deux types de perturbation:

- La panne dôune machine appartenant à un atelier dôun des sites de production ;
- La rupture de stock au niveau dôun des sites.

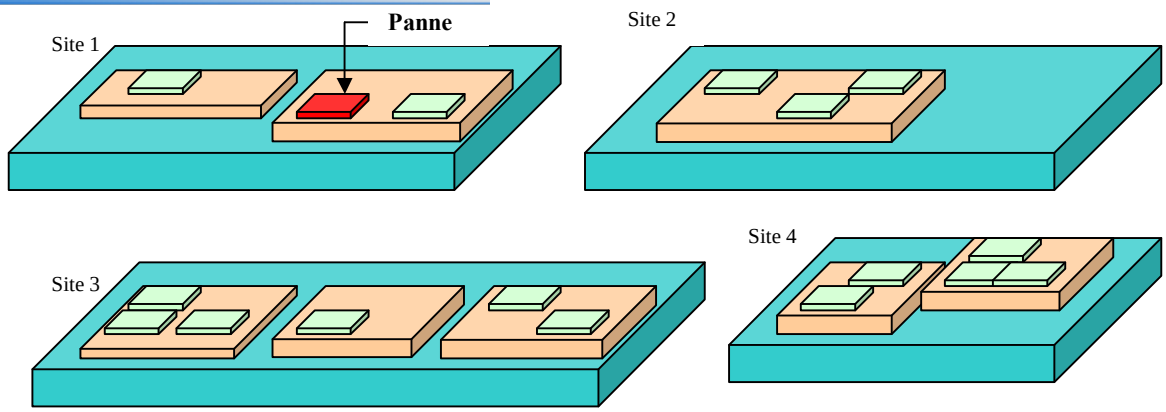

Figure 6.26 Panne au niveau du site1/ atelier1/ machine1

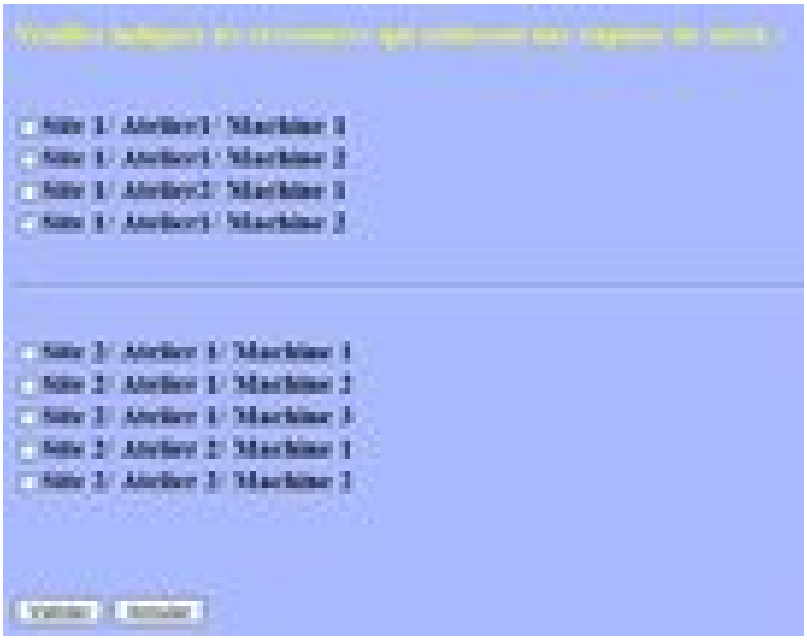

Figure 6.27 Le module de perturbation (rupture de stock)

Pour mieux expliciter le rôle du SMA, nous présentons la figure **6.27** qui englobe les différentes actions quœentreprennent les agents. Cette figure représente lôagent observateur.

Lôagent observateur est un agent propre à la plate-forme Madkit (et réutilisable par les développeurs de SMA sous Madkit), il a pour rôle de tracer les messages qui ont lieu entre les agents et de permettre la visualisation de lôinteraction inter- agent au niveau du noyau Madkit auquel il appartient:

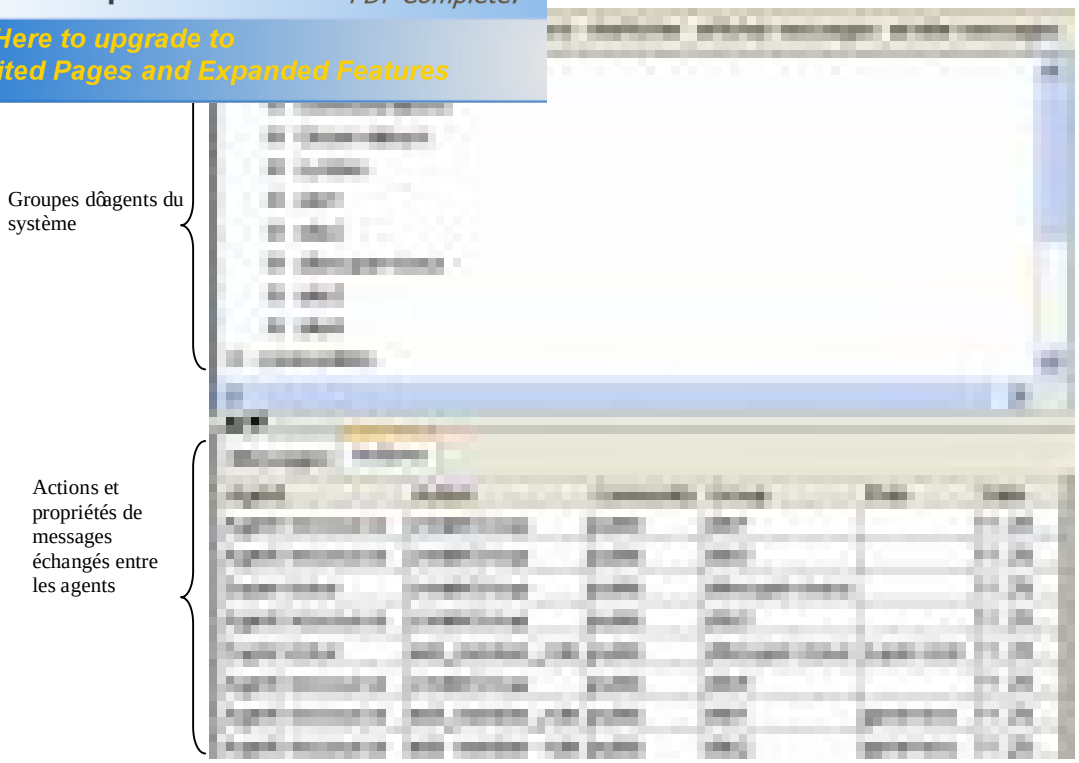

Figure 6.28 Lâinterface de suivi du SMA

La figure suivante montre les messages échangés entre les agents.

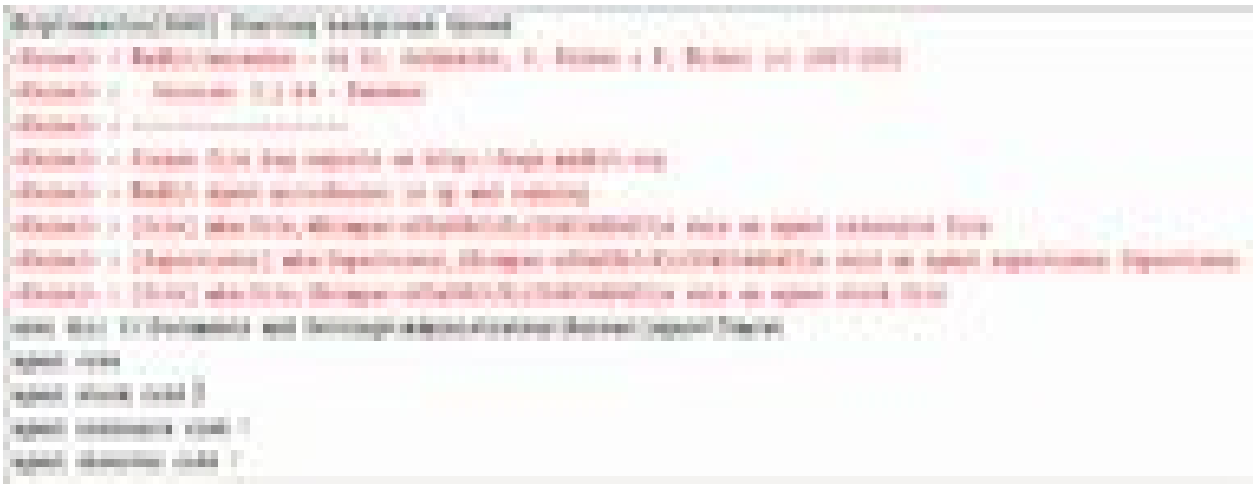

Figure 6.29 Suivi de lâenvoi de messages entre les agents

Dans le cas de la panne dûne ressource sur laquelle sœexécute une tâche, sa durée restante sera recalculée, si la panne est provoquée à un instant *t.* Si sa durée prévue sur cette machine était évaluée à x unités de temps, le reste est évalué selon la formule :

```
Reste = x- [(t*100) /x]
```

Ce pourcentage est calculé pour pouvoir évaluer sa durée dœexécution restante au cas où elle serait affectée à une autre ressource de production.

Après interaction entre les agents, nous obtenons un résultat, ce dernier est représenté dans le diagramme de Gantt suivant (figure 6.30)

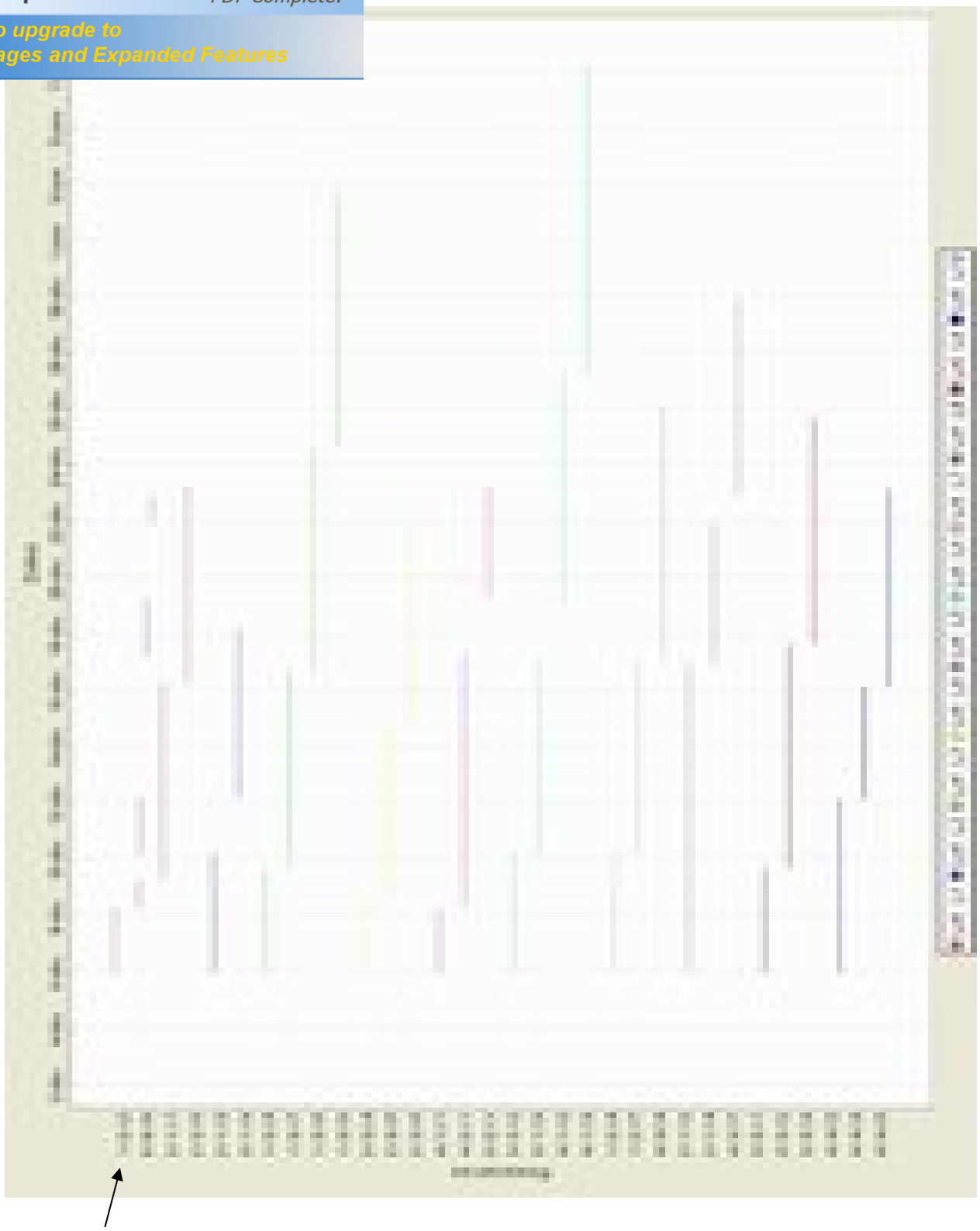

Figure 6.30 Nouveau diagramme de Gantt après traitement de perturbation

2.Rupture de stock :

Dans ce cas, les agents interagissent entre eux pour pouvoir arriver à la meilleure solution, lôinterface de lôagent observateur ne permet pas de voir le contenu des messages envoyés en détail, cela dit, nous avons pris en compte ce détail dans notre système: la figure suivante montre lôenvoi de messages entre les agents.

Figure 6.31 Envoi de messages après détection de rupture de stock

Après communication entre les agents des différents groupes (dont lôagent stock du site concerné par la rupture de stock qui joue le rôle principal dans ce cas), nous obtenons une solution (concernant la rupture de stock) qui sera appliquée au système.

Figure 6.32 Solution proposée après détection de rupture de stock

Nous précisons que, d'un côté, l'architecture hétérarchique nous a été d'une grande utilité du point de vu temporel, car la communication entre les entités (agents) d'un même niveau permet un gain de temps considérable. Par exemple, une communication entre deux entité de même niveau a pu résoudre un problème de rupture de stock immédiatement, alors que si nous avions utilisé une approche purement hiérarchique, il aurait fallu communiquer avec des entités de plus haut niveau hiérarchique, ce qui aurait bien entendu pris plus de temps pour arriver à une solution.

D'un autre côté, l'utilisation de l'agent observateur nous a permis de garder et d'observer l'évolution de la performance globale du système grâce à ses facultés de suivi.

Ainsi, l'application de l'AG et du SMA a permis d'atteindre le but recherché: la planification ou l'ordonnancement.

En effet, après communication entre les différents agents des différents groupes, nous obtenons une solution prête à être appliquée sur le système pour permettre au système de fonctionner au mieux.

6. CONCLUSION

Dans ce chapitre, nous avons présenté notre modélisation à travers les différents diagrammes et schémas montrant les entités (en particulier les agents des différents groupes) qui interagissent entre elles afin d'aboutir à un ordonnancement optimal dans deux optiques : une planification à long terme (ordonnancement préventif) et une allocation des ressources dynamiques (ordonnancement réactif).

Pour ce faire, nous avons utilisé d'une part le système multi agents incluant les différents groupes et, d'autre part, l'algorithme génétique implémenté au sein de l'agent superviseur.

La représentation du résultat est obtenue essentiellement sous forme de diagramme de Gantt explicitant la planification et l'ordonnancement de la production.

ALE :

Actuellement, les chercheurs du domaine industriel font face à des environnements très dynamiques et évolutifs.

Un grand nombre de recherches est réalisé en utilisant les SMAs vu les propriétés prometteuses quôils offrent (voir chapitre 3) dans le domaine industriel.
Les résultats les plus connus sont en effet issus de la communauté multi agents et celle de lôintelligence artificielle mais parmi ceux-ci, très peu sôattachent aux problématiques de la gestion de la production dans un contexte **court, moyen et long terme** au niveau des entreprises de production **multi sites**.

Notre présent travail a été alors la réalisation dôun système pour la planification et lôordonnancement de la production au niveau des entreprises multi sites qui soit le plus performant possible en utilisant une technique dôapprentissage afin de faire évoluer les performances globales dôun système dôune telle organisation. Ce système sôappuie alors sur une approche hybride **préventive- réactive**.

Dans ce mémoire, nous avons apporté une clarification sur la problématique complexe à laquelle nous nous sommes attaqués: lôactivité de planification et dôordonnancement de la production au sein dôentreprises multi sites à court, à moyen et à long terme. Nous avons présenté un état de lôart de la planification multi sites en précisant le manque de planification en court, moyen et long terme au niveau des modèles déjà présentés dans la littérature.

Notre travail a donc été, le développement dôune démarche de modélisation basée sur les SMA dotée de capacités décisionnelles dans une organisation hétérarchique par apprentissage, pour pouvoir assurer une planification à long, moyen et court terme.

En effet, nous précisons que notre objectif était dôélaborer un système qui soit le plus fiable possible dans le respect de la globalité de la performance du système étudié (entreprise multi sites). Pour ce faire, nous avons utilisé une modélisation multi agents de telle façon que les décisions prises par le système soient le résultat dôun travail de groupes dôagents. Ce SMA a été utilisé au sein dôune organisation dont lôarchitecture est hétérarchique, cette architecture

ommunication entre les entités dôun même niveau

Quant à la planification (à long terme) de la production, elle suit le résultat initial donné par lôalgorithme génétique que nous avons développé.

Nous avons alors pu introduire la notion dôapprentissage au niveau du concept dôagent et de rôles dôagents (Planifier, é) en utilisant les algorithmes génétiques, une technique qui a fait ses preuves dans le domaine de lôordonnancement. Ces méthodes ont beaucoup apporté à notre travail; en effet, lôalgorithme génétique que nous avons développé, a permis aux activités de production du système dôêtre planifiées à long terme au sein dôun agent appelé « agent superviseur » qui, par une interaction avec les groupes dôagents et en utilisant lôalgorithme génétique proposé spécifique aux entreprises multi sites, a la responsabilité de gérer et dôimposer ses décisions de production au niveau des différents sites de lôentreprise.

Le but que nous avons pu atteindre a été alors de répondre aux changements et aux aléas que subissent les différents sites du système de production et dôassurer son bon fonctionnement.

Nous avons aussi présenté les moyens que nous nous sommes donnés pour réaliser le modèle élaboré. Les résultats sont concluants, notre modèle a pu répondre à nos attentes citées ci-dessus en un temps raisonnable par rapport aux exigences imposées.

Enfin, nous avons présenté la contribution que nous avons mise en î uvre dans ce projet: lôorganisation du SMA dans la modélisation de notre système et lôintroduction de lôapprentissage au moyen des AGs.

2 PERSPECTIVES :

Nous avons pu constater que de nombreuses perspectives étaient possibles à travers ce travail.

En effet, quelques améliorations peuvent être apportées à ce modèle: il serait intéressant dôutiliser la notion de négociation au sein de notre système multi agents.

de dôoptimisation (algorithmes génétiques) parait être timisation comme les méthodes exactes telles que la méthode de Branch and Bound, ...

Annexes

Annexe A

BOUML

BOUML est une série de logiciels comprenant un **modeleur UML2** et plusieurs programmes externes dont des **générateurs de code** et **reverse**.

Le modeleur BOUML est distribué sous licence GPL, il peut alors être librement utilisé pour modéliser et produire du code. Notons quôil est développé en C++ au dessus de Qt (bibliothèque logicielle orientée objet) ce qui offre la possibilité dôutilisation sous plusieurs systèmes dœexploitation, notamment Windows et Unix (Linux).

Le but de BOUML est de permettre une utilisation allant des besoins à la génération de code. Parmi les langages pris en compte à ce jour, nous avons C++, Java, Php.

Une des raisons qui nous ont poussé à son utilisation est quôil est peu gourmand en ressources CPU et en mémoire et qu'il peut être utilisé dans un contexte multi- utilisateurs avec gestion de configuration.

BOUML est extensible via l'écriture de plug- outs qui sont des programmes permettant d'accéder et/ou de modifier automatiquement des modèles, ils correspondent aux scripts de

en utilisant un vrai langage de programmation, à savoir
dehors du modeleur, ce qui permet d'arrêter leur exécution sans perdre le modèle.

Fonctionnalités :

Parmi les principales fonctionnalités de BOUML (qui représentent une motivation pour son utilisation), nous citons:

* Bouml est libre ;
* Bouml peut être utilisé sous Linux/Unix/Solaris, Mac OS X et Windows grâce à l'utilisation de Qt;
* Bouml permet de programmer simultanément en C++, Java, Php, Python et IDL;
* grâce à un accès complet aux formes générées, lâutilisateur décide ce qui doit être généré;
* Bouml est extensible, les programmes d'extension peuvent être développés en C++ ou Java, en utilisant BOUML comme pour n'importe quel autre programme;
* Bouml est très rapide et nécessite peu de mémoire pour modéliser plusieurs milliers de classes.

Annexe B

MADKIT:

Une plate-forme de développement de systèmes multi agents

Madkit [Net 2] est une plate-forme de développement légère permettant de faire du développement de SMA de manière assez simple. Elle a été développée en java au sein du laboratoire LIRMN.

Lôoriginalité de cette plateforme est quôelle est basée sur un modèle organisationnel plutôt que sur une architecture dôagents ou un modèle dôinteraction spécifique.

MadKit est alors destinée au développement et à lôexécution de SMA et plus particulièrement à des SMA fondés sur des critères organisationnels (groupes et rôles). [FERB 08]

Cette Plateforme est implémentée selon la méthodologie Aalaadin [FERB 98], basée sur les notions dôagent, de groupe et de rôle, ce qui nous facilite la gestion du SMA (voir chapitre 3).

Un de ses points forts est quôelle nôimpose aucune architecture particulière aux agents. Il est ainsi possible de développer aussi bien des applications avec des agents réactifs (comme le fait TurtleKit) que des agents cognitifs et communicationnels, et même de faire interagir facilement tous ces types dôagents.

Cela permet ainsi aux développeurs dôimplémenter lôarchitecture de leur choix sans aucune contrainte liée à lôarchitecture.

e en mode distribué de manière transparente à partir écessiter de serveur dédié. Il est ainsi possible de faire communiquer des agents à distance sans avoir à se préoccuper des problèmes de communication qui sont gérés par la plate-forme.

MadKit est un logiciel libre de type ñOpen Sourceò avec une licence mixte GPL/LGPL. LGPL pour le micro- noyau et les outils de communication, et GPL pour les outils de développement.

On peut ainsi facilement développer à lôaide de MadKit et utiliser par la suite les agents construits dans des applications commerciales.

Madkit facilite son intégration à dôautres plates formes JAVA car une application développée avec Madkit peut sôexécuter en mode distribué sans quôil y ait besoin de modifier le code.

Aussi, lôensemble de la plate forme Madkit a été réalisé dans le langage Java. Nous avons alors réalisé lôensemble de nos développements dans ce langage.

Le mécanisme de distribution est indépendant de MadKit et il est possible dôen créer de nouveaux si les besoins sôen font sentir ou même dôutiliser des plate- formes distribuées existantes. Dans notre cas, nous utilisons la plate forme J2EE [Net 3].

Une des forces de cette plate forme est son aspect graphique. La figure B.1 présente le Desktop de Madkit :

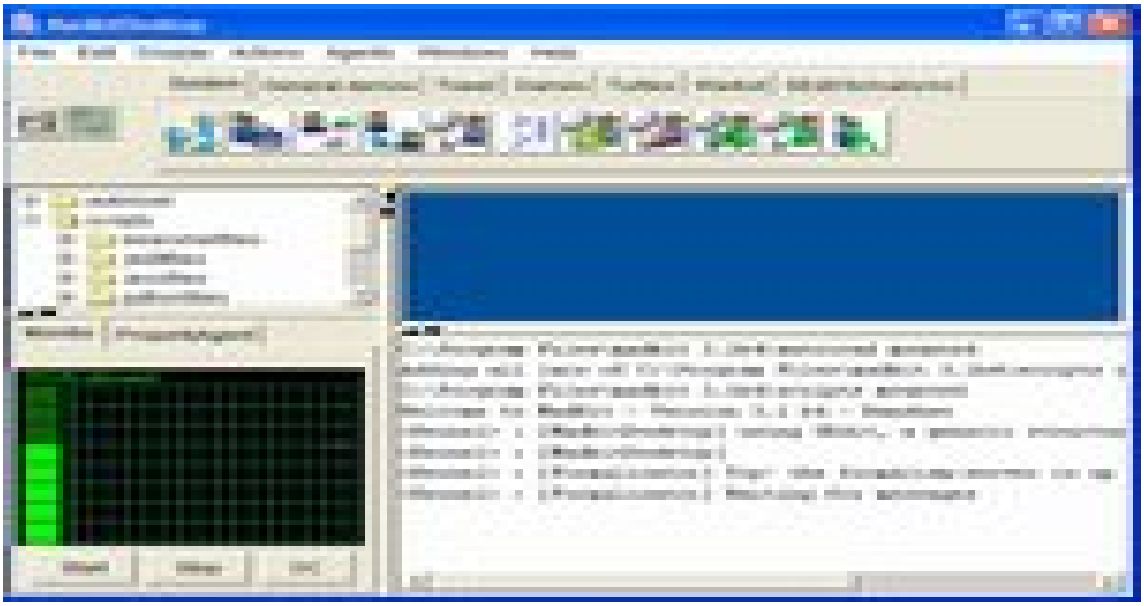

Figure B.1 Desktop de Madkit

Nous allons définir dans ce qui suit le modèle organisationnel Aalaadin présenté dans [FERB 98] et [FERB 02]

Le modèle **Aalaadin** est basé sur trois concepts : lô*agent*, le *groupe* et le *rôle*. La figure B.2 présente un diagramme de ce modèle.

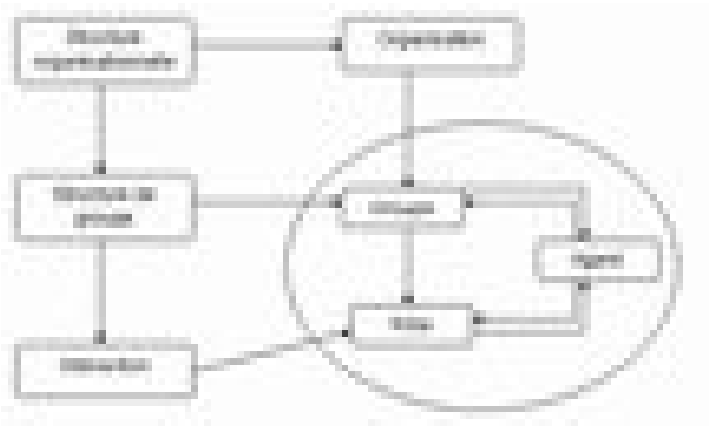

Figure B.2 Le modèle Aalaadin

La figure B.3 représente les trois notions fondamentales du modèle Aalaadin: lôagent, le groupe et le rôle :

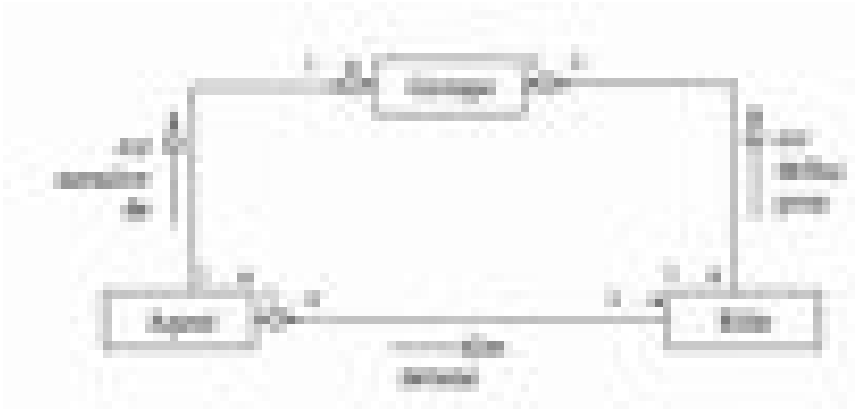

Figure B.3 Agent/groupe/rôle

5.1 L'agent :

Dans ce modèle, aucune contrainte ou pré- requis sur lôarchitecture interne de lôagent nôest posé, ni sur le formalisme ni sur un modèle particulier à utiliser pour en décrire le

nt défini comme une entité autonome communicante
groupes.

Notons que cette définition est intentionnellement générale pour permettre à chaque concepteur dôagents 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.

5.2 Le groupe :

Le groupe est défini dans Aalaadin, comme la notion primitive de regroupement dôagents. Chaque agent peut être membre dôun ou de plusieurs groupes.

Dans sa forme la plus simple, un groupe peut être vu comme un moyen dôidentifier par regroupement un ensemble dôagents, dôune manière plus évoluée, le groupe peut être vu comme un SMA usuel. Un groupe peut être fondé par nôimporte quel agent.

5.3 Le rôle :

Le rôle est la représentation abstraite dôune fonction, dôun service ou dôune identification dôagent au sein dôun 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 lôhétérogénéité des situations dôinteraction est rendue possible par le fait quôun agent peut avoir plusieurs rôles distincts au sein de plusieurs groupes, et que les interactions sont toujours locales à un groupe.

Ainsi, un agent ***a*** peut entrer dans un groupe ***g*** pour y jouer un rôle ***r*** si une fonction dôacceptation booléenne $f\{g\ r\ (a)\}$ est évaluée à vrai.

Voici quelques exemples possibles de fonctions dôadmission à un rôle:

- *Acceptation ou refus automatique*, où un agent voit ses demandes dôadmission systématiquement acceptées ou rejetées.
- *Admission conditionnée par les compétences.* Dans ce mécanisme, un rôle est confié à un agent seulement sôil possède un certain jeu de compétences.
- *Admission contrainte par l'architecture* où lôagent doit exhiber certains pré- requis sur sa structure interne pour se voir accepter dans un groupe.

evra entrer dans un dialogue initial avec lôagent ayant
e pour déterminer ses droits au rôle

Remarquons quôil existe un rôle important et particulier dans un groupe : le rôle de gestionnaire de groupe, confié initialement au fondateur. Ce rôle correspond à la responsabilité de gestion des demandes dôadmission dans le groupe et de tenue de rôle, ainsi que de leurs éventuelles révocations.

Les messages :

Les messages sont définis par héritage à partir dôune classe de base (message) qui ne définit que la notion dôémetteur et de récepteur. Certains messages de base sont aussi fournis (encapsulation dôune chaîne ou dôun objet arbitraire, messages FIPA ï ACL, messages XML) mais restent extensibles pour sôadapter à tout protocole dôintéraction.

Annexe C

JBUILDER :

Un environnement de développement intégré Java

JBuilder est un environnement de développement intégré Java qui, associé à un serveur dôapplications de sociétés comme Borland, BEA, IBM, Sun et Sybase simplifie et accélère considérablement le développement dôapplications J2EE.
Aussi, JBuilder comprend de nombreuses fonctions pour aider à développer des applications J2EE.
Voici les technologies dont dispose JBuilder pour le développement au niveau client :

-Technologies du niveau client :

Å **Applets:** Les applets sont un type spécial dôapplication Java qui sont téléchargées et exécutées par un navigateur Web sur une machine client.
Å **Applications d'interface utilisateur Java :** JBuilder comporte plusieurs fonctions qui peuvent aider à développer une application sôexécutant sur une machine client.

Un serveur Web et/ou un conteneur EJB peuvent sôexécuter sur le niveau intermédiaire.
JBuilder est livré avec Tomcat, un conteneur de servlet qui peut être utilisé comme serveur web.

-Technologies du niveau intermédiaire :
Voici les technologies J2EE du niveau intermédiaire qui utilisent un serveur Web :

on Java côté serveur pouvant traiter les requêtes issues
requêtes en générant des sorties dynamiques renvoyées au client.

Å**Pages JavaServer (JSP):** Les pages JavaServer (extension de la technologie servlet) offrent une façon simplifiée de développer des servlets. Comme les servlets, elles génèrent des sorties dynamiques qui sont renvoyées au navigateur Web du client. InternetBeans Express est une bibliothèque de composants qui complète la technologie JSP et servlet disponible dans JBuilder. Cette bibliothèque facilite la présentation et la manipulation des données dôune base de données afin de construire des pages JavaServer et des servlets orientés données.

Voici les technologies J2EE du niveau intermédiaire qui utilisent un conteneur EJB :

Å**Enterprise JavaBeans (EJB):** Les Enterprise JavaBeans sont des composants côté serveur qui contiennent la logique métier de lôapplication.

Tous les composants application Web et enterprise bean peuvent être combinés et délivrés dans un fichier EAR (Enterprise Archive). JBuilder dispose dôun expert EAR pour la création de vos fichiers EAR.

Autres technologies J2EE :

Ces technologies contribuent au bon déroulement des opérations :

Å**JDBC (Java Database Connectivity):** JDBC est la norme utilisée pour accéder à une base de données sur le niveau des systèmes dôinformation dôentreprise (Enterprise Information Systems, EIS). JDBC définit une API Java pour écrire des instructions SQL envoyées à une base de données.

JBuilder inclut DataExpress, une bibliothèque de composants pour lôaccès aux données dôune base de données. Elle connecte votre application à votre base de données à lôaide de pilotes JDBC.

JBuilder inclut également JDataStore, une bibliothèque de composants et une base de données imbriquée entièrement Java.

Å**JMS (Java Message Service):** JMS est un service de messagerie dôentreprise qui achemine les messages entre les composants et sôexécute dans une application distribuée.

Å**JNDI (Java Naming and Directory Interface):** Tous les serveurs J2EE utilisent JNDI, un service de nommage Java utilisé pour localiser les objets distribués.

ıage): Bien que XML ne soit pas réellement une ıent utilisé par les technologies J2EE. Par exemple, les descripteurs de déploiement des composants Web et des enterprise beans sont écrits en XML. Ces descripteurs décrivent le comportement des composants lorsquôils sont déployés.

Notons quôau fur et à mesure de la création et de la compilation des applications Web et enterprise beans, JBuilder peut créer automatiquement les fichiers WAR (Web Archive) et EJB-JAR (EJB Archive).

Aussi, le module de mise à jour Java 2 édition Entreprise (J2EE) fournit lôarchitecture nécessaire à une plate-forme de développement dans le cas dôapplications dôentreprise distribuées.
Elle a été développée par Sun Microsystems, avec des entrées provenant de la communauté de développement, y compris Borland.
Les produits de la plate-forme J2EE permettent au développeur de construire des applications avec les avantages suivants :

ÅFiabilité et évolutivité, afin de traiter rapidement et avec exactitude les transactions métier.

ÅExcellente sécurité pour protéger la vie privée des utilisateurs et lôintégrité des données de lôentreprise.

ÅDisponibilité pour satisfaire les demandes croissantes de lôenvironnement métier global.

Notons que lôapproche multi- niveau adoptée par la plate-forme J2EE offre plusieurs avantages :

ÅElle réduit la complexité du développement distribué avec une architecture simplifiée et le partage de la charge de travail parmi les rôles.

Å Côest une solution hautement évolutive qui permet le développement de systèmes satisfaisant de nombreux besoins modifiables rapidement.

système, la logique peut être mise à jour facilement en diaire sans charger une nouvelle logique sur chaque machine client.

ÅLes nouvelles applications peuvent sôintégrer correctement avec les systèmes dôinformation existants. JDBC (technologie J2EE) est une API Java pour les bases de données SQL permettant lôaccès à tous les types de données tabulaires possibles dans lôentreprise. Lôinterface de répertoire et de nomenclature Java (JNDI, Java Naming and Directory Interface) permet aux applications utilisant la technologie Java dôaccéder aux services de répertoire et de nommage dôentreprise. Lôarchitecture J2EE Connector fournit des connexions dôapplications Java aux systèmes hétérogènes existants. Le service de messages Java (JMS, Java Message Service) est lôAPI Java pour lôenvoi et la réception de messages à travers les systèmes de messagerie dôentreprise. Les services CORBA sont appelés au moyen de JavaIDL.

ÅLa sécurité est améliorée. Les technologies J2EE sont conçues en prenant en compte la sécurité. Par exemple, seuls les utilisateurs dans des rôles assignés peuvent accéder à certaines méthodes des enterprise beans. Les informations sur les utilisateurs pouvant accéder à ces méthodes ne sont pas codées dans les enterprise beans eux-mêmes. Ces informations sont définies dans les descripteurs de déploiement des enterprise beans, utilisés par le déployeur pour établir leur comportement après le déploiement. Ce type de sécurité est appelé sécurité déclarative.

J2EE permet également une sécurité programmable. Dans ces cas, la logique de sécurité est placée dans le code lui-même.

Å Les développeurs peuvent choisir parmi une diversité dôoutils de développement, de serveurs et de composants pour développer les applications requises. Lôéquipe de développement peut sélectionner les meilleures solutions pour leurs besoins, sans être verrouillé par lôoffre dôun fournisseur unique.

[ALOU 02] ALOULOU. M et PORTMANN
A genetic algorithm to achieve scheduling flexibility for a single machine problem.
RAIRO- operation research, 2002.

[AISS 06] AISSANI N.
Amélioration de la performance d'un système de pilotage en temps réel par apprentissage automatique,
Mémoire de Magister, département Informatique, Université d'Oran, 2006.

[AISS 08 a] AISSANI N., TRENTESEAUX D. et BELDJILALI B.
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, p 474 - 497, 2008.

[AISS 08 b] AISSANI N., BELDJILALI B.
On-line scheduling of Automatics and flexible Manufacturing System using SARSA technique.
Cisa 2008.

[AITS 08 a] AIT SI LARBI E., AISANI N., BELDJILALI B.
Planification des entreprises multi sites : Vers une approche génétique.
Proceeding de: MCSEAI: 10th Maghrebian Conference on Information Technologies. 28- 30 Avril 2008, Oran, Algérie. p.506- 511.

[AITS 08 b] AIT SI LARBI E., AISSANI N., BELDJILALI B.
Un Modèle de Planification pour les Entreprises Multi Sites à Base de Systèmes Multi Agents et Algorithmes Génétiques.
Proceeding de: SNIB'08: Séminaire National en Informatique de Biskra. 06- 08 Mai 2008. Biskra, Algérie. p.139-149.

[AKTU 99] AKTURK M.S. & GORGULU M.S.
Match-up scheduling under a machine breakdown.
European Journal of Operational Research, vol. 112, p. 81_97, 1999.

[ARCH 99] ARCHIMEDE B., COUDERT T. and PRIEM L.
Ordonnancement dynamique dans les ateliers de fabrication. Une solution à base d'agents.
3rd International Industrial Engineering Conference, Proceedings Vol.3, Montréal, Québec, p. 1523- 1532. Mai 1999.

[AVEN 96] AVENIER M. J.
La «stratégie tâtonnante».
Rencontres Internationales de la Gestion, IIIème Congrès Mondial de l'AFSAM, Paris, du 8 au 11 juillet 1996.

ANIAN S. NORRIE D.H.
lligent design system integrating manufacturing and shop-floor control. Division of Manufacturing Engineering, Departement of Mechanical Engineering, University of Calgary, Canada. 1997.

[BEAU 99] BEAUDREE F., BRETHES T., CINQUIN L., GROISE E., HISQUIN F., MASSOL V., PEZZIARDI P., SCHNEIDER D.
Le livre blanc des serveurs dôapplications.
Mars 1999.

[BENA 06] BENAOUDA A., ZERHOUNI N., VARNIER C.
Une approche multi- agents coopératifs pour la gestion des ressources matérielles dans un contexte multi sites de e-manufacturing,
Conférence Francophone de MOdélisation et SIMulation MOSIMô06, du 3 au 5 avril 2006.

[BENA 90] BENASSY J.
La gestion de production.
Éditions Hermes, 1990.

[BERQ 01] BERQUE F., FREZEFOND S., SERRIAUX L.
Java-XML et Oracle,
Éditions Eyrolles, 2001

[BIER 99] BIERWIRTH, MATTFELD
Production scheduling and rescheduling with genetic algorithms, evolutionary computation, p. 1-17, 1999.

[BOUS 04] BOUSBIA S., TRENTESEAUX D.
Proposition dôun système de pilotage hétérarchique pour lôamélioration continue des performances: Approche basée sur lôapprentissage,
Journées GDR MACS, Aix 21 et 22 Octobre 2004.

[BRIN 81] BRNIDLE A.
Genetic algorithms for function optimization.
Thèse de PhD, Université Alberta, Edmonton, 1981.

[BURG 94] BURG B., ARLABOSSE F.
ARCHON : une plate-forme industrielle pour lôIAD.
Deuxième journées francophones Intelligence Artificielle Distribuée et Systèmes Multi- Agents, Voiron, p. 223-234. Mai 1994.

[BUSS 01] BUSSMANN S., SCHILD K.
An agent-based approach to the control of flexible production systems, 8th IEEE International conference on Emerging Technologies and Factory Automation, 15-18 Oct. 2001, Vol. 2, p. 481 ï 488.

[CAMP 94] CAMPS V. CARPUAT B., GLEIZES M.P., GLIZE P., MACHONIN A., PIQUEMAL-BALUARD C., TROUILHET S.

e artificielle collective.
la journée multi agents du PRC-IA. Paris 16 décembre 1994.

[CHAA 09] CHAARI T., AISSANI N., BOUHAMOU F., TRENTESEAUX D., TAHON C.
Scheduling under uncertainty: state of the art,
Journal of Scheduling, 2009.

[CHUR 92] CHURCH, UZSOY. R.H
Analysis of periodic and event drive rescheduling policies in dynamic shops,
International Journal of Computer Integrated Manufacturing, Vol. 5, p.153-163, 1992.

[COUD 99] COUDERT T., ARCHIMEDE B., GRABOT B.,
Un système multi- agents pour la coopération Production/Maintenance.
Conférence Francophone de MOdélisation et SIMulation: MOSIM'99, Annency ï France. 6-8 Octobre 1999.

[DAVE 00] DAVENPORT A.J., BECK J.C.
A survey of techniques for scheduling with uncertainty;
http://www.eil.utoronto.ca/chris/chris.papers.html, 2000.

[DAVI 95] DAVID A.
RATP: la métamorphose, réalités et théorie du pilotage du changement,
Interédition, Paris, 1995.

[DAVI 85] DAVIS L.
Applying adaptive algorithms to epistatic domains.
Proceeding de: The International Joint Conference on Artificial Intelligence, p. 162-164, 1985.

[DECH 03] DECHAMBOUX P.
L'environnement J2EE : principes, fonctions, utilisation. ICAR'03, Ecole d'été sur les intergiciels et la construction d'applications réparties. 2003.

[DUVA 01] DUVALLET C.
Des systèmes d'aide à la décision temps réel et distribués : modélisation par agents.
Thèse de Doctorat de l'université du Havre. Octobre 2001.

[ELKH 03] ELKHYARI A.
Outil d'aide à la décision pour des problèmes d'ordonnancement dynamique,
Thèse de doctorat de l'université de Nantes, 2003.

[ESQU 99] ESQUIROL P. & LOPEZ P.
L'ordonnancement.

[FERB 95] FERBER J.
Les Systèmes Multi-Agent - Vers une Intelligence Collective, InterEditions, 1995.

[FERB 98] FERBER J., GUTKNECHT O.
Un méta- modèle organisationnel pour lôanalyse, la conception et lôexécution de systèmes multi- agents, Proc of 3rd International Conference on Multi- Agent Systems ICMASô98, p 128-135, 1998.

[FERB 00] FERBER J., GUTKNECHT O., MICHEL F.
The MADKIT Agent Platform Architecture,
Rapport de recherche. Laboratoire dôinformatique, de robotique et de micro électronique de Montpelier. Université Montpellier II C 09928. 05/2000.

[FERB 02] FERBER J., GUTKNECHT O.
MadKit User's Guide.
Laboratoire dôInformatique, Robotique et Micro- électronique de Montpellier. Version 3.1 -dernière modification: 09/07/2002.

[FERB 08] FERBER J.
Madkit pas à pas, démarrage te prise en main du logiciel Madkit. Mars 2008.

[FONT 01] FONTAN G., MERCE C., ERSCHLER J.
La planification des flux de production, Performance industrielle et gestion des flux,
Traité IC2 Information-Commande-Communication, N°ISBN 2-7462-0297-2, 2001, Chapitre 3, p.69-112. Hermes Lavoisier, 2001.

[GALLI 07] GALLIANO. D. SOULIE. N.
Organisational and spatial determinants of the multiunit firm: Evidence from the French industry,
Cahiers du GRES, p.17, 2007.

[GLEI 94] GLEIZES MP., GLIZE P., TROUILHET S.
Etude des lois de la conversation entre agents autonomes.
Revue internationale de systémique. Vol 8, p.39-50. 1994.

[GOEL 06] Sushant Goel, Rajkumar Buyya,
Data replication strategies in wide area distributed systems, 2006.

[GOLD 89] GOLDBERG D.E
Genetic Algorithms in Search, Optimization and Machine Learning, Addison Wesley Longman, 1989.

[GOLD 94] GOLDBERG D.E.

étiques Exploration, optimisation et apprentissage

Traduit de l'anglais (Américain) par V. Corruble, Edition Addison-Wesley, France, 1994.

[GOLD 85] GOLDBERG D.E., LINGLE R.
Alleles, loci, and the tsp. In Proceedings of the Fist International Conference on Genetic Algorithms, p. 154-159. Lawrence Erlbaum Associates, 1985.

[GOTH 93] GHOTA
Les problèmes d'ordonnancement, Recherche Opérationnelle/ Operations Research. Vol 27, n°1, p. 77- 150, 1993.

[GOTH 02] GHOTA
Flexibilité et Robustesse en Ordonnancement.
Le bulletin de la ROADEF, Vol. 8, p. 10-12, 2002.

[GOUA 99] GOUARDERES E.
Application de l'approche multi- agents au pilotage d'une cellule flexible de production,
3rd International Industrial Engineering Conference, Proceedings Vol.3, Montréal, Québec, p. 1695-1704. Mai 1999.

[GRAT 05] GRATACAP A., MEDAN P.
Management de la production: Concepts, Méthodes, Cas,
Dunod Collection Gestion Sup, 2005.

[HAFR 01] HAFRI Y., NAJID M.
Utilisation de l'approche multi- agents pour le pilotage en temps réel des systèmes de production, Conférence Francophone de MOdélisation et SIMulation MOSIM'01. 2001.

[HIRO 06] HIROYUKI. H., FURUYA. M
JIT Is Flow: Practice and Principles of Lean Manufacturing,
PCS Press, 2006.

[HOLL 75] HOLLAND J.H.
Adaptation in natural and artificial systems.
University of Michigan Press, 1975.

[JACK 55] JACKSON J.R.
Scheduling a production line to minimize maximum tardiness.
Technical report, Management Science Research Project, University de Californie, Los Angeles. Rapport de recherche 43. 1955.

[JACO 06] JACOB R.
Votre organisation, un espace de création,
Éditorial réseau CEFRIO, 2006, Vol.6 No 1.

Optimal two and three stage production schedules with setup times included.
Naval Research Logistics Quarterly, p. 61-68, 1954.

[JOSS 04] JOSSERAND E.
The Network Organisation: The Experience of French Multinationals (New Dimensions in Networks Series),
Lavoisier, 2004.

[KNAP 97] KNAPP M., GIESEKE W.
HTML et JAVA.
Editions Micro application. ISBN: 2-7429-0924-9. 1997.

[KOLI 01] KOLISCH R.
Make-to-order assembly management,
Springer -Verlag GmbH. 2001.

[LABE 07] LABED K.
Expérimentation des algorithmes génétiques multi objectifs dans un processus décisionnel multicritère en aménagement du territoire,
Mémoire de Magister, département dôinformatique, Université dôOran, 2007.

[LAHO 05] LA HOANG TRUNG
Utilisation dôordres partiels pour la caractérisation de solutions robustes en ordonnancement.
Thèse de doctorat à l'Institut National des Sciences Appliquées de Toulouse, 2005.

[LARO 83] Le petit Larousse illustré.
1983.

[MAIO 01] MAIONE B., NASO D.
Evolutionary adaptation of dispatching agents in heterarchical manufacturing systems.
International journal of production research, Vol. 39, n°7, p. 1481-1503, 2001.

[MARI 05] MARIK V., VRBA P., HALL K.H., MATURANA F. P.
Rockwell Automation Agents for Manufacturing,
AAMAS'05, Utrecht, Pays-Bas, Juillet 25-29 2005.

[MASO 00] MASON G., BELTRAMO JP., PAUL JJ.
Knowledge infrastructure, technical problem solving and industrial performance: electronics in Britain and France,
DRUID Summer Conference on the Learning Economy, University of Aalborg, 2000.

ZSOY R.H.
ling of a single machine subject to breakdowns.
International Journal of Computer Integrated Manufacturing, Vol. 12, no. 1, p. 15-38, 1999.

[MEYR 05] MEYR H. WAGNER M. ROHDE J.
Structure of Advanced Planning Systems in Supply Chain Management and Advanced Planning.
Springer Berlin Heidelberg, p.109-115, 2005.

[MILE 92] MILES R. E., SNOW C. C.
Managing 21st century network organisations. Organizational Dynamics, printemps 1992.

[MINT 80] MINTZBERG H.
Structure in 5ôs: a synthesis of the research,
Organization design Management Science, Vol 26, n°3, Mars 1980.

[MINT 93] MINTZBERG H.,
The pitfalls of strategic planning,
California Management Review, fin 1993.

[MINT 85] MINTZBERG H. WATERS J.A.
Strategies, deliberate and emergent,
Strategic Management Journal, Vol 6, 1985.

[MOLI 05] MOLIERE J.
Les cahiers du programmeur J2EE.
Groupe Eyrolles, ISBN : 2-212-11574-1, 2003, 2005.

[MOYA 00] MOYAUX T.
Spécification de comportement dôagents dans un système multi- agents.
Mémoire DEA de productique et informatique. Université de droit, dôéconomie et des sciences dôAix-Marseille III. 2000.

[MULL 02] MULLER J.P.
Des systèmes autonomes aux systèmes multi- agents : Interaction, émergence et systèmes complexes.
Rapport présenté pour lôobtention de lôHabilitation à Diriger les Recherches en Informatique (27ème section) le 8 novembre 2002.

[NEUB 97] NEUBERT G.
Contribution à la spécification dôun pilotage proactif et réactif pour la gestion des aléas.
Thèse pour lôobtention de grade de docteur. Décembre 1997.

[OLIV 87] OLIVER I.M., SMITH D.J., HOLLAND J.H.
A study of permutation crossover operators on the traveling salesman problem.

The Second International Conference on Genetic
24-230. Lawrence Erlbaum Associates, 1987.

[PIRA 06] PIRARD F., RIANE F., IASSINOVSKI S.
Une démarche hybride dôaide à la décision pour la reconfiguration et la planification stratégique des réseaux logistiques des entreprises multi sites,
Conférence Francophone de MOdélisation et SIMulation - MOSIMô06, du 3 au 5 avril 2006.

[RAMO 94] RAMOS C.
An Architecture and a Negociation Protocol for the Dynamic Scheduling of Manufacturing Systems,
IEEE, p. 1050-4729. 1994.

[REND 94] RENDERS J.M.
Algorithmes génétiques et réseaux de neurones,
Editions Hèrmes, France, 1994.

[RICO 01] RICORDEL P. M.
Programmation orientée multi agents : Développement et Déploiement de Systèmes Multi- Agents Voyelles,
Thèse pour obtenir le grade de docteur de lôINPG, 25 Octobre 2001.

[RORI 05] RORIVE B.
Lôorganisation des entreprises en réseau,
La revue de la CFDT, p. 27-31, Septembre- Octobre 2005.

[ROY 97] ROY D., VERNADAT F.
Reactive Shop-Floor Control with Multi- Agent System.
IFAC/IFIP Int. Conf. On Management and Control of Production and Logistics, Brésil, p. 426-431. 1997.

[SAUE 00] SAUER J., FREESE T., TESCHKE T.
Towards Agent- Based Multi-Site Scheduling,
ECAI 2000 Workshop on New Results in Planning, Scheduling, and Design, p. 123-130, Berlin, 2000.

[SHEI 02] SHEIKH K.
Manufacturing resource planning (MRP II) with introduction to ERP, SCM and CRM,
Lavoisier, 2002.

[SHEN 98] SHEN W., NORRIE D.H.
An Agent-Based Approach for Dynamic Manufacturing Scheduling.
In workshop notes of the agent-based manufacturing workshop at autonomous agentsô98. Division of Manufacturing Engineering, Université de Calgary, Canada. 1998.

[SMIT 56] SMITH W.E.

rs for single-stage production.
Logis- tics Quarterly, Vol. 3, p.59-66, 1956.

[SMIT 95] SMITH F. S.
Reactive scheduling systems, in Brown, D. and Scherer, W. T. (Eds.), Intelligent Scheduling Systems, p. 155-192, Kluwer Academic Publisher, 1995.

[SPAL 99] SPALANZANI A.
Algorithmes évolutionnaires pour lôétude de la robustesse des systèmes de reconnaissance automatique de la parole,
Thèse de Doctorat, Université Joseph- Fourier - Grenoble I, France, 1999.

[TAGH 08] TAGHZOUT N.
Un système multi agents pour lôaide à la décision en temps réel appliquée à une gestion de production dynamique.
Thèse de doctorat, Université dôOran. Décembre 2008.

[TAII 88] TAIICHI O.
Just-In-Time for Today and Tomorrow, Productivity Press, 1988.

[TARO 01] TARONDEAU J.C., WRIGHT R.W.
La transversalité dans les organisations ou le contrôle par les processus, Revue Française de Gestion, Juin- Juillet- Août, 1995.

[TAYU 99] TAYUR S., GANESHAN R., Magazine M.
Quantitative models for supply chain management, Kluwer Academic Publishers, 1999.

[TEMP 01] TEMPELMEIER H.,
Master Planning mit Advanced Planning Systems,
Books on Demand GmbH, 2001.

[THIE 03] THIERRY C.
Gestion de chaînes logistiques, modèles et mise en oeuvre pour lôaide à la décision à moyen terme,
Mémoire dôhabilitation à diriger des recherches, 25 Juin 2003.

[TRAN 01] TRANVOUEZ E.
IAD et ordonnancement : Une approche coopérative du réordonnancement par systèmes multi agents.
Thèse pour obtention de grade de docteur en sciences.Université de droit, dôéconomie et des sciences dôAix Marseille. 23 Mai 2001.

[TRAN 99] TRANVOUEZ E., ESPINASSE B., FERRARINI A.
Résolution coopérative et distribuée de problèmes : Une application Multi- Agents au réordonnancement dôatelier.
3rd International Industrial Engineering Conference, Proceedings, Vol.3, p. 1543-1552. Mai 1999.

D.
Pilotage hétérarchique des systèmes de production.
Habilitation à diriger des recherches, 19 Décembre 2002.

[TREN 07] TRENTESEAUX D.
Les systèmes de pilotage hétérarchiques : innovations réelles ou modèles stériles ?
JESA 2007

[VACH 00] 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*M,
Thèse de doctorat de lôuniversité du Havre, 2000.

[VERN 97] VERNADAT F.
A process/Agent /Operation paradigm for manufacturing Systems modelling.
Proceedings de IFAC/IFIP conference on management and control of production and logistics (MCPLô97) Campinas. SP (Brésil). Vol. 2, p. 412-419. 31 Août ï 3 Septembre 1997.

[VIEI 00] VIEIRA G.E, HERRMANN J.W, LIN E.
Analytical models to predict the performance of single-machine system under periodic and event-driven rescheduling strategies.
International journal of production research, p.1899- 1915, 2000.

[WOOL 95] WOOLDRIDGE M., JENNINGS N.
Intelligent Agents: Theory and Practice,
Knowledge Engineering Review, Vol. 10, No. 2, p. 115-152, 1995.

[WU 93] WU S.D, STORER R.H., CHANG P.C.
One-machine rescheduling heuristics with efficiency and stability as criteria.
Computers and operations research, Vol. 20, p.1-14, 1993.

Références Internet :

[Net 1] Cours sur les Algorithmes Génétiques:
http://www.enseignement.polytechnique.fr/profs/informatique/Eric.Goubault/poly/ cours009.html

[Net 2] http://www.madkit.org

[Net 3] http://java.sun.com/j2ee/1.4 /download.html

[Net 4] http://www.jmdoudoux.fr/accueil_java.htm