ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

COMME EXIGENCE PARTIELLE
À L'OBTENTION DE LA
MAÎTRISE EN GÉNIE DE LA PRODUCTION AUTOMATISÉE
M. Ing.

PAR
BENOÎT SAENZ DE UGARTE

INTEGRATION DE LA SIMULATION ET DE L'OPTIMISATION POUR L'AIDE À LA DÉCISION

MONTRÉAL, LE 8 DECEMBRE 2006

CE MÉMOIRE A ÉTÉ ÉVALUÉ

PAR UN JURY COMPOSÉ DE :

M. Artiba, Abdelhakim, directeur de mémoire
Département de génie de la production automatisée à l'École de technologie supérieure

M. Gharbi, Ali, codirecteur de mémoire
Département de génie de la production automatisée à l'École de technologie supérieure

M. Beneditti, Claudio, président du jury
Département de génie de la production automatisée à l'École de technologie supérieure

M. Pellerin, Robert, examinateur externe
Département de mathématiques et de génie industriel à l'École Polytechnique de Montréal

IL A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET PUBLIC

LE 4 DÉCEMBRE 2006

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

INTÉGRATION DE LA SIMULATION ET DE L'OPTIMISATION POUR L'AIDE À LA DÉCISION

Benoît Saenz de Ugarte

SOMMAIRE

Le besoin de réviser les priorités de fabrication pour compenser des conditions changeantes sur la demande dans l'optique de respecter les objectifs financiers a poussé un certain nombre d'entreprises à passer d'une mentalité de « réaction aux changements sur les prévisions » à un raisonnement à base de « processus adaptatifs ». Dans ce contexte, les entreprises ont besoin de détecter les exceptions de façon proactive pour améliorer les procédés en temps réel. Les organisations manufacturières adaptatives ont aussi besoin d'une plateforme d'exécution qui connecte les processus manufacturiers au reste de l'entreprise et de la chaîne logistique et qui fournisse une aide à la décision au personnel de l'atelier, leur permettant de livrer tout en respectant les objectifs de performance.

Nous proposons une architecture générique permettant de supporter les processus manufacturiers adaptatifs. Cette architecture se compose d'un ERP, d'un MES et d'un module décisionnel en temps réel basé sur la simulation. Le concept est indépendant des outils logiciels qui seront utilisés pour le mettre en œuvre. Pour l'intégration des systèmes, nous préconisons l'utilisation d'outils et d'interfaces standards tels que le standard OPC ou la norme S95. Pour tester sa faisabilité, l'architecture proposée est mise en œuvre dans le cadre d'un scénario tiré d'un cas réel de l'industrie de l'aluminium. Toutes les étapes, depuis la conception des processus adaptatifs jusqu'à la création des composants d'intégration, sont détaillées et peuvent servir de méthodologie de base pour un cas réel d'intégration.

Sur la base de cette expérience, nous discutons des difficultés de l'intégration des solutions propriétaires et des risques d'instabilité que peut entraîner, dans certains cas, une utilisation trop intensive de l'optimisation. Nous abordons aussi le potentiel que représente l'approche orientée service pour les organisations manufacturières adaptatives.

SIMULATION AND OPTIMIZATION INTEGRATION FOR THE DECISION-MAKING SUPPORT

Benoît Saenz de Ugarte

ABSTRACT

The need of balancing manufacturing priorities against changing demand conditions in alignment with business targets has brought a number of manufacturers to move from a "react to forecast changes" mindset to "adaptive processes". In that context, manufacturers need to proactively detect exceptions to improve processes in real-time. Adaptive manufacturing organizations also need an execution platform that connects manufacturing processes with enterprise and supply chain processes and that provides decision support to production personnel so they can deliver on their performance goals.

We propose a generic architecture to support the adaptive manufacturing processes. This architecture is composed of an ERP, a MES and a real time decision-making module based on simulation. The concept is independent of the software tools the enterprise will use to implement this architecture. To integrate systems, we recommend the use of standard tools and interfaces such as the S95 or OPC standards. To test its feasibility, our suggested architecture is implemented within a demonstrative scenario based on a real case of the aluminium industry. All the steps are detailed, since the design of the adaptive process until the implementation of the integration components, and it can be used as basic methodology for a real case of integration.

On the basis of this experiment, we discuss the difficulties to integrate owner solutions and the risks of instabilities, in particular cases, inhérent in a too intensive use of optimization. We also discuss the potential of the service-oriented architecture for the adaptive manufacturing organizations.

REMERCIEMENTS

Je tiens à remercier chaleureusement mon directeur de recherche, M. Abdelhakim Artiba, mon codirecteur de recherche, M. Ali Gharbi, tous deux professeurs à l'École de Technologie Supérieure de Montréal, ainsi que M. Robert Pellerin, professeur à l'École Polytechnique de Montréal, qui ont rendu possible ce travail par leur écoute, leur soutien et leurs conseils judicieux.

Je remercie particulièrement M. Charles-André Horth de STI pour nous avoir proposé le scénario de l'aluminerie et pour nous avoir fourni suffisamment d'information pour mener à bien notre expérimentation. Je remercie aussi M. Johan Destrooper pour nous avoir partagé sa vision et ses connaissances des processus adaptatifs d'entreprise. Je remercie finalement toute l'équipe de SIMPARTNERS pour leur support et leur rapidité d'implémentation de nouvelles fonctionnalités à leur produit, SDBuilder.

Je remercie aussi les étudiants du laboratoire recherche sur les processus manufacturiers adaptatifs de l'École Polytechnique pour la bonne humeur qu'ils ont apporté au quotidien et les connaissances qu'ils ont accepté de partager avec moi.

J'ai une pensée toute particulière pour ma famille. Je remercie mes parents pour leur aide et leur appui moral malgré la distance qui nous sépare. Enfin, un remerciement tout particulier à Marcela Santos do Carmo pour sa confiance, sa patience et sa ténacité.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES ABRÉVIATIONS ET SIGLES

ANSI	*American National Standards Institute*
API	Interface de programmation ou *Application Program Interface*
APS	Système avancé de planification ou *Advance Planning System*
BAPI	*Business Application Program Interface*
B2B	Commerce interentreprises ou *Business to Business*
B2C	Commerce orienté client ou *Business to Customer*
COM	*Component Object Model*
CORBA	*Common Object Request Broker Architecture*
CRM	Gestion des relations avec la clientèle ou *Customer Relationship Management*
CSV	*Comma-Separated Values*
DCOM	*Distributed Component Object Model*
DCS	Système de commande distribué ou *Distributed Control System*
DDE	Échange Dynamique de données ou *Dynamic Data Exchange*
DNC	Commande numérique distribué ou *Distributed Numerical Control*
DSS	Système d'aide à la décision ou *Decision Support System*
EDD	Règle de la date d'éxigibilité la plus tôt ou *Earlier Due Date*
ERP	Progiciel de gestion intégré ou *Enterprise Resource Planning*
IHM	Interface Homme Machine
ISA	*Instrumentation, Systems & Automation Committee*
JCo	*Java Connector*
LPT	Règle du temps d'opération le plus long ou *Largest Processing Time*
MES	Système d'exécution de la fabrication ou *Manufacturing Execution System*
MESA	*Manufacturing Execution System Associtation*
MRP	Planification des besoins matières ou *Material Requirements Planning*

MRPII	Planification des ressources de production ou *Manufacturing Resource Planning*
MTBF	Intervalle moyen entre les défaillances ou *Mean Time Between Failure*
MTTR	Délai moyen de réparation ou *Mean Time To Repair*
NOF	Numéro d'OF
OF	Ordre de Fabrication
OLE	*Object Linking and Embedding*
OLTP	Traitement transactionnel en ligne ou *On-Line Transaction Processing*
OLAP	Traitement analytique en ligne ou *On-line Analytical Processing*
OPC	*OLE for Process Control*
P&PE	Ingénierie des procédés et des produits ou *Product and Process Engineering*
PDP	Programme Directeur de Production
PIC	Plan Industriel & Commercial
PLC	Automate programmable industriel ou *Programmable Logic Controller*
PLM	Gestion du cycle de vie des produits ou *Product Lifecycle Management*
PRM	*Purdue University Reference Model*
RFC	Appel de fonction distante ou *Remote Function Call*
RO	Recherche Opérationnelle
SCADA	Télésurveillance et acquisition de données ou *Supervisory Control And Data Acquisition*
SCM	Système de gestion de la chaîne logistique ou *Supply Chain Management*
SPT	Règle du temps d'opération le plus court ou *Shortest Processing Time*
SOA	*Service-Oriented Architecture*
SQL	*Structured Query Language*
SSM	*Sales and Service Management*
TRS	taux de rendement synthétique
UML	*Unified Modeling Language*
XML	*eXtensible Markup Language*

XSLT *eXtensible Stylesheet Language Transformation*

INTRODUCTION

Les progiciels de gestion intégrés plus connus sous le nom de *Enterprise Resource Planning* (ERP) sont conçus pour intégrer tous les processus d'affaire internes à l'entreprise [1]. Malgré des coûts de maintenance et d'intégration élevés [2], le taux d'adoption des ERP continue d'augmenter. À tel point qu'aujourd'hui 80% des entreprises figurant dans le classement de Fortune 500 utilisent un ERP pour gérer leurs opérations et qu'un nombre croissant d'organismes publics et de petites et moyennes entreprises sont en train d'adopter la même stratégie [3].

Néanmoins, les ERP présentent toujours un certain nombre de limitations dans le domaine de la gestion de production. La première d'entre toutes provient de l'héritage de deux défauts associés à sa fonction de planification. Les ERP utilisent en effet la méthode de décomposition des besoins appelée *Material Requirements Planning* (MRP1) qui permet de synchroniser les stocks de produits finis, de sous-ensembles, de composants et de matières premières en quantité et dans le temps en fonction des demandes commerciales réelles et prévisionnelles [4], [5]. Cette méthode repose sur deux hypothèses simplificatrices : la capacité des ressources est illimitée et de nature déterministe [6]. En plus de cela, les ERP ont besoin de systèmes externes additionnels pour collecter et contrôler les données en temps réel [7]. Finalement, les ERP sont conçus pour collecter et retracer des événements, mais ne les exploitent pas dans le cadre d'un processus d'aide à la prise de décisions [8]. En conséquence, l'inaptitude des ERP à appréhender les incertitudes et les évènements imprévisibles limite leur utilisation comme un support au processus de prise de décisions dans un environnement dynamique de production.

Dans un tel contexte, il n'est pas surprenant que les départements de production aient toujours favorisé le développement d'applications et d'outils hautement personnalisés répondant spécifiquement à leurs besoins et agissant comme support aux opérations industrielles. La collecte des données relatives à la production au travers de base de

données ou de chiffrier a été et est encore communément développée dans les ateliers de fabrication pour contrôler et commander en temps réel des processus incertains et dynamiques. Les logiciels de maintenance et de consolidation de données sont bien évidemment complexes dans de tels environnements et le nombre et la structure de ces petites applications varient grandement avec le temps.

La difficulté d'intégrer et de faire interagir ces multiples systèmes a poussé les fournisseurs de logiciels à regrouper tous ces composants servant à la gestion de la production en une solution unique et intégrée. Ces systèmes d'exécution de la production, généralement appelés *Manufacturing Execution System* (MES), proposent une interface d'utilisation commune ainsi qu'un système de gestion des données. De tels outils couvrent un très large éventail de fonctionnalités et peuvent servir à la gestion des opérations de production depuis l'arrivée des ordres de production dans l'atelier jusqu'à la livraison du produit fini [9]. En utilisant des données actualisées et précises, les MES peuvent guider, signaler et lancer l'exécution d'activités à l'apparition d'événements dans l'environnement de production.

Le concept MES permet aussi de créer le lien entre le système de gestion de l'entreprise et l'atelier. Il met à disposition des outils de communication en provenance des innovations d'Internet et rend plus transparentes les opérations. Il augmente la compétitivité sur le marché des entreprises qui l'utilise [10]. Par tous ces aspects, le MES intéresse particulièrement les entreprises. Le Tableau II présente, pour différents outils logiciels, les taux d'implantation de ceux-ci ainsi que le taux de prévision d'implantation. On remarque alors que le MES est l'un des outils les moins implémentés dans le monde de l'industrie, mais que le potentiel du marché MES est énorme.

Tableau I

Taux d'implantation des différents outils logiciels des entreprises en 2001
(source : IndustryWeek's 2000 Census of Manufacturers)

	Implémenté	Prévu	Non prévu
Computer-aided design	72%	9%	19%
Bar coding	47%	36%	17%
MRPII	38%	21%	41%
Advanced planning/scheduling	38%	41%	21%
Computerized maintainance management	36%	34%	30%
Computer-integrated manufacturing	35%	30%	35%
Forecast/demand-management software	27%	30%	43%
Enterprise resource planning	25%	28%	47%
Product data management	23%	25%	52%
Warehouse management systems	22%	24%	54%
Transportation management systems	14%	20%	66%
Manufacturing execution systems	**12%**	**24%**	**64%**

En accord avec l'étude du cabinet Marketing international Frost & Sullivan [11], le nombre d'entreprise investissant des fonds significatifs dans le MES ne cesse de croître. Enregistrant des ventes pour 976,6 millions de dollars en 2001, Frost & Sullivan prévoient que le marché européen du MES atteindra le chiffre de 2,12 milliards de dollars en 2008. Avec 65,3% des revenus totaux en 2001, les services constituent le segment le plus important du marché MES européen. La complexité des installations, le haut niveau de sophistication des opérations qu'une solution MES nécessite pour fonctionner sont responsables de l'importante proportion des activités de services dans le marché MES. Cette tendance est particulièrement évidente dans l'industrie automobile à cause du nombre important de facteurs menant à une amélioration globale de l'efficacité.

Il en va de même pour l'industrie de l'agroalimentaire, la raison étant ici une nécessité croissante de traçabilité du produit depuis le producteur jusqu'au consommateur.

Dans le cadre d'un travail effectué pour la société Simpartners, j'ai effectué une succincte étude de marché portant sur la couverture des fonctions de 15 MES destinés au marché des moyennes entreprises (50 à 250 salariés). Les fonctions couvertes sont présentées dans le Tableau II. Les taux de couverture des fonctions sont présentés dans la Figure 1. Les détails pour chacun des MES sont fournis dans l'Annexe 1.

Tableau II

Fonctionnalités couvertes par les MES retenus pour l'étude de marché

1	Collecte et acquisition des données
2	Analyse des performances
3	Traçabilité du produit et généalogie
4	Ordonnancement
5	Gestion des documents
6	Cheminement des produits et des lots
7	Gestion du personnel
8	Gestion de la qualité
9	Gestion du procédé
10	Gestion de la maintenance
11	Gestion des ressources

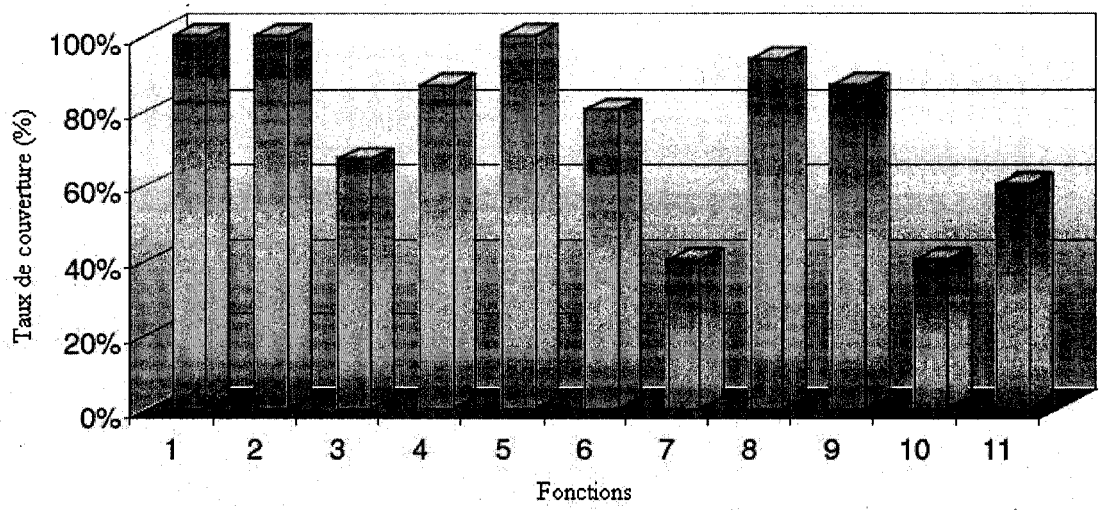


Figure 1 Couverture fonctionnelle des MES retenus pour l'étude de marché

Cette étude se base sur des documents commerciaux et la couverture (ou non) d'une fonctionnalité a été jugée en « tout ou rien ». Un test logiciel plus poussé aurait peut-être pu permettre de déterminer la qualité de la couverture de ces fonctionnalités. De plus, il n'est pas rare qu'un produit soit développé pour répondre spécifiquement à la demande d'un secteur particulier rendant la comparaison des divers produits de l'offre MES encore plus délicate.

Il est cependant possible de tirer quelques conclusions utiles de cette étude. L'un des constats les plus frappant est que les vendeurs de MES ont peu de recul sur les apports de leur produits et se contentent généralement de citer les apports mis en évidence par la *Manufacturing Execution System Associtation* (MESA, cf. Chapitre 1). Ensuite , l'offre MES est assez homogène en terme de fonctionnalité couverte et reprend là encore les recommandations de la MESA. Enfin, l'aide à la décision, quand elle est évoquée, semble généralement se limiter à la présentation des données critiques pouvant être utile à la résolution du problème. À aucun moment, le recours aux techniques de la recherche opérationnelle (RO) ou à la simulation pour présenter des solutions alternatives n'a été évoqué.

Le besoin de réviser les priorités de fabrication pour compenser des conditions changeantes sur la demande dans l'optique de respecter les objectifs financiers a poussé un certain nombre d'entreprises à passer d'une mentalité de « réaction aux changements sur les prévisions » à un raisonnement à base de « processus adaptatifs ». Dans ce contexte, les entreprises ont besoin de détecter les exceptions de façon proactive, de résoudre ces difficultés en collaborant avec leurs fournisseurs et leurs sous-traitants, et d'en tirer les leçons qui permettront l'amélioration des procédés en temps réel. Les organisations manufacturières adaptatives ont aussi besoin d'une plateforme d'exécution qui connecte les procédés de fabrication au reste de l'entreprise et de la chaîne logistique et qui fournisse une aide à la décision au personnel de l'atelier, leur permettant de livrer tout en respectant les objectifs de performance.

Ainsi, les MES n'abordent pas tous les processus de production nécessaires pour pouvoir intervenir sur l'ensemble de la chaîne logistique quand ils tentent de répondre dynamiquement à un événement imprévu. Les stratégies de production adaptative nécessitent aussi de bien définir la couche de gestion des processus entre les ERP et les MES, de traduire les besoins de la chaîne logistique, tirés par la demande du client, en investissements dans l'intégration des processus d'affaires, des systèmes et des aptitudes [12]. Ainsi, la production adaptative a besoin d'une plateforme d'exécution qui connecte les processus de production aux processus d'affaire de l'entreprise et de la chaîne logistique, qui rend possible les mécanismes en boucle fermée et qui fournit un support à la prise de décision au personnel, lui permettant d'atteindre ses objectifs de performance.

Traditionnellement, la simulation est utilisée pour améliorer les systèmes de production [13]. Les modèles de simulation sont principalement utilisés pendant la phase de conception soit pour comprendre le comportement d'un système soit pour évaluer diverses stratégies opératoires [8]. Ces modèles de simulation ne sont pas construits pour

un usage répétitif et en temps réel. En effet, ces modèles ne sont pas directement couplés aux systèmes d'informations en place [14].

Une des différences majeures entre la simulation traditionnelle et les systèmes d'aide à la décision en temps réel se trouve au niveau des données. Ces derniers prennent une photo du statut actuel de l'atelier et tentent de prévoir les événements à venir. Dans un environnement d'affaires hautement compétitif connaissant des changements rapides, le système de décision doit avoir accès aux données critiques les plus récentes. Les systèmes d'aide à la décision doivent aussi être en mesure de combiner différents outils de résolution à de multiples sources de données pour élaborer des solutions tout cela dans un intervalle de temps compatible avec celle de l'horizon de planification.

Les avancées dans le domaine de la puissance de calcul que nous connaissons depuis les dernières décennies ont heureusement rendu possible l'optimisation basée sur des modèles de simulation en temps réel. Les derniers développements de la recherche offrent de grandes opportunités à la simulation et les bénéfices potentiels sont significatifs. Une application maintenant courante de la simulation en temps réel est le support pour des décisions proactives dans les problèmes d'ordonnancement des systèmes manufacturier [15]. On commence également à voir des modèles combinant à la fois les MES et la simulation pour commander des systèmes de production flexibles [16].

Jusqu'à présent, la littérature a surtout abordé les problèmes de production adaptative en proposant différents modèles d'intégration entre les ERP et les MES. Bien que ces modèles puissent résoudre certains problèmes d'interopérabilité, il n'ajoute pas d'outils d'aide à la décision à ces systèmes. Nous présentons ici une approche différente en proposant une plateforme d'exécution générique dans laquelle les modèles de simulation en temps réel à événements discrets viennent compléter le traditionnel modèle ERP/MES. L'objectif est d'avoir un outil d'aide à la décision capable de réagir en temps

à divers événements perturbant le processus de production, que ces événements proviennent du niveau de l'ERP ou du niveau du MES. L'intérêt de l'architecture présentée réside principalement dans sa capacité à pouvoir interagir avec les deux plus importants progiciels des entreprises, d'accéder à l'ensemble des données de l'entreprise bien que mise à jour sur des mailles de temps différentes. Ainsi alimenté par des données exactes et actualisées, le système d'aide à la décision basé sur la simulation, sera en mesure de réagir à divers évènements pouvant aussi bien surgir au niveau du MES que de l'ERP. Il permettra alors au personnel de prendre des décisions en ayant une meilleure vision du problème et de l'impact des décisions qu'il prendra sur les performances de l'entreprise.

Une fois l'architecture générique présentée, nous nous appliquerons à la mettre en œuvre au travers d'un scénario de production adaptative d'une aluminerie factice. L'objectif de ce scénario est triple. Il doit d'abord nous permettre de démontrer la viabilité de l'architecture générique proposée. Il doit aussi nous permettre d'aborder les aspects techniques qui permettront les dialogues entre les trois acteurs de l'architecture générique (ERP, MES et le système d'aide à la décision). Il doit enfin nous permettre de mettre en lumière les apports concrets de l'implantation d'une telle stratégie. Pour ce scénario, nous utiliserons 2 produits commerciaux largement répandus dans le secteur manufacturier, à savoir SAP dans le rôle de l'ERP, Proficy dans celui du MES. Le simulateur retenu sera SDBuilder.

Ce mémoire est structuré comme suit (Figure 2). Nous présentons d'abord au chapitre 1 une revue de la littérature. La partie portant sur les ERP et les MES nous permettra de relever les manques de chacun de ces systèmes s'ils sont pris indépendamment les uns des autres. Une partie de la revue de la littérature sera aussi consacrée à l'optimisation basée sur la simulation en temps réel et montrera les difficultés engendrées par ces différentes méthodes. Au travers de cette revue de littérature, nous évoquerons aussi des architectures existantes mettant en œuvre certains des systèmes déjà présentés.

L'architecture d'exécution générique sera ensuite présentée au chapitre 2. Le principal objectif de cette architecture est de fournir une aide à la décision en temps réel réagissant à des événements perturbateurs dans l'environnement manufacturier. La seconde partie du chapitre 2, ainsi que les chapitres 3 à 6 seront consacrés plus spécialement au scénario qui nous a permis de tester l'architecture générique que nous proposons. Ce scénario s'inspire librement d'un cas réel de l'industriel d'aluminium. Nous avons retenu ce scénario principalement à cause des contraintes industrielles qu'il faisait intervenir nous permettant ainsi de confronter notre architecture générique à la réalité d'un atelier de production. Ce scénario est présenté plus longuement dans la seconde partie du chapitre 2. Ce chapitre présente aussi les études préliminaires menées avant l'intégration de l'ERP, du MES et du module d'aide à la décision. Les chapitres 3 à 5 sont consacrés à la présentation de chacun des systèmes utilisés ici, à savoir SAP, Proficy et SDBuilder. Nous exposerons ici les rôles qui seront dévolus à chacun de ces systèmes, les techniques utilisées pour les intégrer au sein d'une architecture fonctionnelle. Le chapitre 6 présente, d'un point de vu plus général, le *modus operandis* de nos expérimentations, les résultats et les analyses que nous pouvons en tirer. Les discussions sur les résultats obtenus et les futurs développements envisagés viendront conclure ce chapitre et ce mémoire.

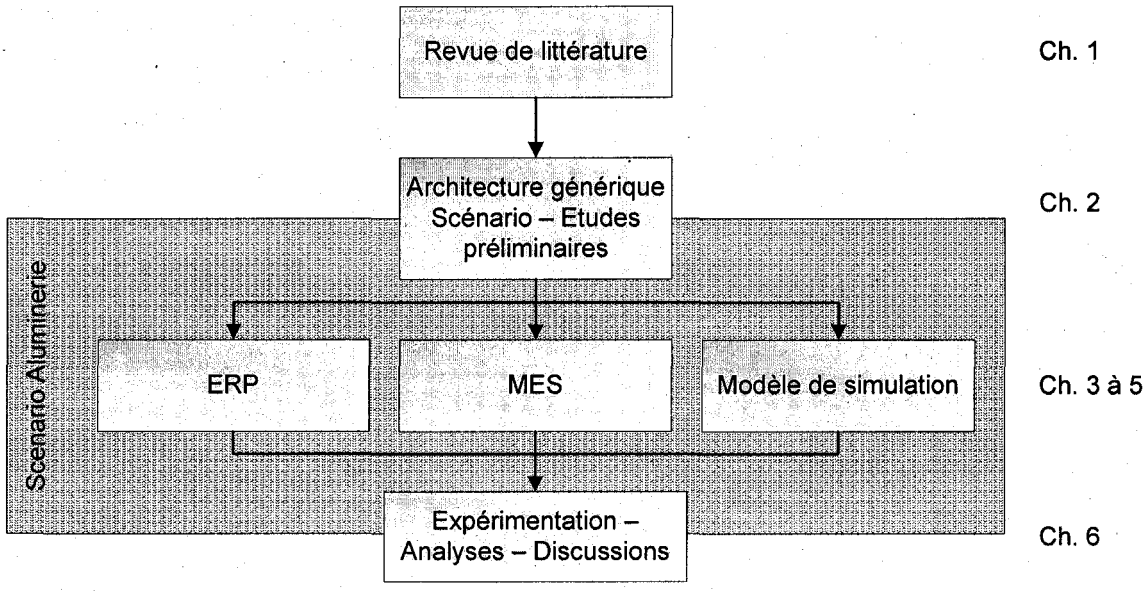


Figure 2 Démarche suivie et structure du mémoire

CHAPITRE 1

REVUE DE LA LITTERATURE

1.1 Introduction

Les systèmes d'exécution manufacturière (MES) ont été développés pour combler le manque de connexion entre les systèmes de contrôle de l'atelier et les ERP. Historiquement, le MES participe au concept d'usine numérique et forme un pont entre la gestion de la production (planification, stock, allocation des ressources) et le contrôle du processus (régulation, acquisition de données, alarmes). Ce chapitre présente une revue de la littérature des MES. Il aborde des points aussi variés que les modèles d'architecture, de connectivité, d'intégration avec les autres systèmes d'informations. Nous présentons rapidement les autres systèmes d'information de l'entreprise et nous nous attardons un peu plus longuement sur le plus connu d'entre eux, l'ERP. La force des processus adaptatifs résident dans la capacité du système à réagir aux événements et à proposer des actions correctives. C'est pourquoi nous abordons aussi les aspects d'optimisation et de simulation en temps réel. Nous terminons enfin sur les limitations actuelles de ces systèmes.

1.2 Revue des systèmes MES

1.2.1 Définition et fonctionnalités

Depuis plus de 25 ans, les entreprises ont investi dans des systèmes informatiques pour les aider à produire au mieux. Au cours des 10 dernières années, les produits commerciaux se sont partagés un marché en pleine croissance. Certains de ces produits, tel que les ERP ont gagné un statut majeur sur le marché et ont été largement implantés. Mais l'atelier est longtemps resté le parent pauvre en matière d'informatique [17]. Désormais il dispose de son système d'information, le *Manufacturing Execution System*

(MES) qui présente une collection de fonctions dédiées à l'exécution. Les fonctions du MES sont essentiellement tournées vers les activités de production, principal vecteur de la valeur ajoutée des entreprises manufacturières.

Le concept de MES est né de la forte demande des industriels pour répondre aux exigences des marchés dans une optique de réactivité, de qualité, de respect des normes, de diminution des coûts et des délais. Historiquement bien implanté dans les industries de *process* (les industries pharmaceutiques, agro-alimentaires et des semi-conducteurs, en particulier) où il est une très bonne réponse aux exigences de traçabilité et d'archivage imposées par les réglementations, ce système s'ouvre de plus en plus à tous les secteurs d'activités, notamment aux entreprises manufacturières.

Pendant longtemps, chaque fournisseur définissait lui même ce qu'il entendait par MES et les fonctions qu'un tel outil se devait de couvrir, basant généralement sa définition sur les capacités de son propre outil tout autant que sur ce que son client en attendait. La *Manufacturing Enterprise Solutions Association* (MESA) a donné le premier élan aux travaux de normalisation couvrant le domaine des MES, sigle dont ils sont d'ailleurs les créateurs. Cette organisation regroupe les principaux acteurs du marché et l'un de ses premiers travaux a été de définir de manière générale ce qu'était un MES. Sa définition, qui est maintenant retenue par tous, est la suivante [18] :

> « Le *Manufacturing Execution System* (MES) fournit les informations nécessaires à l'optimisation des activités de production allant de la création de l'ordre de fabrication au produit fini. Par l'utilisation d'informations à jour et précises, le MES guide, initie, réagit aux activités de l'atelier au fur et à mesure de leur déroulement, et fournit des rapports sur ces activités. Sa forte réactivité à l'évolution des conditions de fabrication, associée à un objectif de réduction des activités à faible valeur ajoutée conduit à un fonctionnement plus efficace de l'atelier de production, diminue les stocks et augmente les marges. En alimentant un flux bidirectionnel d'informations, le MES fournit à toute l'entreprise et à sa chaîne logistique, les données critiques sur les activités de fabrication. »

Comme avec les autres systèmes d'information, le MES ne peut se résumer à une simple et unique fonctionnalité. Tout ce qui peut être suivi, dirigé et analysé pour amener la production au meilleur de ses performances et de sa profitabilité a une fonctionnalité dans le MES. Chacune de ces fonctions interagit avec les autres. Le MESA [19] en a identifié 11 qui sont résumées dans la Figure 3 :

- **la collecte et l'acquisition des données** : par sa connexion directe au processus, le MES peut accéder aux données issues des équipements de production, les mettre en forme et les présenter aux différents utilisateurs que sont les exploitants et les systèmes d'analyse et de gestion de la production ;
- **l'analyse des performances** : à partir de ces informations, le MES peut déterminer la valeur des indicateurs de performance prédéfinis (TRS , MTBF, temps d'intervention, indicateurs qualité, ...) et en proposer une analyse en temps réel pour provoquer une réaction automatique et/ou une alarme en direction de l'exploitant ;
- **la traçabilité produit et la généalogie** : le respect des contraintes et des normes imposées par le système d'assurance qualité ainsi que la maîtrise du savoir-faire obligent l'entreprise à assurer la traçabilité d'exécution du processus de fabrication, pour le lot ou pour chaque produit. Cette fonction permet de connaître à tout moment le niveau d'engagement d'un produit ou d'un lot. Elle met à jour un historique qui permet d'en connaître la généalogie ;
- **l'ordonnancement** : par la prise en compte de la disponibilité réelle des équipements et du personnel associé, cette fonction organise les tâches et les ordres de fabrication pour optimiser la productivité de l'atelier ;
- **la gestion des documents** : par la gestion et la mise à disposition de tous les documents utiles à la production ;
- **le cheminement des produits et des lots** : par le suivi des produits en temps réel, à travers les gammes de fabrication en prenant en compte les

aléas de production, cette fonction permet l'optimisation des stocks et la prise en compte de tâches non prévues (maintenance, attente, ...) ;

- **la gestion du personnel** : par le suivi de l'activité et de la disponibilité des opérateurs ;
- **la gestion de la qualité** : par l'échantillonnage en continu tout au long du procédé de fabrication et l'analyse en temps réel des indicateurs qualité, cette fonction identifie les problèmes et en informe les exploitants (en proposant des actions correctives) ;
- **la gestion du procédé** : par l'analyse en continu de l'évolution du procédé, cette fonction identifie les problèmes, en informe les exploitants et exécute ou propose des actions correctrices.
- **la gestion de la maintenance** : par le suivi des machines et la gestion des temps de marche et d'arrêt, cette fonction planifie les tâches de maintenance et met à jour l'historique des actions et des pannes.
- **la gestion des ressources** : par le suivi de l'utilisation et de la disponibilité de l'ensemble des ressources de l'atelier (machines, équipements, personnels et documents).

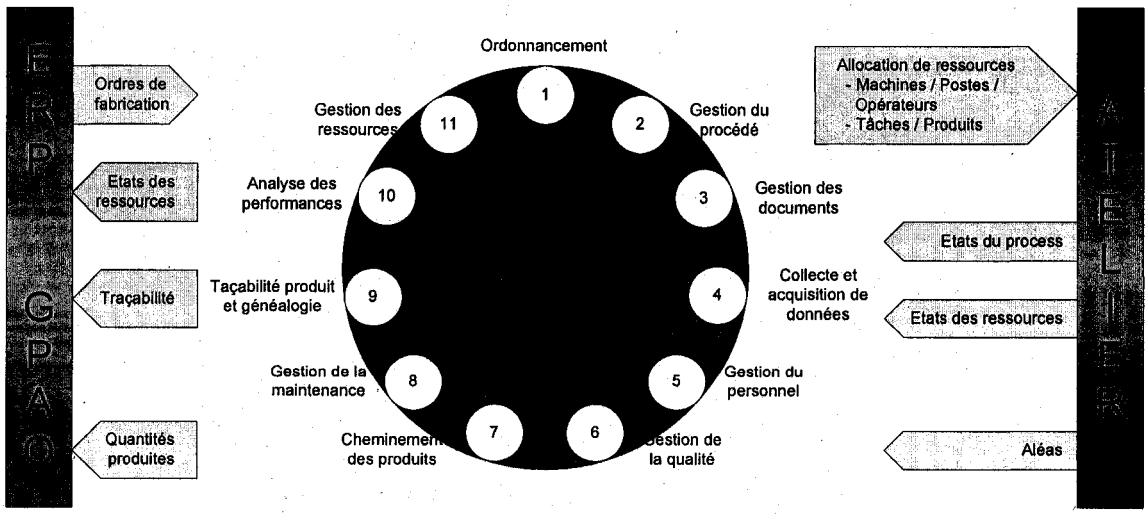


Figure 3 Les fonctionnalités du MES.

Historiquement les solutions MES sont conçues sur une base modulaire. Ainsi, si le panel de fonctions du MES est théoriquement invariable, la priorité entre elles et la configuration peuvent varier et s'adapter à l'entreprise où il est mis en œuvre. Certaines fonctions du MES sont redondantes avec celles des autres systèmes (principalement avec les ERP) même si l'approche du MES est davantage tournée sur les performances de l'outil de production. La norme S95 (basée sur les travaux du MESA) s'attache à la formalisation des échanges autour du système de production vers les autres domaines de l'entreprise. Conçue pour s'appliquer à tous les types de production, elle n'impose pas de modèle d'organisation de l'entreprise ni d'architecture du système de production. Elle suggère toutefois un modèle physique de l'entreprise extrapolé à partir de la norme S88 et une définition des fonctions et des flux d'informations basée sur le *Purdue University Reference Model* (PRM) publié par l'*Instrumentation, Systems & Automation Committee* (ISA). Normalisée par l'*American National Standards Institute* (ANSI), et rassemblant un travail d'analyse des processus industriels et de leur traduction dans le MES, la S95 constitue aujourd'hui une référence reconnue par l'ensemble des acteurs du domaine des MES et des ERP. Au final, la norme S95 cherche à définir l'interface entre les stratégies logistiques et les stratégies de production (Figure 4), entre l'ERP et le MES (Figure 5).

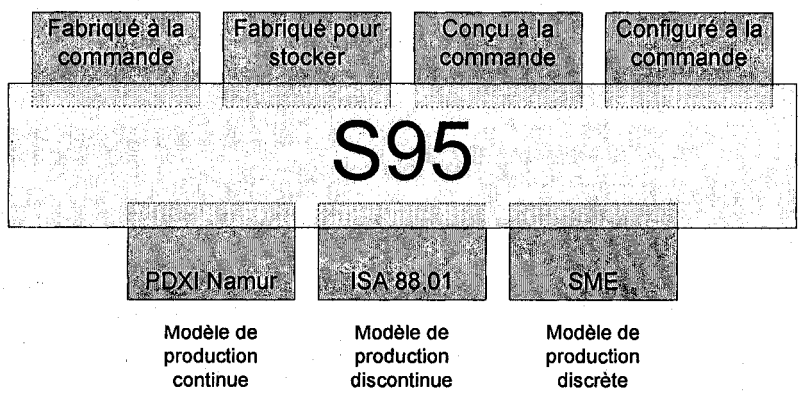


Figure 4 Interface entre les stratégies logistiques et les stratégies de production.

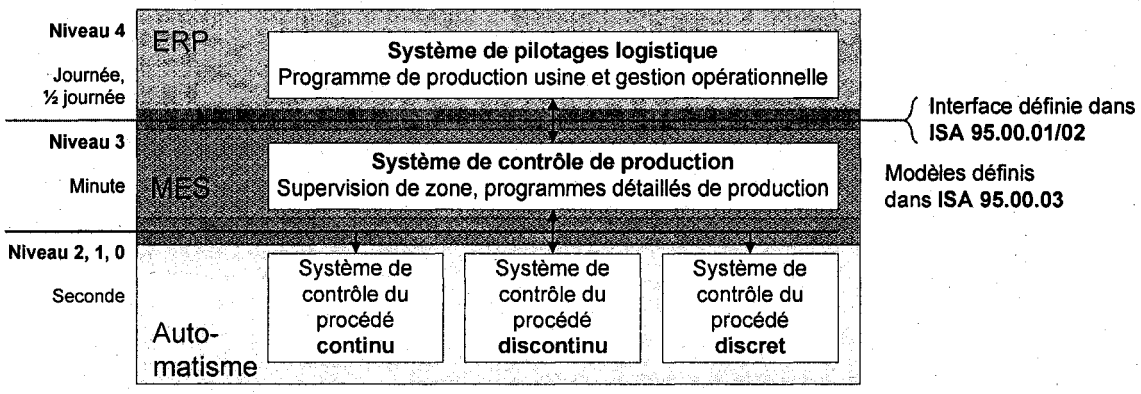


Figure 5 Interface entre l'ERP et le MES.

Comme nous le précisions précédemment, les travaux du MESA définissent 11 fonctionnalités. On remarque tout de suite que certaines d'entre elles sont directement opératoires, comme l'ordonnancement et le contrôle de la qualité, tandis que d'autres apparaissent plutôt comme des fonctions transversales, comme la gestion des ressources et la traçabilité. Cette difficulté de classification des fonctions du MES a deux conséquences. La première est que la notion de MES est souvent difficile à percevoir clairement par les industriels concernés. La seconde est que l'on a tendance à y faire entrer tout ce que l'on n'a pas déjà rattaché à d'autres domaines, comme celui des ERP. C'est cette dernière conséquence qui ne facilite pas sa structuration. En réalité, toutes les fonctions du MES ne sont pas situées sur le même plan.

L'apport fondamental de la S95 est la mise en évidence des interactions indispensables entre les différentes parties d'un système MES. En l'absence de ce système, les solutions mises en place risquent fort de ne pas pouvoir suivre l'évolution des besoins des industriels, voire même de ne pas répondre efficacement au problème posé. Par exemple, la mise en évidence de non-qualités périodiques sur les produits fabriqués ne pourra être interprétée correctement que si elle ne peut être mise en correspondance avec les opérations de maintenance sur les installations. La traçabilité produit est insuffisante si

elle n'est pas couplée à la gestion des équipes, d'une part, et aux informations de contrôle du process, d'autre part.

Le modèle fonctionnel de la S95 met évidemment en position centrale le contrôle de production. Pour l'approvisionnement en matières premières, il interagit directement avec l'ordonnancement et la gestion des stocks. Pour planifier la production, il devra en outre connaître le planning de disponibilité du personnel auprès de la gestion des ressources. En fin ou en cours de production, il interagit avec l'assurance qualité et avec la gestion de la maintenance. Au final, le coût de fabrication sera évalué en liaison avec le service comptabilité. On voit que le MES est un intermédiaire entre les différents services de l'entreprise dont il modélise les flux d'information. La S95 propose en exemple un modèle fonctionnel qui permet d'analyser les flux d'information mis en jeu par le MES et leur traitement. Ce modèle ne préjuge pas de l'organisation effective de l'entreprise selon ce modèle.

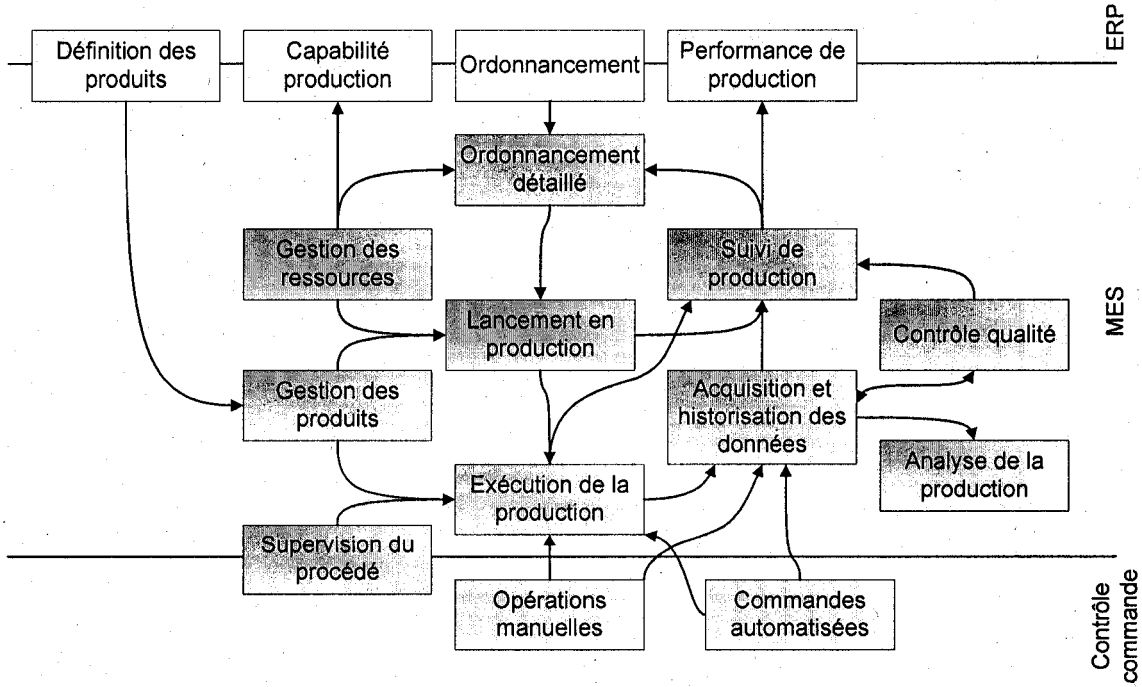


Figure 6 Diagramme fonctionnel S95 (d'après ISA S95)

L'analyse conduite dans la S95 conduit à modéliser les fonctionnalités du MES comme indiqué dans le diagramme de la Figure 6. Elle est devenue une norme largement acceptée par les acteurs du marché pour la conception des flux d'information entre les applications du niveau atelier et celles des niveaux supérieurs. Elle permet aux industriels de disposer d'un périmètre et d'une terminologie cohérente en réponse à leurs besoins.

1.2.2 Technologies

La totalité des offres MES est conçue sur une base modulaire. Le meilleur exemple de cette conception modulaire est le MES InSite™ de Camstar [20] qui se compose de 18 modules intégrés et interactifs permettant au système de suivre et de collecter des données sur les activités de production en continue. De manière générale, ces différents modules peuvent être regroupé en 3 couches [21], [22], [23], [24], [25] :

- *la couche client/serveur* : c'est la couche qui rend visible à l'utilisateur les informations du MES. C'est aussi par l'intermédiaire de cette couche que l'on peut intégrer d'autres systèmes, leur permettant de communiquer dans un environnement distribué ;
- *la couche d'intégration* : c'est le cœur de l'architecture, elle contient des composants standards et réutilisables qui fournissent une infrastructure pour les flux d'information vers ou en provenance d'autres applications. Elle met en œuvre des outils standards tels que CORBA, COM/DCOM, OLE, OPC ;
- *la couche de gestion/stockage des données* : elle fournit des services indispensables pour les autres couches tels que la communication en réseau, les services de gestion d'objets. Ces services sont construits pour des systèmes d'exploitation et des technologies de base de données robustes et standards.

Plus précisément, les fournisseurs de solution MES améliorent leurs solutions sur des plans aussi variés que l'intégration, l'extensibilité, la configuration graphique pour se rapprocher des caractéristiques mises en lumière en 1997 par le MESA [18]. Ces améliorations sont le fruit des rapides progrès de l'informatique et des réseaux (protocoles, Internet, intranet et extranet) et des technologies logicielles (objets, agents, architecture client/serveur, COBRA, ActiveX, XML, XSLT, Web services, connectivité universelle des équipements, langages de programmation). Il est possible de résumer la tendance des améliorations apportées aux MES comme suit :

- l'intégration : la clé de l'implantation d'un système d'information est encore plus vitale au MES qui se veut un pont entre les systèmes de commande de l'atelier et les systèmes de plus haut niveau ;
- la synthèse de connaissances hétérogènes : des mécanismes pour rendre les données collectées plus intelligibles et facilitant les prises de décision pour l'utilisateur ;
- la connectivité standardisée : un modèle de connectivité permettant l'utilisation de l'équipement dès son branchement et fonctionnant sous divers environnements informatiques ;
- les performances en temps réel : la pertinence d'informations telles que les statuts du procédé de fabrication, des matières, du personnel à rendre accessible au bon utilisateur au bon moment ;
- l'architecture orientée vers l'Internet : une architecture modulaire et ouverte pour faciliter l'interopérabilité de solutions hétérogènes, mais aussi anticiper l'adoption de futures technologies ;
- l'extensibilité et la reconfigurabilité : le MES doit répondre aux contraintes des entreprises actuelles, leur taille et leur dynamisme ;
- la collaboration : être un des maillons communiquant d'une chaîne logistique allant des fournisseurs de matières premières aux

consommateurs finaux afin d'améliorer l'efficacité, la flexibilité et le temps de réponse de l'ensemble de la chaîne.

1.2.3 Architectures et modélisations

Dans la littérature scientifique, les efforts se portent principalement sur la proposition de modèles, d'architectures pour un MES polyvalent et performant, qui facilitent son intégration au sein des entreprises. Dans la pratique, cette intégration comporte actuellement certaines limites.

Le schéma organisationnel traditionnel en est l'une des causes. Dans ce schéma , le MES exécute et contrôle les ordres de production générés par l'ERP. Cette structure descendante intègre difficilement d'autres formes de management comme la production en juste à temps ou la production tirée pour laquelle la demande est ascendante [26].

Les technologies des réseaux apportent une évolution aussi bien au niveau des techniques de programmation (architecture client/serveur, travail en parallèle, distribué ou concourant) qu'au niveau de la relation entre l'entreprise et son client (B2B, B2C, *e-business*). Ces technologies sont en train d'être implémentées à tous les niveaux de l'entreprise et ont permis de faire émerger des concepts novateurs tels que l'entreprise agile ou encore l'entreprise numérique [27].

La complexité croissante des produits et des processus nous a poussés à nous poser la question de la pertinence d'un outil totalement centralisé. La technologie des multi-agents a pu alors montrer toute la simplicité d'implémentation, la modularité de cette organisation décentralisée [28], [29]. Les agents peuvent être représentés comme de simples ressources (une machine, une cellule, un outil, un opérateur) qui effectuent des tâches individuelles, qui interagissent entre eux. Une des approches les plus fructueuses formalisées avec l'aide des agents est le système manufacturier holonique [30] où un

holon est l'association d'un agent logiciel et d'un équipement ou d'un ensemble d'équipement. La coopération des holons ne se fait pas ici de manière centralisée ou décentralisée, mais elle reprend une hiérarchie basée sur une décomposition fonctionnelle.

Il est possible de classer les architectures distribuées proposées en deux groupes [31], [32]:

- les structures hiérarchiques où il y a plusieurs niveaux avec des relations du type maître/esclave :
- les structures héterarchique où les agents communiquent sur le modèle du partage, sans relation maître/esclave. Les agents sont localement autonomes et coopèrent au travers de procédures de négociation pour atteindre leur but.

Les avantages de l'architecture héterarchique résident dans sa complexité réduite, sa robustesse intrinsèque, son extensibilité, sa modularité et sa tolérance aux erreurs. Les décisions sont prises localement et uniquement quand cela est nécessaire. Le projet PABADIS (*Plant Automation BAsed on DIstributed Systems*) [33], [34] est financé par l'Union Européenne et a pour objectifs la reconfiguration automatique et dynamique des chaînes de production, basée sur la notion de "plug-and-participate", et la suppression totale ou partielle des tâches de planification et d'ordonnancement, par une organisation décentralisée de la production. Il a eu pour résultat le développement d'un MES basé sur une telle architecture où des agents commandent et contrôlent une production variée dans un environnement changeant d'atelier flexible.

Le projet PABADIS n'est pas la seule initiative de définition d'une architecture pour le MES. Les holons sont aussi à la base d'une architecture MES adaptée à la tendance de la personnalisation de masse [35]. D'autre part, l'industrie des semi-conducteurs, l'un des berceaux et un des acteurs majeurs du MES, est aussi à l'origine de l'initiative

SEMATECH [36] visant une architecture ouverte dédiée à cette industrie. La version originale de cette architecture était surtout focalisée sur le contrôle de l'atelier. Des fonctionnalités telles que la conception de produit et la planification ont été ajoutées [37], [38]. Cette architecture est accompagnée d'outils d'analyse, de conception et de documentation utilisant le langage UML et les réseaux de Pétri [39], [40]. Dans le même ordre d'idée, l'architecture NIIIP-SMART [41] tente de n'utiliser que des composants standards afin de réduire les coûts et les ressources absorbées par la personnalisation des MES, alors que l'architecture OpenMES [42] propose un modèle orienté objet à structure modulaire ainsi qu'une connectivité ouverte avec tous les autres systèmes.

Parmi les modélisations proposées, les auteurs prennent rarement en considération les interactions entre MES, représentant les interactions possibles entre plusieurs ateliers d'une même entreprise ou encore d'une même chaîne d'approvisionnement. Or le travail collaboratif devient de plus en plus courant, tout comme l'externalisation de certaines fonctions. Des termes tels que l'entreprise étendue, les organisations contractantes et l'entreprise virtuelle ont permis de décrire la coopération entre les entreprises. Mais ces alliances ont surtout été étudiées au niveau financier et rarement au niveau des ateliers. Huang [43] propose un modèle d'intégration des MES entre eux basé sur les modes de coopération et les flux d'opération. Mais des limitations importantes apparaissent dues aux modèles de données que peuvent avoir chacun des MES ainsi intégrés.

C'est pourquoi les efforts de modélisation ne se portent pas uniquement sur l'architecture même du MES. Une des clés pour concevoir et implanter un outil qui réponde aux demandes des industriels est de cartographier les besoins en information de tous les niveaux opérationnels et de se baser sur ça pour concevoir une base de données répondant aux besoins spécifiques énoncés. Bien que le troisième volet de la norme S95 s'attache à décrire le modèle de données du MES, certains auteurs n'attendent pas sa parution pour faire des propositions. Par exemple, Bing-hai *et al.* [44] proposent

d'utiliser la méthode de modélisation par entité/relation étendue pour construire le modèle de données du MES.

1.2.4 Connectivités

La demande d'automatisation des processus manufacturiers est toujours élevée et les entreprises font des efforts importants pour adopter ou améliorer les techniques d'intégration et de communication des équipements. Yang *et al.* [45], [46] ont développé un gestionnaire générique d'équipements. Les règles gouvernant ce gestionnaire sont séparées en 3 portions indépendantes : les règles du système, les règles d'affaires et les règles des équipements. Ces règles étaient implantées dans l'interface du MES et dans les pilotes des équipements. Les règles des équipements étant dépendantes de l'équipement, la connexion de machines différentes obligeait à réécrire le pilote. Les auteurs ont alors proposé d'utiliser la technologie d'objet mobile pour développer un pilote d'équipement accessible par l'internet. Le gestionnaire d'équipement télécharge le pilote via un navigateur Internet et il peut dès lors communiquer avec l'équipement par les messages Java ou bien par la technologie des objets Internet avec les protocoles CORBA ou DCOM [47].

Ye [48] et Qiu [49] définissent une connectivité générique pour les équipements comme étant une plateforme capable d'être personnalisée à tout moment. Le support des différents protocoles de communication se fait par l'intermédiaire d'un composant de connectivité de l'équipement et d'un descripteur de ses caractéristiques. L'utilisation d'un schéma basé sur la description des équipements permet d'assimiler des structures de données et des sémantiques différentes dans un contexte où les sources de données sont hétérogènes et permet ainsi de créer des mécanismes d'extraction d'information pertinente pour l'utilisateur.

1.2.5 Filtrage des données

Comme on le comprend bien, les données et tout ce qui les entoure sont au cœur de l'efficacité d'un tel système. Bien avant l'initialisation du flux de matières, le flux de données est en action. Mais si ce flux de données se trouve à être lent et de faible qualité, comment peut-on espérer que le flux de matières soit rapide et de qualité ? S'il est clair qu'avoir la bonne information au bon moment aide à prendre les bonnes décisions et que les technologies actuelles peuvent assurer la disponibilité de l'information, rien ne garantit en revanche qu'elle soit précise et sensée [50]. Si la précision de l'information acquise dépend de la chaîne d'acquisition mise en œuvre, le sens qui lui est attribué ne dépend que de l'analyse qu'on en fait [51], [52].

Les données acquises directement dans l'atelier ont la granularité suffisante au contrôle de ce même atelier. A l'heure actuelle, elles sont largement utilisées dans la maîtrise statistique des procédés et dans l'ajustement de ces procédés [53]. Mais si des systèmes d'information de plus haut niveau désirent utiliser ces mêmes données, elles perdront tout leur sens si elles ne sont pas préalablement filtrées et placées dans le contexte approprié à ces utilisateurs [54], [55], [52].

Une solution pour le filtrage des données peut venir des technologies de standardisation des données comme par exemple le modèle d'objet de document (*Document Object Model*, DOM) [48], [56] ou bien encore de modèles communs de représentation des connaissances [57]. Les règles d'extraction des informations sont dépendantes du module les mettant en œuvre et indépendantes des sources de données. Avec l'utilisation du langage XML [58], l'intégration est alors dynamique et extensible. En y ajoutant l'*eXtensible Style Language Transformations* (XSLT), la configuration des documents transmis, et donc des données, peut être reconfiguré dynamiquement [59], [53].

Une possibilité pour rassembler des données en vue de les analyser est de les extraire depuis une ou plusieurs bases opérationnelles. Les bases de données opérationnelles ou transactionnelles sont conçues pour traiter des transactions rapidement et efficacement : c'est le traitement transactionnel en ligne (OLTP, *On-Line Transaction Processing*). Quand les données ne sont plus utiles pour l'environnement manufacturier, elles peuvent être transférées dans un entrepôt de données pour être exploitées par un environnement d'aide à la décision. La technique de modélisation entité/relation est communément utilisée pour créer les modèles de données utilisés dans les systèmes OLTP. Pour les entrepôts de données, la modélisation dimensionnelle lui est préférée et aboutit à des schémas en étoile ou en flocon de neige [60]. Le traitement analytique en ligne (OLAP, *On-line Analytical Processing*) est une méthodologie basée sur les requêtes qui supporte l'analyse de données dans un environnement dimensionnel. Un moteur OLAP structure logiquement les données sous la forme d'un cube. Par exemple, MS Excel fournit une interface pour de tels traitements. Ainsi, les cubes OLAP et les entrepôts de données du MES peuvent être utilisés pour analyser les données de l'atelier. Chen et Wu [61] proposent ainsi des cubes pour analyser les performances des équipements, les en-cours ou encore la qualité.

Dans le contexte adaptatif, les entreprises ont besoin de détecter les exceptions de façon proactive. Les MES sont capables de le faire uniquement pour les exceptions surgissant dans l'atelier. De plus, les entreprises ont aussi besoin de résoudre ces difficultés en collaborant avec leurs fournisseurs et leurs sous-traitants. Un tel résultat ne peut être atteint par un seul système mais bien en mettant en commun les points forts de chacun des systèmes de l'entreprise.

1.3 Revue des systèmes d'information

1.3.1 Une vue d'ensemble des systèmes d'information

Le MES est maintenant devenu un système d'information majeur implémenté dans les entreprises. Chacune des catégories de système inclut de nombreuses fonctions et types de produits. Les principaux outils dédiés à la production sont aujourd'hui classifiés comme suit (Figure 7) :

- Gestion des relations avec la clientèle ou *Customer Relationship Management* (CRM) – ils ont pour but de créer et entretenir une relation mutuellement bénéfique entre une entreprise et ses clients.
- Progiciel de gestion intégré ou *Enterprise Resources Planning* (ERP) – ils consistent en ces systèmes qui fournissent des plannings financiers, de management des ordres de production, ainsi que les fonctions en relation.
- Système de gestion de la chaîne logistique ou *Supply Chain Management* (SCM) – ils incluent les fonctions telles que les prévisions, la distribution et la logistique, le management du transport, le commerce électronique et les systèmes de planification avancée.
- *Sales and Service Management* (SSM) – ils comprennent les logiciels pour l'automatisation des forces de ventes, les configurateurs de produit, les services d'évaluation, les retours de produits.
- Ingénierie des procédés et des produits ou *Product and Process Engineering* (P&PE) – ils incluent la conception et la fabrication assistée par ordinateur, la modélisation des processus et le management des données concernant le produit.
- Commandes – ce sont habituellement des systèmes hybrides logiciel/matériel tels que les systèmes de contrôle distribué (DCS), les automates programmables industriels (PLC), les contrôles numériques distribués (DNC), les systèmes de contrôles, supervision et d'acquisition

de données (SCADA) et d'autres contrôles de processus informatisés conçus pour contrôler la façon dont les produits sont fabriqués.

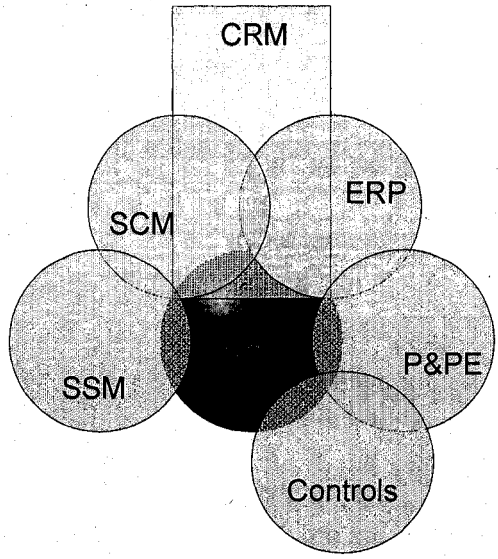


Figure 7 Place du MES par rapport aux autres systèmes d'information

Comme le montre la Figure 7, le MES est lié aux 5 autres catégories de systèmes d'information. Il forme une passerelle entre eux et de son intégration avec ces systèmes dépend le succès de sa mise en œuvre. Comme nous l'avons déjà souligné, les fonctionnalités du MES sont redondantes avec celles de certains de ces systèmes. Par exemple, le MES et l'ERP ont tous deux un module d'allocation des ressources; le MES et le SCM peuvent inclure un module de gestion des stocks; les documents techniques peuvent provenir du MES ou du P&PE; le contrôle qualité, la collecte et l'analyse des données peuvent être dévolus soit au MES soit au contrôle. Les fonctionnalités implémentées dans le MES sont bien plus orientées vers la production, les performances en temps réel et exécutent des instructions plus fines (comparées aux décisions prises aux niveaux supérieurs). Ces fonctions sont conçues d'abord pour être facilement accessibles par les équipements d'une part et par le personnel opérationnel d'autre part [18], [19], [62].

Étant donné la large implantation des systèmes ERP et surtout le lien particulier qui existe entre les ERP et les MES comme le montre la norme S95, intéressons-nous un peu plus à ce système d'information.

1.3.2 Enterprise Resources Planning

1.3.2.1 Historique et définition

Jusque dans les années 1960, les entreprises se concentraient principalement sur les concepts de gestion des stocks et les applications industrielles étaient principalement conçues autour de ce besoin. Pour suivre l'évolution des besoins, ces logiciels se sont orientés par la suite vers la planification des besoins matières (*Material Requirement Planning* ou MRP) dont la finalité est de synchroniser les différents stocks de produits finis, de sous-ensembles, de composants, de matières premières en quantité et dans le temps sur la base d'une demande commerciale réelle et prévisionnelle. À partir de 1975, le MRP évolue, les dimensions industrielles et commerciales sont aussi introduites avec le Plan Industriel & Commercial (PIC), l'adéquation charge/capacité est mieux prise en compte grâce à une boucle de contrôle à chaque niveau de planification. On parle alors de planification des ressources de production (*Manufacturing Resource Planning*, MRPII) [4]. Ainsi, la structure de planification MRPII comprend cinq niveaux de décision et de planification [63], [64] (Figure 8) :

- plan stratégique,
- plan industriel et commercial (PIC),
- programme directeur de production (PDP),
- planification des besoins en composants (MRP),
- pilotage d'atelier.

Cette hiérarchisation des fonctions et des décisions est un principe fondamental des systèmes MRPII. Hiérarchiser les fonctions et les décisions consiste à concevoir une

structure en plusieurs plans, chaque plan étant le lieu d'un ensemble de décisions d'échelle et d'horizon spécifiques. Pour chacun des plans, on identifie un objectif, un horizon et un niveau de détail, et chaque plan est revu avec une périodicité spécifique. À chaque niveau de planification, de multiples itérations ont lieu entre le plan à capacité infinie et la fonction « Gestion des capacités » qui l'évalue au regard des contraintes du système afin d'obtenir un plan réalisable. Ce plan sert ensuite de cadre de décisions pour l'établissement du plan du niveau inférieur.

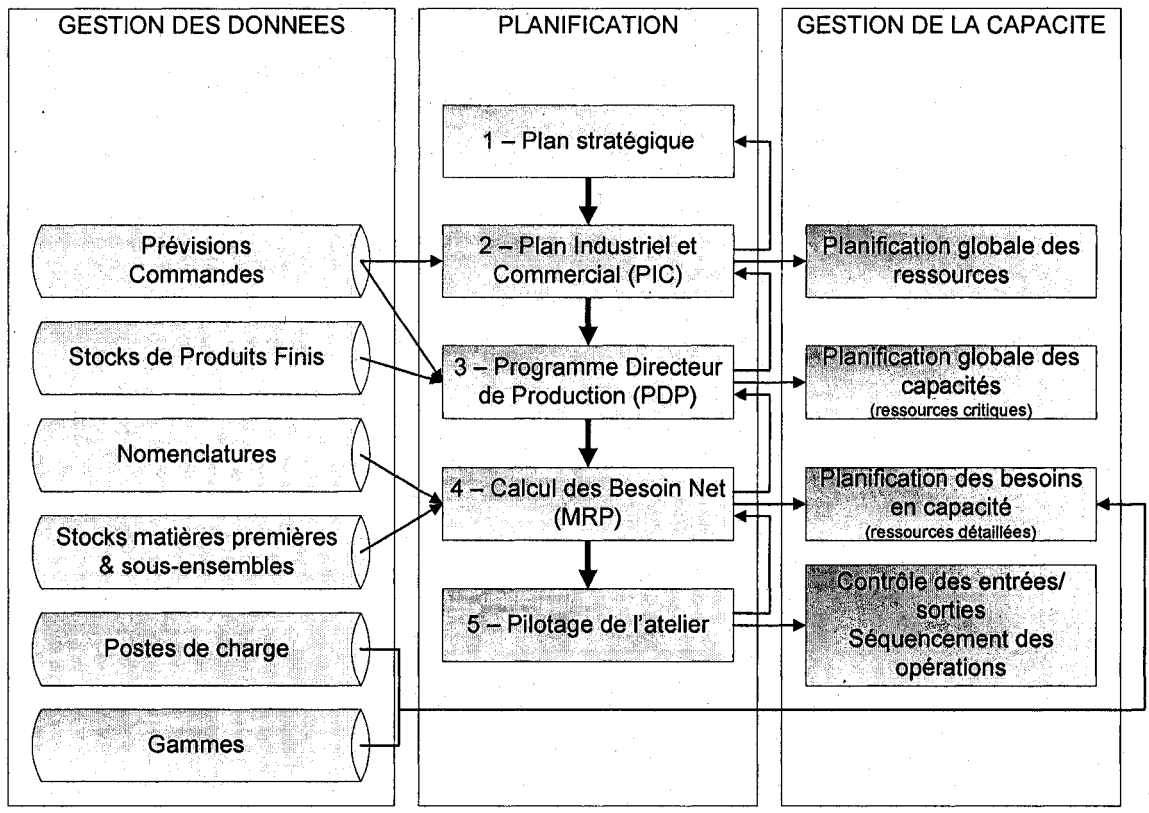


Figure 8 La structure hiérarchique de planification MRPII

Le terme Plan Industriel et Commercial montre qu'il ne s'agit pas uniquement d'une fonction qui produit des plans de production, mais bien une fonction concernée par la coordination des activités critiques de la chaîne logistique . En effet, le PIC établit les objectifs de chacune des fonctions principales, développe le meilleur compromis pour

l'ensemble de la chaîne logistique, inclut les objectifs de ventes/marketing, les coûts et objectifs de production, et les objectifs financiers/de stocks, décline les objectifs stratégiques. Ce plan reflète les stratégies (augmenter la part de marché) et les tactiques (augmenter les stocks pour un service amélioré) qui sont réalisables par la chaîne logistique [64].

Le PDP est un programme établi par anticipation qui précise pour chaque produit fini, les quantités à produire, période par période [4]. Il a une logique de faisabilité. Son rôle est d'adapter la production aux besoins en :

- traduisant la volonté de la direction, exprimée dans le PIC, en forme opérationnelle, c'est-à-dire en quantités à produire ou à acheter ;
- assurant le niveau de service client désiré en agissant sur les niveaux de stock et les programmes de fabrication ;
- affectant les ressources matières, main-d'œuvre et équipements selon le meilleur usage possible ;
- maintenant les immobilisations stockées aux niveaux requis.

L'étape suivante est le MRP. Il utilise le PDP comme point de départ et il éclate la demande en produits finis en besoins dépendants pour chaque composant entrant dans la fabrication des produits finis. Les données utilisées ici sont les nomenclatures, les niveaux de stock et les délais d'obtention. À ce niveau de planification, l'hypothèse de capacité infinie est utilisée.

Dans les années 1990, la nécessité d'intégrer tous ces outils a abouti à l'émergence des progiciels de gestion intégrée, plus connus sous l'acronyme ERP. Ces outils intègrent le traditionnel MRP et toutes ces données maîtresses avec des données relatives aux finances, aux ventes et aux ressources humaines permettant ainsi aux entreprises de mieux évaluer la valeur de leurs produits, de plus rapidement produire des états financiers, de mieux gérer les ressources en hommes, en matières et en argent.

Aujourd'hui le mot clé dans *Enterprise Resource Planning* est bien le mot *Entreprise* car la vraie ambition des ERP est d'intégrer tous les départements d'une entreprise, toutes les fonctions, tous les besoins particuliers de chaque fonction dans un seul système, une seule base de données, favorisant ainsi la communication et les échanges de données transversaux.

L'implantation et la maintenance de tels systèmes est très coûteuse, typiquement entre 15 et 50 millions de dollars [65], [66], [67]. Il ne faut pas s'étonner alors si le marché américain des ERP a été de 25,4 milliards de dollars en 2005 et est estimé à 29 milliards pour 2006 [68]. Le marché bénéficie de l'acceptation répandue que les entreprises ont besoin d'un système d'information intégré pour être compétitives. Le Tableau III présente la répartition du marché entre les vendeurs de solutions ERP en 2005 ainsi que les prévisions pour 2006.

Tableau III

Classement des vendeurs ERP en fonction de leurs revenus de 2005
(source : AMR Research 2006)

2005 Revenue Rank	Vendor	Revenue Share 2004	Revenue Share 2005	Revenue Share Forecast 2006	Growth Rate 2005	Growth Rate Forecast 2006
1	SAP	40%	42%	43%	12%	17%
2	Oracle	10%	20%	23%	110%	29%
3	Sage Group	5%	6%	5%	16%	10%
4	Microsoft	3%	4%	4%	15%	18%
5	SSA Global	3%	3%	3%	7%	3%

Mais malgré ces coûts importants, plus de 80% des entreprises classées dans Fortune 500 utilisent cet outil pour gérer leurs opérations et un nombre croissant de petites et moyennes entreprises et d'administration sont en train d'adopter la même stratégie [3].

1.3.2.2 Les limitations des ERP

Malgré sa large implantation, les ERP souffrent de limitations bien connues. Tout d'abord, les ERP implémentent et utilisent la logique MRP comme fonction de planification centrale. Par conséquent, ce système hérite d'un certain nombre de défauts associés aux systèmes MRP. Ces défauts sont dus à 2 hypothèses critiquables posées par la version originale de la logique MRP :

- les capacités sont supposées infinies ;
- les paramètres sont déterministes (les délais d'obtention principalement).

L'hypothèse de capacité peut être rectifiée par l'ajout de certaines contraintes et des mécanismes en boucle fermée, mais l'inaptitude à se confronter à des situations stochastiques demeure. Le calcul des Besoins Nets généré par le système ne considère pas la disponibilité des ressources simultanément, mais considère que la ressource requise comme une activité à part, séparée des autres [69], [70]. Cela résulte en des replanifications à répétition. Mais en réalité le processus de replanification n'est pas mené à terme, causant alors des inexactitudes dans les dates de lancement. La logique MRP ne prend en compte aucun effet significatif de perte d'efficacité et d'utilisation.

Ces systèmes utilisent aussi des délais d'obtention fixes pour planifier les achats de matières et les dates de production des sous-ensembles et des produits finis. Ils ne tiennent pas compte des incertitudes qui règnent dans l'atelier. Ainsi, quand des pénuries ou des surplus apparaissent, les décisions générées par le MRP ne sont pas exactes [71], [72].

On comprend bien que ce qui pose avant tout problème à une production contrôlée par un ERP c'est l'incertitude faisant partie intégrante de la vie d'un atelier, que cette incertitude se manifeste sous la forme d'événement imprévisible perturbant l'appareil de production [73], [74]. Des études statistiques ont été menées pour déterminer les principales causes d'incertitudes (et leurs interactions) dans un environnement de production contrôlé par un ERP [75]. De nombreuses techniques ont été proposées pour contrer cette faiblesse, mais les stocks de sécurité et les délais de sécurité sont parmi les plus robustes et les plus utilisées [76]. Les heures supplémentaires et la polyvalence sont aussi très utilisées par les gestionnaires [77]. Pour maîtriser l'incertitude sur la demande et la nervosité du système qui en découle, Ho *et al.* [78] utilisent des stocks, des capacités et des délais de sécurité, mais les performances de ces systèmes dépendent de l'environnement manufacturier et une baisse du niveau d'incertitude ne se traduit pas nécessairement par une amélioration des performances. Murthy et Ma [79] ont, quant à eux, développé un modèle mathématique permettant d'évaluer un facteur corrigeant les prévisions pour prendre en compte l'augmentation des niveaux de rebus, alors que Krupp [80] propose un modèle statistique permettant de réévaluer les niveaux des stocks de sécurité mis en place pour palier à l'inexactitude des prévisions. Brennan et Gupta [81] ont appliqué l'analyse de la variance pour examiner les effets de l'incertitude sur les performances des entreprises et montrent qu'elle est affectée par les incertitudes sur la demande et les délais et par leurs interactions.

Une autre faiblesse de la logique MRP vient dans sa définition rigide des flux de matière (au travers des gammes opératoires) et des structures de produits (au travers des nomenclatures). Si un événement imprévu survient dans l'atelier, par exemple un bris de machine, le flux initial aura sûrement besoin d'être redirigé. Mais le système MRP n'a pas de dispositif permettant de suggérer une gamme alternative dans ce cas-ci. Cela se traduira alors par une pénurie en composants et peut être même un dépassement de la date de complétion [82].

Enfin, les ERP actuels sont conçus pour garder trace de toutes les transactions passées dans le système. En effet, le but initial de cette fonctionnalité était de pouvoir suivre l'apparition d'événements et d'en garder une trace en mémoire, mais pas pour aider à la prise de décision. Et même s'il peut avoir des dispositions pour travailler avec les données en temps réel de son environnement manufacturier, les systèmes ERP ont besoin d'un système externe tel que le MES pour pouvoir contrôler et collecter des données en temps réel [7], [83].

1.3.3 Modèles d'intégration des systèmes d'information

L'intégration du MES dans le réseau des systèmes d'information de l'entreprise est la clé de voûte qui rendra les données collectées en temps réel de l'atelier (stocks disponibles, utilisation des équipements, séquence de production en cours,..) accessibles et permettant de prendre des décisions en connaissance de cause. La plupart des systèmes d'information ne sont pas conçus pour travailler avec ces données. Howells [84] observe que les entreprises commencent par implanter les applications financières et poursuivent avec celles des ressources humaines, de la logistique, du marketing, de la chaîne d'approvisionnement, et seulement après tentent d'ajouter à l'ensemble un système lié à l'atelier. À cause du manque de considération de l'intégration au moment de la conception des premiers systèmes implantés, la connexion avec l'atelier, ou tout autre système, se fait difficilement, au moyen d'interface de programmation (API) qui rend le développement coûteux, le système non flexible et difficilement maintenable.

Après le développement de solutions à grande échelle, non ouverte et propriétaire, la définition d'interfaces telles que les BAPI (*Business Application Programming Interface*) de SAP R/3 est devenue une méthode d'intégration courante dans les systèmes hétérogènes. En raison de la nécessité à combiner les meilleurs modules de différents fournisseurs, le concept de composants prédéfinis réutilisables a été développé et peut être appliqué pour intégrer différents systèmes, que ce soit au sein d'une même

entreprise ou bien entre organisations désirant développer leurs coordinations [85]. L'utilisation de ce genre d'interface permet de s'intégrer dans les processus internes du logiciel. Par exemple, les modèles de maintenance qui font l'objet de nombreuses recherches sont souvent intégrés aux ERP. Par exemple, Ip *et al.* [86] ont conçu grâce à la méthodologie IDEF un système de maintenance s'intégrant dans la logique MRPII d'un ERP. Alors que Nikolopoulos *et al.* [87] ont préféré l'approche objet. Rizzi et Zamboni [88] ont développé un module intégré à un ERP permettant d'optimiser le stockage de matières périssables.

Nous avons déjà vu que les agents mobiles étaient utilisés pour modéliser certaines architectures MES comme c'est le cas dans le projet PABADIS. Mais les agents, grâce à leurs capacités à négocier de manière autonome pour réussir à atteindre le but qui leur est assigné, peuvent aussi être utiliser comme vecteur d'intégration et ont déjà été implémenté pour gérer la capacité d'un atelier depuis un ERP [89]. L'intégration de cette méthode est en plus facilitée par les capacités du langage de programmation JAVA qui implémente nativement les agents mobiles et est capable de transformer l'architecture complexe d'un agent en simple fichier texte par exemple, et inversement [90].

Howells [84] met aussi en lumière la réalité des entreprises manufacturières :

- indépendamment du produit fabriqué, un système financier peut être mis en place dans toute industrie ;
- indépendamment du produit fabriqué, les impératifs logistiques sont plus ou moins les mêmes ;
- c'est dans l'atelier que se trouvent les différences majeures.

En effet, les entreprises fabriquent des produits différents avec des équipements différents. C'est là d'ailleurs la source de la complexité de l'intégration de l'atelier. À l'heure actuelle, toutes ces données sont transférées depuis des sources diverses par le

biais de formats et de protocoles de communication variés. De nombreux modèles d'intégration de système d'information passent d'abord par une bonne intégration des données. Par exemple, Wong [91] propose un modèle de données permettant l'intégration d'un ERP et d'un système de gestion du cycle de vie des produits (*Product Lifecycle Management*, PLM). La méthode la plus utilisée pour réaliser une intégration par les données reste l'utilisation du standard XML. L'intégration est alors dynamique, extensible et rapidement reconfigurable [92].

Lobecke et Slawinski [93] rapportent leur expérience d'intégration d'un MES et d'un ERP et démontrent l'importance de la consistance des modèles de données pour intégrer verticalement et horizontalement ces systèmes. Quand les processus d'affaires sont bien intégrés dans le modèle de données, tout le potentiel de l'optimisation peut alors être exploité.

1.4 Revue de la simulation en temps réel

1.4.1 La prise de décisions et la simulation

De nombreux systèmes réels nécessitent une prise de décision à un moment donné, mais comment prendre la bonne décision ? Intuitivement, la bonne décision est celle qui permet d'optimiser un ou plusieurs indicateurs de performance tels que les coûts, le débit, les délais de livraison ou les profits. Par exemple, dans un atelier flexible, la décision à prendre pourrait porter sur le choix d'un parcours parmi plusieurs alternatives [94]. Dans une situation particulière, diriger un élément vers une machine particulière pourrait faire baisser le délai moyen dans l'atelier.

Le champ de la recherche opérationnelle est riche en méthodes d'optimisation. Ces méthodes peuvent être classées en deux grandes catégories : celles qui aboutissent à une solution optimale et les autres, les heuristiques. Toutes ces heuristiques ont été développées en raison de la complexité des systèmes réels. En effet, quand il est encore

possible de modéliser mathématiquement un problème de grande taille, la résolution devient alors soit trop complexe soit trop longue pour permettre d'obtenir une solution dans un intervalle de temps raisonnable. Or les systèmes de décision en temps réel sont constamment pris entre le temps d'élaboration d'une solution et sa qualité. Généralement la qualité d'une solution augmente avec le temps que l'on se donne pour délibérer. Malheureusement, un temps de délibération trop long peut avoir de sérieuses conséquences (sur les stocks par exemple).

Dans les systèmes complexes, il est bien souvent difficile d'évaluer l'impact d'une décision locale sur la totalité du système et sur ses performances. C'est parce qu'un système complexe est susceptible d'impliquer des processus stochastiques (les machines pourraient avoir des temps d'exécution probabilistes), des dépendances complexes parmi ses composants (une machine pourrait alimenter de multiples machines), et un environnement externe incertain (la demande réelle peut varier dans le temps et peut différer des prévisions). Ainsi, prendre une décision dans le cadre d'un système complexe devient encore plus difficile quand on se soumet à des contraintes réalistes. Comme de nombreux systèmes complexes sont difficilement modélisables en utilisant les techniques analytiques tels que la programmation linéaire [95], il est devenu courant de coupler la simulation aux heuristiques pour évaluer les performances d'un système. Ce processus de trouver la meilleure valeur des variables de décision en utilisant la simulation est appelé optimisation basée sur la simulation [96], [97].

La simulation est communément utilisée pour améliorer les systèmes manufacturiers [13], [98], [99]. Il est possible de modéliser des problèmes discrets ou continus, déterministes ou stochastiques de grande taille simplement et cette facilité est encore accrue par les possibilités de réutilisation des éléments d'un modèle. Les modèles de simulation traditionnels sont utilisés hors ligne soit pour améliorer la compréhension du système soit pour évaluer l'impact de différentes stratégies opérationnelles sur ce même système [8]. Ces modèles de simulation ne sont pas conçus pour un usage répété en

temps réel et ils ne sont d'ailleurs jamais couplés aux vrais systèmes d'informations [14].

Les données sont l'une des différences majeures entre les modèles de simulation hors ligne et les systèmes d'aide à la décision en temps réel. Contrairement aux modèles classiques, les systèmes d'aide à la décision en temps réel (DSS, *Decision Support System*) prennent une photo instantanée des statuts réels de l'atelier et prévoient les événements à venir. Dans un environnement changeant aussi rapidement et étant aussi compétitif, les DSS ont nécessairement besoin d'accéder en temps réel aux informations critiques de la production. Les DSS en temps réel se doivent aussi de combiner différentes techniques de résolution avec de multiples sources de données pour élaborer des solutions en un temps compatible avec l'horizon de planification.

1.4.2 L'optimisation basée sur la simulation en temps réel

Les avancées faites dans les dernières décennies dans les domaines de la puissance de calcul et de la capacité des mémoires vives ont rendu possible l'optimisation basée sur la simulation en temps réel. Ce champ de recherche récent offre une grande opportunité pour la simulation et les bénéfices potentiels dans ce domaine sont significatifs. Scott [100] nous donne une vision générale des différentes approches que nous pouvons trouver dans la littérature.

L'aide à la décision proactive dans les problèmes d'ordonnancement des systèmes manufacturiers est un champ d'application typique pour la simulation en ligne ou en temps réel [15]. Kouiss et Pierreval [16] proposent d'associer un MES à un modèle de simulation en ligne pour piloter un système manufacturier flexible. La simulation est ici utilisée pour évaluer différents scénarios proposés par le module décisionnel et prend en compte l'état réel du système de production. La solution retenue est alors proposée à un opérateur humain pour être validée et exécutée. Watt [101] propose une démarche

similaire pour ordonnancer la production dans le domaine des semi-conducteurs. Son modèle utilise un moteur d'inférences qui peut être constamment enrichi de nouvelles règles et faire ainsi l'objet d'une démarche de progrès permanent. Shin *et al.* [102] traitent d'un problème auquel est confronté le personnel des lignes d'assemblage quand une panne survient dans un système à plusieurs lignes de production. Une approche basée sur la simulation à événements discrets est proposée pour aider le personnel des lignes à minimiser la dégradation des flux des lignes de production. Chong *et al.* [103] ont utilisé une méthode en 2 phases pour l'ordonnancement des goulots d'étrangement. Une première phase est utilisée pour détecter et déterminer les goulots et une seconde phase permet de réduire leur chargement en appliquant différentes stratégies. Bagatourova et Mallya [104] proposent de coupler un algorithme de planification du personnel à la simulation pour s'ajuster en temps réel aux changements de l'environnement tels que des fluctuations du débit ou du volume de produits à traiter.

Dans le but de promouvoir la personnalisation et l'amélioration des systèmes, Hwa-Gyoo *et al.* [105] ont développé un simulateur orienté-objet pour les MES. En effet, il n'est pas réaliste d'imaginer qu'un logiciel unique soit en mesure de répondre à toutes les exigences de tous les ateliers de production. Il propose un ensemble générique et personnalisable d'objets permettant de contrôler les MES qui sont, par nature, complexes et variés. D'une manière un peu similaire, Weygandt [106] propose une approche de modélisation orientée objet pour les MES. Il met en lumière les bénéfices d'utiliser un tel outil pour concevoir directement des éléments répondant aux besoins des opérateurs ou du système sans pour autant les encombrer de détails. Étendre les capacités des MES orientés objet est alors facile grâce aux possibilités apportées par des outils de conception graphique.

Morel *et al.* [107] soutiennent que la spécialisation du langage UML pour les entreprises et l'intégration de leurs attentes pourraient faciliter l'interopérabilité entre les modèles résultants de différentes techniques quantitatives ou qualitatives utilisées par les

processus de pilotage des entreprises. D'ailleurs, le langage UML a été utilisé par Cheng et Teng [108] pour développer un modèle contrôlant la communication entre équipements dans l'industrie des semiconducteurs. L'utilisation des langages UML et XML peut résoudre les problèmes d'interopérabilité dans les contextes de systèmes d'informations hétérogènes [109].

Contrairement aux simulations traditionnelles, à cause de la rapidité d'exécution qu'elle nécessite, la simulation en temps réel ne peut pas utiliser une phase transitoire pour affiner les paramètres du modèle. Dans ce contexte, l'initialisation des modèles représente un problème d'importance. Hanish *et al.* [110] proposent deux méthodes d'initialisation. La première méthode utilise une simulation-parent reflétant le système réel pour initialiser et synchroniser des modèles-enfants avec ses prévisions. La seconde méthode consiste à générer de toutes pièces les fichiers qui seront utilisés par le simulateur pour créer le modèle. La seconde méthode est simple à mettre en œuvre ; par contre, la première méthode donne une meilleure réaction au manque de précision des données initiales grâce aux prévisions de la simulation-parent.

Créer un modèle de simulation d'un atelier de grande taille en codant manuellement tous les détails dans le simulateur va non seulement prendre beaucoup de temps, mais va aussi rendre le modèle très difficile à maintenir. Harrison *et al.* [111] proposent donc de séparer les données de configuration du modèle. Un haut niveau d'automatisation peut être atteint et l'utilité de la simulation n'en sera que plus grande. Les données sous-jacentes peuvent être manipulées par les experts du domaine et par la suite transformées en une structure appropriée à l'usage de la simulation. Enfin, XML est utilisé pour transmettre ces données au modèle. Qiao *et al.* [112] vont plus loin. Ils se servent de XML non seulement pour décrire les informations critiques des processus, mais aussi pour configurer et modifier les modèles de simulation et pour échanger des données entre la simulation et les applications externes.

1.5 Les limitations actuelles

Théoriquement, le MES doit être en mesure d'accéder aux données de l'atelier en temps réel (minutes/heures) pour pouvoir contrôler et commander les activités de production [18], [19] . Pour être réactif, il doit non seulement saisir ces données, mais aussi les analyser en temps réel.

Alors que le contrôle des systèmes manufacturiers se fait à l'échelle de la seconde, voire de la milliseconde, le MES prend plusieurs minutes voire plusieurs heures pour leur répondre. En comparaison avec la vieille image de l'atelier où les données étaient trop rarement mises à jour quand il existait des données en provenance de l'atelier, ou bien encore quand les ERP étaient implantés en boucle ouverte et prenaient plusieurs jours pour répondre, on peut comprendre l'utilisation de la notion de « temps réel » des solutions MES actuelles, mais il faut garder à l'esprit que si le système est plus réactif il n'est pas encore en temps réel au sens strict.

En plus de cela, les MES ont de plus en plus de difficulté à coller à la notion de temps réel quand le système de production prend de l'ampleur. En effet plus le système de production est important et plus le MES aura de données à gérer, et d'autant plus si l'automatisation augmente aussi. De plus, la croissance de ces systèmes de production s'accompagne bien souvent d'un accroissement de la complexité du système de production pour permettre de mieux répondre aux attentes du marché. Pour rester efficace, le MES doit alors être mieux intégrés ce qui se traduit par une augmentation de la quantité de données à traiter. C'est un cercle vicieux : plus le MES veut coller à la notion de temps réel, plus il doit être intégré. plus il doit être intégré, plus il doit traiter de données. Et finalement, plus il doit traiter de données, moins il colle à la notion de temps réel.

En plus de toutes ces considérations locales, pour répondre à des problématiques d'amélioration de la productivité multi-sites, le MES doit être en mesure d'obtenir les données sur les processus manufacturiers et sur les produits de toutes les usines géographiquement dispersées. Ces données devraient théoriquement être analysées en temps réel pour répondre à des problèmes tels que : la validation des setups des équipements en fonction de l'ordonnancement, une déviation observée dans la qualité de la production, une réponse dynamique aux demandes dans le cas d'un produit composé d'éléments provenant de diverses unités de production.

Une autre difficulté s'ajoutant à celle du traitement des données, provient de l'hétérogénéité logicielle de l'environnement manufacturier. En effet, il n'est pas rare au sein d'une même solution MES de devoir implanter des modules de provenances variées, chacun avec son modèle de données, ses mécanismes de communication voire même sa propre base de données. Bien souvent, les mécanismes propriétaires sont retenus pour faire fonctionner l'ensemble, ce qui rend le MES difficile à mettre à jour.

L'adoption par les fournisseurs de formats, d'interfaces et d'API propriétaires reste et restera ainsi la principale raison limitant l'intégration de ces outils. C'est ce qui rend l'implémentation, l'intégration et la maintenance des MES si coûteuses. Les grandes entreprises manquent de temps et les petites manquent de ressources pour mener à bien ces projets [113]. Bien que le développement des MES soit une industrie prospère, les coûts élevés de développement et les formats propriétaires freinent l'adoption de ces outils dans nombre d'entreprises. Les MES sont par nature et par définition des outils ouverts à l'intégration et assez standardisés sur l'ensemble du marché. Le problème vient bien du fait que les ERP ont chacun leurs interfaces propriétaires, mais surtout les équipements des ateliers sont innombrables et bien souvent uniques. Les travaux académiques s'intéressant à ces problèmes d'intégration le montrent et tous sont unanimes pour souligner l'importance d'une intégration du MES avec d'autres systèmes d'information. Dès 1996, Westerlund [114] insistait sur l'importance des capacités

d'intégration dans le choix des systèmes ERP et MES car c'est une des clés pour permettre aux entrepreneurs d'atteindre leurs objectifs financiers. En 2001, Rondeau et Litteral [115] réitéraient cet appel. Liu *et al.* [116] décrivent les difficultés d'intégrer un système de planification avancé (*Advanced Planning System*, APS), un ERP et un MES dans le but de pouvoir vérifier les contraintes de capacité des plans de production.

Une solution à ces problèmes d'intégration pourrait bien venir de l'architecture orientée service (SOA, *Service-Oriented Architecture*) : un modèle d'interaction applicative mettant en ouvre des services avec une forte cohérence interne (par l'utilisation d'un format d'échange pivot, le plus souvent XML), et des couplages externes « lâches » (par l'utilisation de couche d'interface interopérable, le plus souvent un service web). Le service est une action exécutée par un « fournisseur » (ou « producteur ») à l'attention d'un « client » (ou « consommateur »). L'interaction entre consommateur et producteur est faite par le biais d'un médiateur (qui peut être un bus) responsable de la mise en relation des composants. Chaque application propose un certain nombre de services et met à disposition une description de ce service. Il suffirait ainsi de mettre à disposition du médiateur tous les services disponibles et de les appeler quand cela est nécessaire. L'intégration en serait alors grandement facilitée. Le seul bémol à cette solution, c'est qu'elle nécessiterait la réécriture complète de toutes les applications actuelles [117].

Pour l'entreprise moderne, les processus sont la forme privilégiée de l'action, ils sont la valeur ajoutée. C'est pour cela aussi qu'elle a fortement investi dans l'intégration de ses processus de gestion grâce à l'ERP. Mais jusqu'à l'émergence de nouveaux besoins en réactivité, en qualité, en adaptabilité, les processus de production étaient délaissés. L'objectif derrière le concept de MES est donc bien l'optimisation des processus de production et des ressources de production. La première étape dans cette démarche d'optimisation est bien entendu la mesure des performances du système actuel. Le MES étant, par sa nature même, fortement connecté aux processus de production, c'est le meilleur outil pour mesurer en temps réel des indicateurs de performance liés à

l'utilisation des matières, à la productivité en fonction des lots, aux arrêts machines... Il permet alors de créer de véritable tableau de bord de pilotage en évaluant des indicateurs tels que le taux de rendement synthétique (TRS), le taux de pannes machines (MTBF), ou le temps de réparation (MTTR), permettant une véritable gestion des actifs. Dès lors, les actions à mettre en place apparaissent comme évidentes. On se trouve ici face à une optimisation suivant le schéma *Connaître*, *Analyser* et *Améliorer*, étapes mettant chacune largement à profit les technologies informatiques, mais nécessitant également chacune une expertise humaine. C'est une première étape dans le support d'aide à la décision, le MES alerte, présente et met en forme les données désirées par l'utilisateur.

Lobecke et Slawinski [93] rapportent leur expérience d'intégration d'un MES et d'un ERP et démontrent l'importance de la consistance des modèles de données pour intégrer verticalement et horizontalement ces systèmes. Quand les processus d'affaires sont bien intégrés dans le modèle de données, tout le potentiel de l'optimisation peut alors être exploité. L'intégration du MES avec les autres catégories de systèmes d'information, si elle est suffisante, pourra permettre aux organisations de répondre rapidement à des événements perturbateurs apparaissant dans l'atelier. Mais les méthodes d'optimisation du MES se basent sur des modèles rigides d'atelier, incapables de reproduire certains événements imprévus [118]. Les capacités d'aide à la décision ont besoin d'être développées.

1.6 Conclusion

Dans cette revue de la littérature nous avons pu voir que le MES est un système dédié à l'exécution de la production. La valeur ajoutée des entreprises manufacturières se créant dans l'atelier, le MES occupe une position particulière dans l'architecture des systèmes d'information. Il fait office de pont entre les différents systèmes d'information, notamment entre le contrôle de l'atelier et les ERP. Mais tous les modèles d'intégration existants ne sont faits que pour répondre à des problématiques ou des produits

commerciaux bien spécifiques. Tous ces systèmes incorporent un nombre plus ou moins important de méthodes d'optimisation. Mais leur rigidité les rend inefficaces à reproduire l'incertitude intrinsèque aux ateliers. La capacité du MES à travailler avec les données en temps réel en fait une source inestimable d'information nécessaire à l'optimisation. La relative facilité de modélisation de la complexité inhérente des ateliers et des organisations actuelles par les outils de simulation en font de redoutables outils d'analyse des performances. C'est donc naturellement que les heuristiques et la simulation ont été intégrées pour permettre d'optimiser en temps réel les performances de ces systèmes complexes. Nous allons voir dans le prochain chapitre comment il va être possible d'intégrer ces différents outils afin de répondre de manière générique aux problématiques des processus adaptatifs, comment il va être possible de fournir une aide à la décision en temps réel réagissant à des événements perturbateurs dans l'environnement manufacturier.

CHAPITRE 2

ARCHITECTURE GÉNÉRIQUE D'EXÉCUTION – SCÉNARIO D'ORDONNANCEMENT RÉACTIF

2.1 Introduction

Jusqu'ici, la littérature scientifique a proposé différents modèles d'intégration entre les systèmes ERP et MES. Alors que ces modèles sont faits pour résoudre certains problèmes d'interopérabilité, ils n'ajoutent pas d'outils d'aide à la décision à l'un ou l'autre des systèmes. La première partie de ce chapitre présente une architecture d'exécution où l'optimisation basée sur la simulation en temps réel vient compléter le traditionnel modèle ERP/MES.

La seconde partie du chapitre est consacrée à la présentation du cas qui nous permettra de tester l'architecture générique. Cet exemple démonstratif est basé sur un scénario de production de l'industrie de l'aluminium. Nous présenterons d'abord les grandes lignes de la production d'aluminium. Nous verrons ensuite qu'elles sont les limites du scénario, quelles hypothèses nous avons posées, à quelles contraintes le système est soumis et quels sont nos objectifs. Nous présenterons le processus actuel d'ordonnancement avant de proposer une architecture réactive basée sur l'architecture générique proposée plu tôt dans ce chapitre.

2.2 Architecture générique d'exécution

2.2.1 Hypothèses

Dans le contexte de la production adaptative, les entreprises ont besoin de détecter les exceptions de façon proactive, de résoudre ces difficultés en collaborant avec leurs fournisseurs et leurs sous-traitants, et d'en tirer les leçons qui permettront l'amélioration

des procédés en temps réel. Les organisations manufacturières adaptatives ont aussi besoin d'une plateforme d'exécution qui connecte les procédés de fabrication au reste de l'entreprise et de la chaîne logistique et qui fournisse une aide à la décision au personnel de l'atelier, leur permettant de livrer tout en respectant les objectifs de performance.

Les exceptions ou les événements perturbateurs peuvent surgir au cœur de l'entreprise ou provenir d'une source extérieure. Le MES, grâce à ses capacités d'analyse en temps réel, est tout désigné pour détecter les événements perturbateurs surgissant dans l'atelier. Suivant les exceptions que l'entreprise souhaite capter, les systèmes entrant dans la composition du modèle peuvent être différents. Par exemple, le mieux placé pour capter des perturbations sur la demande sera l'ERP. Si maintenant, l'entreprise souhaite détecter des exceptions dans ses processus de conception, il faudra probablement prendre en compte un système tel que le PLM. Par soucis de clarté, nous avons fait le choix de ne faire intervenir que l'un de ces systèmes dans l'architecture générique présentée ici, l'ERP. Il est bien entendu qu'il est tout à fait possible de le remplacer par un autre, voir même d'en ajouter sans rien enlever à l'aspect générique de l'architecture proposée.

Intuitivement, une bonne décision peut améliorer un ou plusieurs indicateurs de performance. Dans un système complexe, il est bien souvent difficile d'évaluer l'impact d'une décision locale sur la totalité du système et sur ses performances. Ceci s'explique par le fait qu'un système complexe est susceptible d'impliquer des processus stochastiques (les machines pourraient avoir des temps d'exécution probabilistes), des dépendances complexes parmi ses composants (une machine pourrait alimenter de multiples machines), et un environnement externe incertain (la demande réelle peut varier dans le temps et peut différer des prévisions). Dans un tel contexte, la simulation en temps réel est intéressante, mais elle doit s'intégrer dans le réseau des systèmes d'information en place dans l'entreprise. Elle doit aussi combiner de nombreuses

techniques d'optimisation pour être en mesure de répondre à différents scénarios de prise de décision.

2.2.2 Architecture fonctionnelle proposée

Comme nous le disions plus tôt, le principal objectif de cette architecture est de fournir une aide à la prise décision pour pouvoir réagir aux événements perturbateurs apparaissant dans l'environnement de production. Ces événements peuvent surgir des niveaux supérieurs (un nouvel ordre de production) ou des niveaux inférieurs (un bris de machine). Nous allons présenter les outils liés à l'acquisition, le traitement, le stockage et la partage des données. Ce sont ces outils qui nous permettront de saisir l'apparition des événements. Nous parlerons ensuite du noyau d'aide à la prise de décisions en temps réel qui nous permettra d'être réactif à l'apparition de ces événements perturbateurs.

2.2.2.1 Acquisition, traitement et stockage des données

Les différents composants logiciels peuvent être répartis sur différents serveurs liés par un réseau local ou global. La connexion aux processus réels de production se fait par le biais de clients OPC (*OLE for Process Control*) connectés à différents serveurs OPC collectant en temps réel les données de l'atelier. Ces clients prennent en charge l'acquisition, le stockage temporaire et le traitement des données brutes. En fonction des règles de traitement et de stockage, un client OPC peut stocker les données traitées dans une base de données SQL (*Structured Query Language*), échanger des données grâce à un client XML, ou encore générer des messages à destination des autres composants pour les avertir de l'occurrence d'événements significatifs. Les clients ou serveurs SQL/XML assurent l'échange de données avec les logiciels externes qu'il s'agisse d'un ERP, d'un serveur web, d'une supervision,... Ils peuvent aussi générer des messages pour les autres composants du système dépendamment de leur configuration. La Figure 9 présente ces différents éléments.

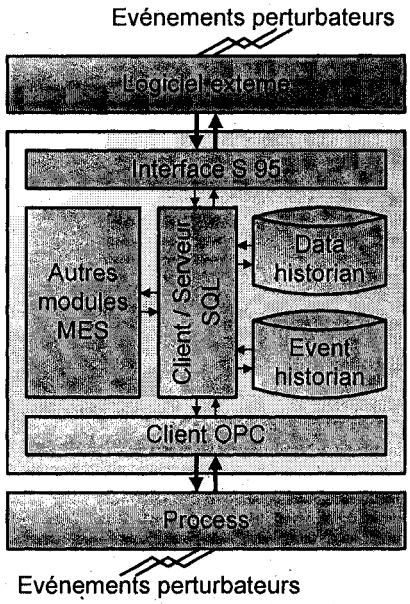


Figure 9 Acquisition, traitement et stockage des données

2.2.2.2 Le noyau d'aide à la prise de décisions

Le noyau de prise de décision de notre modèle est basé sur SDBuilder (*System Dynamics Builder*). C'est essentiellement un hybride entre un simulateur à événements discrets, un système expert basé sur des règles de production et un moteur d'optimisation basé sur la recherche dans un graphe d'état. Les idées fondatrices de SDBuilder ont partiellement été inspirées par la méthode et le simulateur RAO [119], [120]. SDBuilder est basé sur une représentation orientée-objet des éléments du système complexe discret et sur une description des changements d'état utilisés dans les règles de production [121].

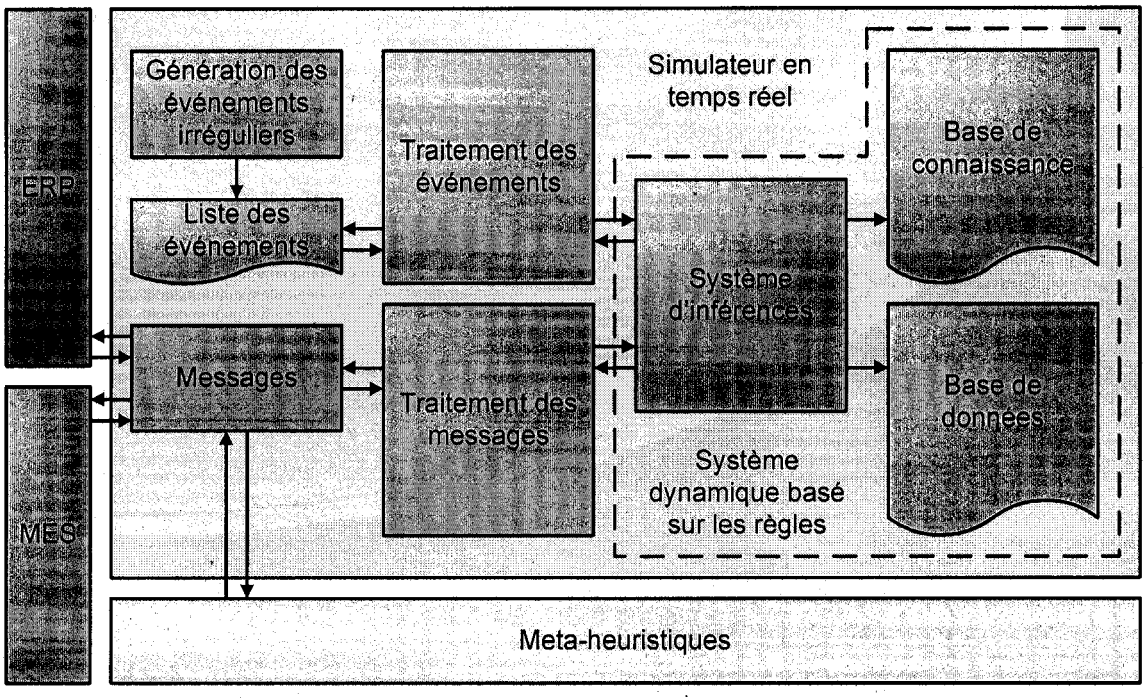


Figure 10 Le noyau d'aide à la prise de décisions

SDBuilder exécute des modèles des différentes parties du système à piloter et répond aux messages entrants en initialisant la logique de prise de décision en fonction de l'événement perturbateur et de l'état actuel du système. Au début du processus de prise de décision, le modèle de simulation est initialisé avec les données de l'ERP (les ordres de production, les séquences de production,...) et du MES (le statut des ordres de production, des machines,...). SDBuilder peut exécuter des modèles de simulation, déclencher le démarrage du moteur d'inférences, initialiser une optimisation utilisant la technique de recherche dans le graphe d'état ou simplement exécuter la règle appropriée de la table de décision. La simulation peut aussi être utilisée en support de métaheuristiques pour évaluer des indicateurs de performance du modèle sujet aux contraintes et aux règles de production. À son tour, il envoie des messages reprenant les décisions prises aux différents acteurs de l'architecture et aux clients OPC qui sont en mesure de directement agir sur le processus. La Figure 10 présente ces différents éléments.

2.2.2.3 L'architecture générique

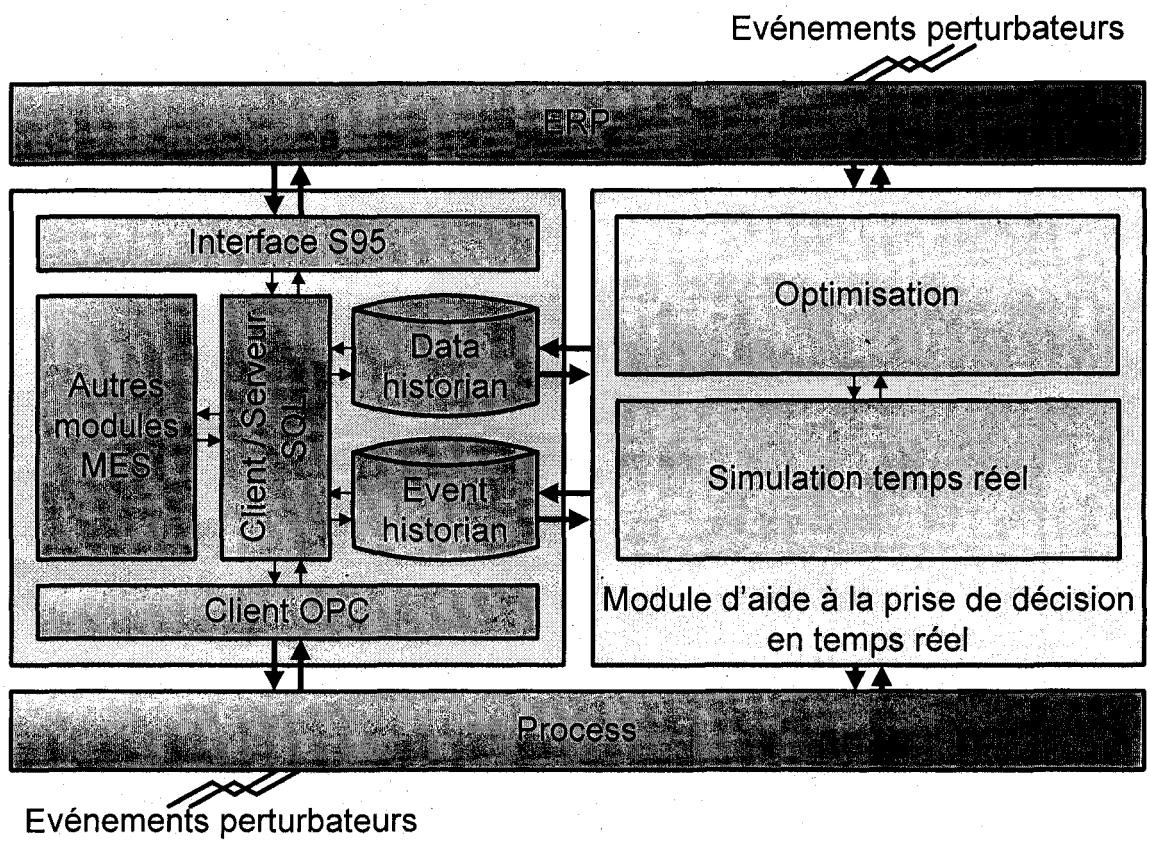


Figure 11 Architecture d'exécution en temps réel

La Figure 11 présente l'architecture d'exécution au complet, reprenant les différents composants présentés auparavant. Nous rappelons que le système ERP présent sur cette figure peut être remplacé par un tout autre système dépendamment des événements que l'entreprise souhaite capter.

D'autres modules ou d'autres systèmes peuvent être intégrés à cette architecture pour rapporter, surveiller, expédier des courriels par exemple. Une interface utilisateur est

aussi nécessaire pour pouvoir configurer la composition du système, les adresses des différents modules, les données à acquérir, le type et le format des messages, et bien entendu les événements à surveiller. Cette interface est aussi utilisée pour développer les modèles d'optimisation, les modèles de simulation et définir les indicateurs de performance.

L'architecture dans cette première partie supporte la prise de décision en temps réel. Mais les plus beaux concepts restent vides de sens s'ils ne sont pas confrontés à la réalité de l'expérimentation. Nous allons présenter l'exemple qui nous permettra de tester notre architecture générique. Cet exemple démonstratif est basé sur un scénario de production de l'industrie de l'aluminium.

2.3 Scénario d'ordonnancement réactif

2.3.1 Processus de production de l'aluminium

Avant toute chose, il est important de se familiariser avec le contexte de la production d'aluminium. Cette production comporte deux phases principales, soit l'extraction de l'alumine par la pulvérisation de la bauxite qui la contient et la transformation de l'alumine en aluminium par réduction électrolytique. La Figure 12 présente les grandes lignes de ce processus de production.

L'alumine extraite de la bauxite constitue la principale matière première utilisée pour obtenir de l'aluminium. Elle provient de différentes sources réparties à travers le monde. Les autres matières premières sont le coke de pétrole, le brai liquide et, bien entendu, l'énergie électrique.

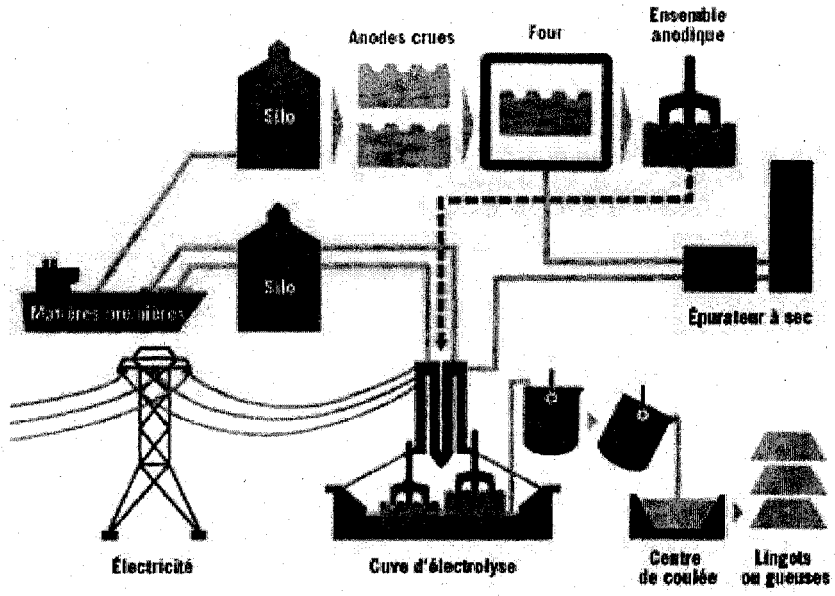


Figure 12 Processus de fabrication de l'aluminium (source : Aluminerie Alouette)

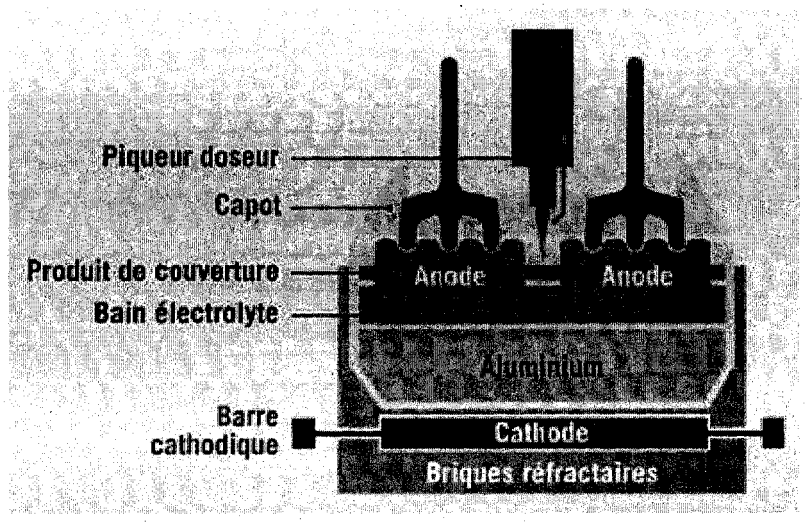


Figure 13 Coupe d'une cuve (source : Aluminerie Alouette)

L'étape suivante consiste à fabriquer les anodes d'alumines. Ces anodes sont formées à partir d'un mélange de coke de pétrole et de brai liquide. Elles sont ensuite durcies à 1100 °C et l'on y fixe des tiges pour faciliter leur manipulation. Par la suite, les électrodes d'alumine sont plongées dans les cuves qui renferment un électrolyte en fusion qui dissout l'alumine. Les cuves en acier sont revêtues de briques réfractaires et de blocs de carbone qui forment la cathode (Figure 13).

Lors du procédé d'électrolyse, l'alumine est décomposée en aluminium et en oxygène à une température d'environ 960 °C. Les anodes consumées sous l'effet de la réaction chimique sont remplacées après leur vie utile qui peut atteindre un mois (processus continu). Le reste de l'anode, appelé mégot, est recyclé pour en produire de nouvelles. L'aluminium en fusion se dépose au fond de la cuve d'où, à intervalle régulier, il est siphonné dans des creusets (processus discret). Les creusets contenant l'aluminium liquide sont transportés vers le centre de coulée. Dans des fours prévus à cet effet, l'aluminium est mélangé à différents composants pour produire les alliages souhaités. Finalement, l'alliage sortant des fours de mélange est transféré vers les lingotières qui assurent la production des lingots.

2.3.2 Périmètre, hypothèses, contraintes et objectifs du système

2.3.2.1 Périmètre du système et hypothèses

Dans l'exemple traité ici, nous ne considérons que la partie finale du processus, c'est-à-dire que nous nous intéressons au processus à partir des cuves d'électrolyse jusqu'au centre de coulée (Figure 14).

Notre salle des cuves a une capacité de 240 cuves qui sont individuellement vidées toutes les 24 heures. Chaque cuve a des propriétés chimiques et un horaire de vidange qui lui sont propres. Les propriétés chimiques sont dépendantes de l'environnement dans lequel s'effectue la réaction d'électrolyse et de la qualité des réactifs. Ces propriétés

peuvent changer au cours du temps et sont mise à jour par des analyses en laboratoire. Dans notre scénario, le taux de contamination de l'aluminium sera la seule propriété que nous prendrons en compte. Les cuves ayant un fort taux de contamination ne pourront pas être utilisées pour fabriquer les alliages de haute qualité qui ont la valeur marchande la plus élevée.

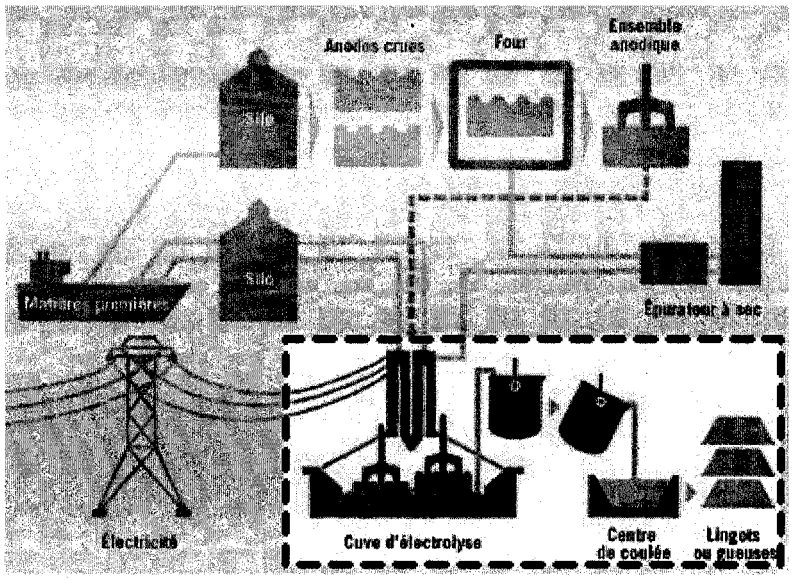


Figure 14 Périmètre de l'étude

2.3.2.2 Contraintes et objectifs

Quand une cuve est pleine, l'aluminium en fusion est vidé dans un creuset pour être transporté vers le centre de coulée où les alliages sont assemblés avant de les couler en lingots. Le métal en fusion arrive au centre de coulée dans une séquence incontrôlable qui dépend des horaires de vidange des cuves et de la qualité de leur contenu.

Des fournées d'alliage sont produites dans le centre de coulée. Comme le métal arrive en une séquence incontrôlée, il est possible de vider les creusets dans un four d'attente. Les

propriétés chimiques de ce four d'attente varient en fonction des propriétés des creusets qui sont vidés dedans.

Les opérateurs de ce système manufacturier ont plusieurs objectifs. Au niveau global, l'objectif est d'optimiser l'allocation des alliages entre les différentes fonderies de l'entreprise en fonction de leur charge actuelle et de leur capacité. Au niveau des fonderies, l'objectif est d'optimiser le planning quotidien dépendamment des conditions réelles de la fonderie. Dans le cadre de cet exemple, nous nous limiterons à cet unique objectif. La mesure de performance retenue compare la qualité du métal utilisé par rapport à la qualité de l'alliage vendu. Globalement, l'objectif du personnel de la fonderie est de restreindre tant que possible l'utilisation de métal de haute qualité à la production d'alliage à haute valeur ajoutée et de minimiser l'utilisation du métal de haute qualité dans les alliages de plus bas calibre.

2.3.3 Processus adaptatifs

2.3.3.1 Processus actuel

Une des premières étapes dans toute modélisation consiste à observer le système existant avant de proposer des améliorations. C'est ce que nous allons faire ici. La Figure 15 cartographie le processus d'ordonnancement actuel. Dans ce scénario, les ordres de production sont répartis dans chaque fonderie par un planificateur général. Quand une fonderie reçoit un nouvel ordre de production, l'ordonnancement est mis à jour en fonction des statuts de l'atelier. Suivant les propriétés de l'aluminium qui arrive, les ordres peuvent être ou non complétés. C'est pourquoi à la sortie du centre de coulée les statuts des ordres sont aussi mis à jour. Si l'ordre n'est pas complété, il faut le réordonnancer. S'il est complété, il peut être confirmé et la marchandise peut être expédiée. Dans l'état actuel des choses, c'est le responsable du centre de coulée qui décide de l'utilisation des cuves d'aluminium.

Dans le cadre d'un processus manufacturier, l'entreprise désire s'adapter à l'occurrence de certains événements. Les occurrences des événements conduisant à un ré-ordonnancement de la production sont les suivants : l'arrivée d'un nouvel ordre et la sortie d'un ordre du centre de coulée (ordre complété ou non). Le modèle d'optimisation sur la simulation devra pouvoir répondre à ces deux éventualités.

La mesure de performance utilisée dans le processus actuel est le niveau de service mesuré comme la quantité d'alliage livrée en temps et en heure par rapport à celle qui aurait dû être livrée. On se rend compte que la mesure de performance actuelle ne tient pas compte de l'utilisation qui est faite de l'aluminium de haute qualité.

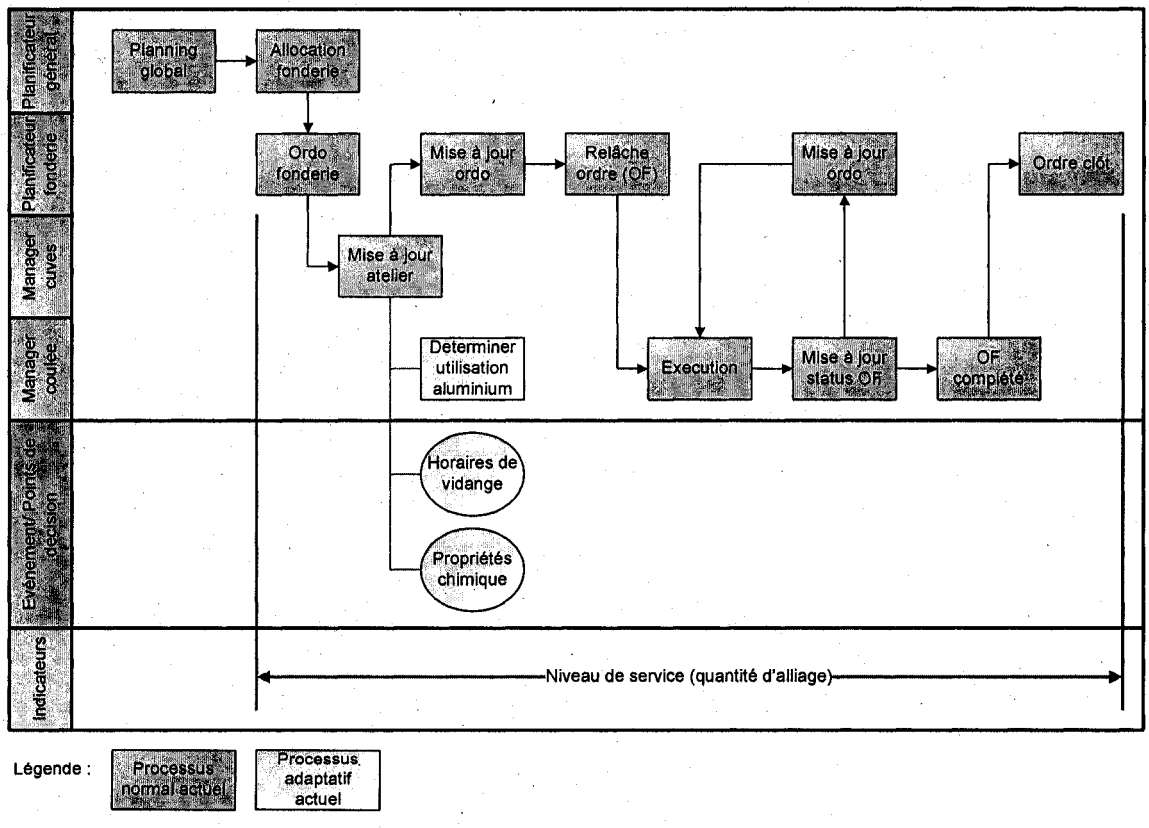


Figure 15 Processus actuel d'ordonnancement

2.3.3.2 Processus adaptatif proposé

L'entreprise souhaite que son processus d'ordonnancement réagisse à l'arrivée d'un nouvel ordre et à la sortie d'un ordre du centre de coulée.

Comme nous pouvons le constater, à aucun moment du processus décisionnel l'utilisation d'aluminium de qualité supérieure dans des alliages à faible rendement commercial n'est prise en compte. Or, c'est ce déclassement d'aluminium de haute qualité qui coûte cher à l'entreprise. Le coût des matières premières étant ce qu'il est actuellement, une utilisation au mieux des propriétés de l'aluminium est primordiale. Le nouveau processus d'ordonnancement sera donc évalué en fonction des trois critères suivants. Le premier critère est le pourcentage volume déclassé (Formule 3.1). Le second critère compare les quantités réellement produites aux quantités planifiées, il s'évalue sous la forme d'un pourcentage (Formule 3.2). Le dernier critère est plus classique puisqu'il s'agit du délai de production réel (Formule 3.3). On trouvera dans l'Annexe 2 un exemple de fiche synthétique de ces indicateurs.

$$C^{VD} = \frac{\sum_{ordres} \text{Volume d'aluminium déclassé}}{\sum_{ordres} \text{Volume d'aluminium utilisé}} \tag{2.1}$$

$$C^{PP} = \frac{\sum_{ordres} \left|\text{Volume d'alliage produit - volume d'alliage planifié}\right|}{\sum_{ordres} \text{Volume d'alliage planifié}} \tag{2.2}$$

$$C^{D} = \text{Date de début - Date de fin} \tag{2.3}$$

Dans le nouveau processus, quand se présente l'occurrence d'un événement déclencheur du processus adaptatif, la première des tâches à effectuer consiste à obtenir les statuts de l'atelier. Cela signifie qu'il faut connaître :

- pour les cuves et les creusets : le contenu, leurs propriétés chimiques et l'heure de vidange ;
- pour le four d'attente : le contenu et les propriétés chimiques du four ;
- pour les fours du centre de coulée : le contenu, le numéro d'ordre traité, le temps restant avant la fin du traitement de l'ordre.

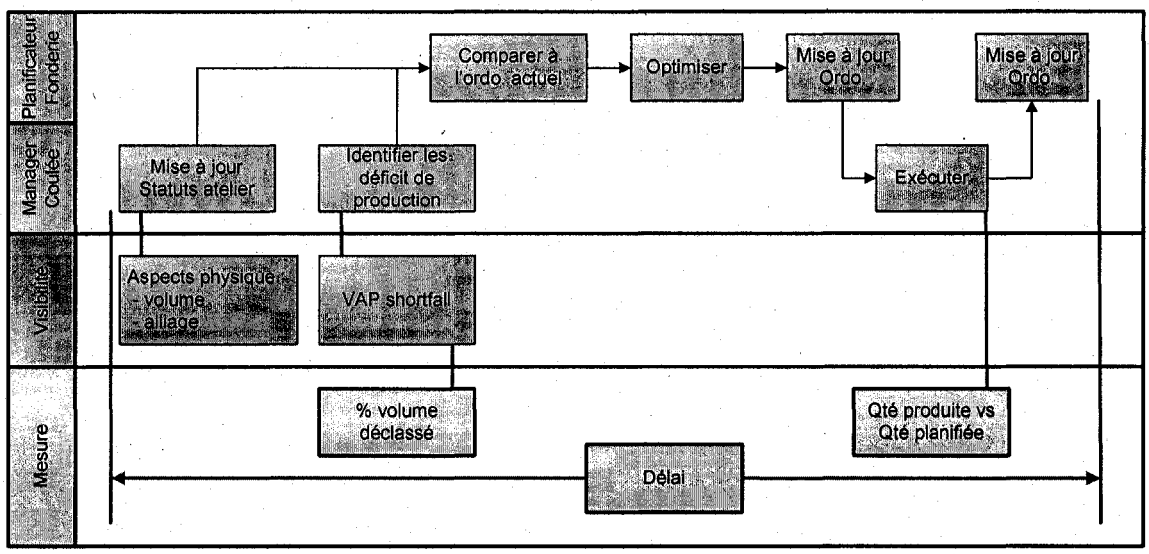


Figure 16 Nouveau processus adaptatif

Dans le cas où l'événement déclencheur correspond à la sortie d'un ordre de production du centre de coulée, il faut aussi connaître le déficit de matière existant par rapport à l'ordre planifié dans l'ordonnancement actuel. Quand toutes les données de l'atelier sont connues, l'optimisation de l'ordonnancement est lancée. À la fin de cette étape, on obtient une séquence de production mise à jour. Soit cette séquence intègre le nouvel ordre, soit l'ordre est mise à jour en fonction de l'état de l'atelier. Quand la nouvelle

séquence est connue, la production se poursuit et le processus se répétera à la prochaine occurrence d'un événement déclencheur. La Figure 16 présente le nouveau processus adaptatif mis en place dans le cadre de ce scénario.

2.3.4 Architecture proposée

2.3.4.1 Architecture systémique

Maintenant que le nouveau processus adaptatif est connu il est possible de proposer l'architecture système qui le supportera et d'affecter les tâches au système le plus apte à la mener à bien.

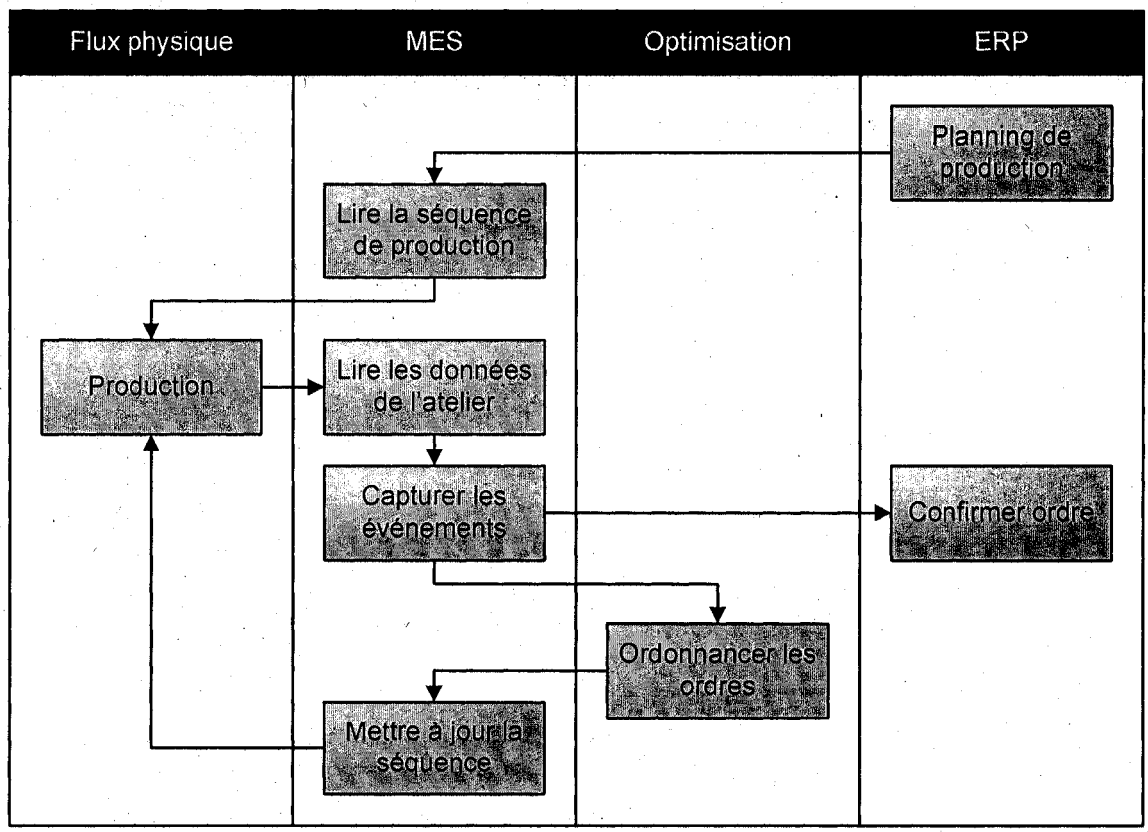
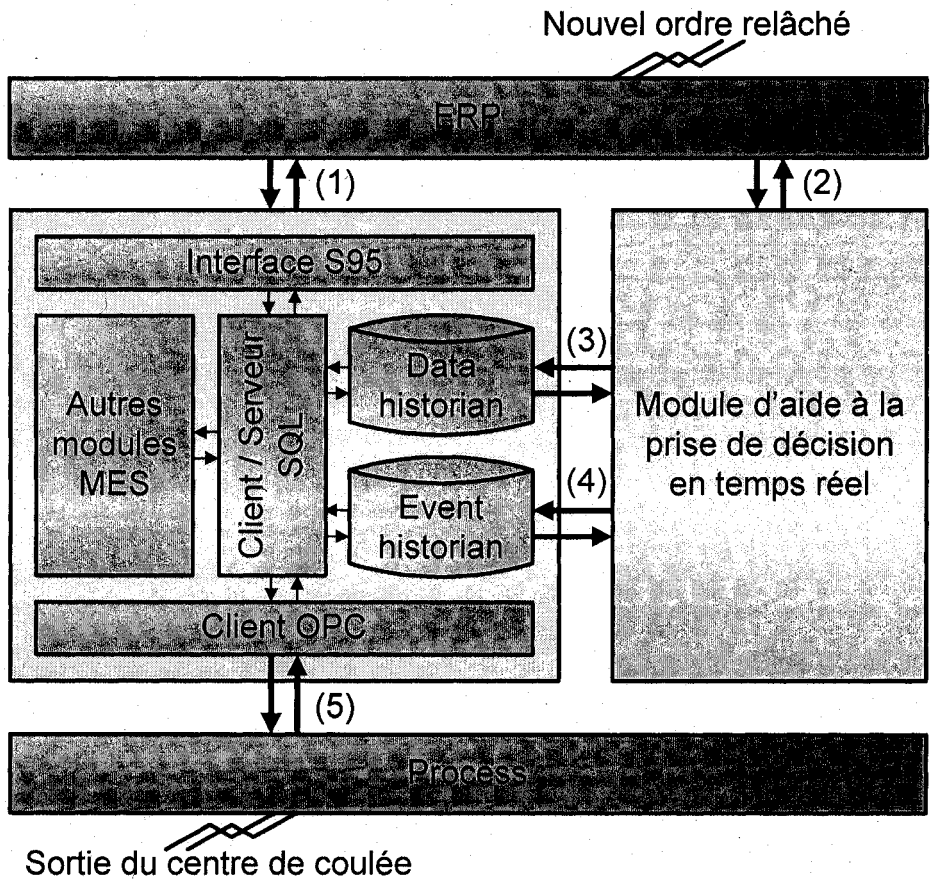


Figure 17 Déroulement du processus adaptatif du point de vue des systèmes



(1) : ordres de productions relâchés, confirmations des ordres

(2) : ordres de productions relâchés

(3) : statuts actuel de l'atelier

(4) : sortie du centre de coulée

(5) : données de l'atelier en temps réel

Figure 18 Architecture supportant le processus adaptatif

Les événements déclencheurs étant l'arrivée d'un nouvel ordre de production ou la sortie d'un ordre du centre de coulée, le meilleur système pour les capter est le MES. La détection de la sortie d'un ordre du centre de coulée est rendue possible par sa connectivité avec l'atelier, alors que la détection des nouveaux ordres relâchés l'est grâce à l'intégration entre le MES et l'ERP. L'ERP doit aussi faire partie de l'architecture, car c'est dans ce système que sont créés les ordres de production et c'est

aussi là qu'ils doivent être confirmés. Enfin, il est nécessaire d'intégrer un module d'optimisation dans l'optique d'ordonnancer au mieux la séquence de production. La Figure 17 présente le déroulement des opérations d'un point de vue des systèmes. On y retrouve les trois systèmes que l'on vient d'identifier et les tâches qui leur sont affectées. La Figure 18 présente l'architecture adaptative retenue pour ce scénario. En plus des trois systèmes intervenant ici, elle contient une description sommaire des données échangées. Les détails seront précisés par la suite, dans le chapitre 3, 4 et 5.

2.3.4.2 Choix des systèmes

Le lecteur remarquera que jusqu'à présent nous avons uniquement parlé des systèmes de manière conceptuelle. En effet à ce moment de l'étude, le scénario est indépendant de l'offre commerciale retenue. Dans un cadre plus réaliste, l'étude fonctionnelle préliminaire conduirait à la rédaction d'un cahier des charges permettant de cibler les besoins de l'entreprise en vue de l'implantation du processus adaptatif. Le choix des applications commerciales se ferait alors très certainement par un processus d'appel d'offres et de consultation des différents vendeurs.

Dans le cadre de ce scénario, les solutions logicielles retenues sont :

- SAP R/3 version 4.6c comme ERP;
- Proficy comme MES;
- SDBuilder comme simulateur.

En effet, SAP est un premier rang des vendeurs d'ERP avec 43% des prévisions des revenus du marché en 2006 (Tableau III). C'est une solution très largement implantée dans les grandes entreprises au Québec et dans le monde. Proficy™ est le MES de chez GE Fanuc. C'est aussi un des acteurs majeurs dans son marché et un des plus grands intégrateurs de Proficy est une société québécoise (STI à Trois-Rivières). SDBuilder a

été retenu pour les possibilités et la souplesse de modélisation qu'il apporte, son moteur d'inférences et la simplicité à l'intégrer dans un processus d'optimisation.

2.3.4.3 Connectivité

Dans le processus adaptatif proposé, c'est trois systèmes vont devoir communiquer ensemble et s'échanger des données. La Figure 18 présente sommairement les données échangées. Les difficultés de l'intégration ont été évoquées dans la revue de littérature et nous avons pu constater qu'il n'y pas de mécanisme simple et standard pour faire communiquer les systèmes de vendeurs différents. Nous nous retrouvons ici dans une situation similaire et nous n'allons pas avoir d'autre solution que de développer les outils qui nous permettront de réussir l'intégration des systèmes retenus pour notre architecture.

Heureusement, il y a certains avantages d'avoir choisi de travailler avec des outils fortement répandus. En effet, nous ne sommes pas les premiers à être confrontés à ces problèmes et certains outils ont déjà été développés et mis librement à la disposition des intégrateurs. Ainsi, pour communiquer avec SAP R/3, SAP met à disposition un connecteur Java (JCo) qui permet d'appeler depuis un système externe, non SAP, un certain nombre de fonctions et de méthodes utilisées par les objets de SAP (BAPI). De la même façon, le standard OPC est très souvent utilisé pour connecter le MES à l'atelier. Les MES comportent donc au moins un client et un serveur OPC. Les interfaces OPC, comme nous le verrons, ont l'avantage d'être indépendantes de l'offre commerciale, car elles sont maintenues par un organisme indépendant. Pour cette raison, nous avons retenu cette solution pour interfacer le MES et le module d'optimisation. En plus de gagner en indépendance vis-à-vis du MES utilisé, il serait même possible de s'interfacer direct avec l'atelier pour récupérer les statuts du processus de production. La Figure 19 récapitule les différents éléments de connectivité intégrés dans l'architecture proposée.

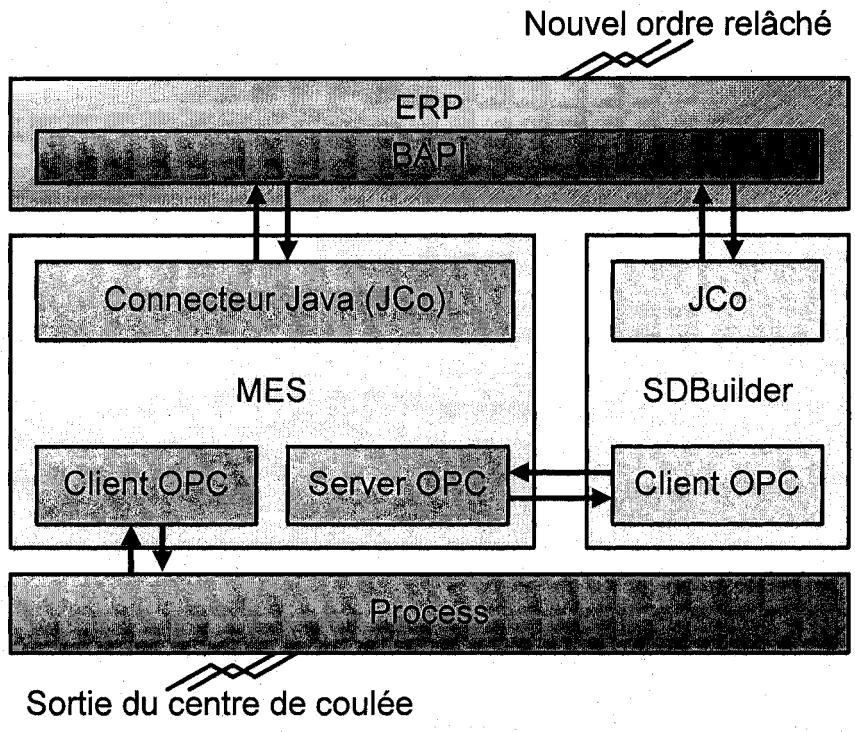


Figure 19 Connectivités de l'architecture proposée

2.4 Conclusion

L'architecture ouverte développée dans ce chapitre supporte la prise de décision en temps réel. Ce système de contrôle en temps réel intègre des méthodes d'optimisation et de simulation avec des systèmes ERP et MES. Les fonctionnalités et les interactions entre les différents composants de ce système d'exécution en temps réel ont été présentées. Mais les plus beaux concepts restent vides de sens s'ils ne sont pas confrontés à la réalité de l'expérimentation. Il est temps de passer à un exemple concret.

L'étude présentée dans ce chapitre présente le travail préliminaire indispensable dans la mise en place d'un scénario adaptatif, de la cartographie du processus actuel à la

conception du nouveau processus, en passant par l'identification des événements perturbateurs et la proposition d'une architecture système. Un des intérêts de cette étude est de pouvoir rédiger un cahier des charges plus précis et de retenir les offres qui répondront le mieux à l'utilisation que l'entreprise souhaite en faire. Dans la réalité, il sera fréquent que l'entreprise possède déjà un ou plusieurs des systèmes mis en jeu. Dans ce cas, l'étude préliminaire devra en tenir compte dès le début afin de limiter les efforts d'implantation et de maintenance.

La dernière partie du chapitre présente les solutions commerciales retenues pour notre scénario ainsi qu'une présentation succincte de la connectivité qui sera utilisée. Les chapitres suivants vont préciser les détails de l'intégration entre les différents systèmes, des données échangées ainsi que les éléments que nous avons développés pour faire de cette architecture conceptuelle une réalité.

CHAPITRE 3

SCÉNARIO D'ORDONNANCEMENT RÉACTIF - ERP

3.1 Introduction

Après les concepts indépendants des solutions commerciales, nous allons voir les éléments de configuration de l'ERP, les éléments organisationnels et les données maîtresses. Nous verrons ensuite les composants que nous avons développés pour permettre l'intégration de l'ERP avec les autres systèmes de l'architecture proposée pour supporter le processus adaptatif. Nous aborderons les notions de BAPI, de modules d'appel de fonction distante (*Remote Function Call*, RFC). Nous présenterons enfin le connecteur Java de SAP (JCo) nécessaire afin de pouvoir utiliser les BAPI et les RFC depuis un système externe.

3.2 Structure de données

3.2.1 Éléments organisationnels

L'entreprise de notre scénario fabrique ses alliages à la commande. En effet, les clients de l'aluminerie ont chacun des besoins différents en fonction de l'utilisation qu'ils vont faire de l'alliage. Nous allons présenter ici les éléments organisationnels configurés dans l'ERP. Ces éléments organisationnels sont représentatifs de la structure réelle de l'entreprise. Le but principal de ces éléments est de segmenter les données transactionnelles qui seront collectées par le système. Certains éléments, comme le code de compagnie, sont requis par les normes financières, car ils sont nécessaires pour retracer l'imputation des revenus et des coûts. Il est à noter que ces éléments sont purement logiques, c'est-à-dire qu'ils représentent des concepts logiques et virtuels. Par exemple, il peut y avoir plus d'un stock à une même adresse physique pour permettre

différentes approches d'estimations. D'un autre côté, le stock en transit peut être mis dans un stock virtuel qui n'aura pas d'adresse physique.

La Figure 20 présente la modélisation de l'aluminerie que nous avons mise en place dans l'ERP. Pour plus d'information, le lecteur peut se référer à [122]. Nous avons essayé de simplifier au possible la partie financière qui n'est pas ici notre point d'intérêt. Le code d'entreprise (ZME1) est la plus petite entité organisationnelle prise en compte au niveau financier, l'aire de contrôle (ZBA1) est utilisée à des fins comptables. L'usine (ZPL1) est l'aire opérationnelle, là où les matières sont transformées. Les stocks servent pour les matières premières (ZS1) et pour les alliages (ZS2). L'organisation des achats (ZPO1) est responsable de toutes les activités touchant aux achats. L'aire de vente sert à segmenter l'organisation des ventes en fonction des conditions de vente. Elle est composée d'une organisation des ventes (ZSO1) qui est responsable de la vente des alliages et de canaux de vente (Z1 et Z2).

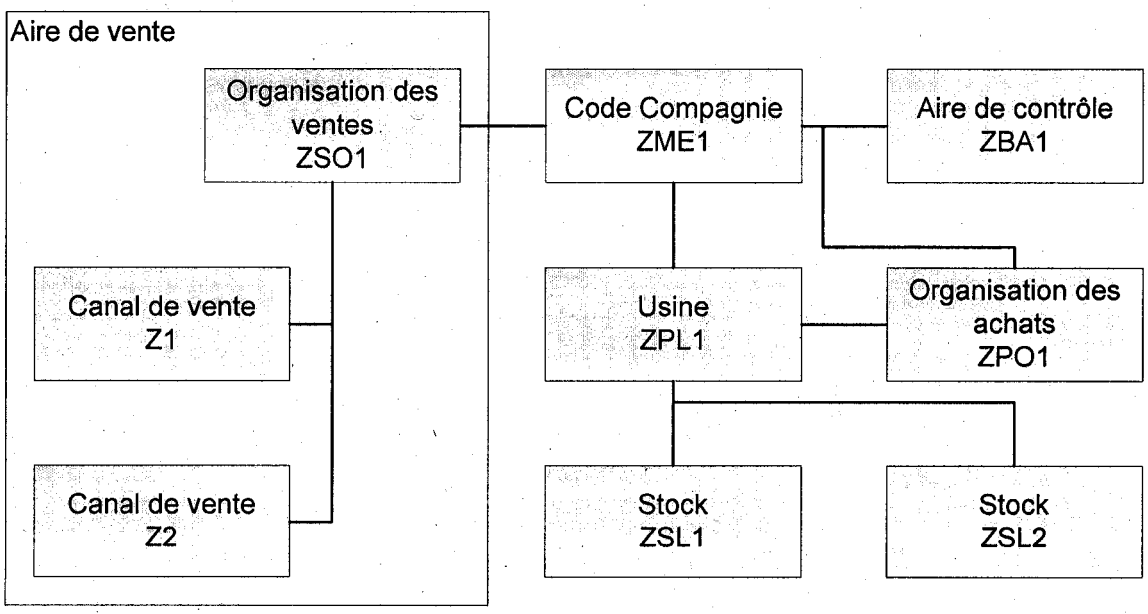


Figure 20 Éléments organisationnels du modèle d'entreprise du scénario

3.2.2 Données maîtresses

Les données maîtresses forment la pierre angulaire d'un ERP. Ces données sont stockées dans la base de données du système et elles sont utilisées par un grand nombre de processus d'affaires. Les données maîtresses sont ainsi disponibles à tous les utilisateurs autorisés au travers des différents modules de l'application. Là encore, nous avons fait en sorte de simplifier autant que possible la structure de nos produits.

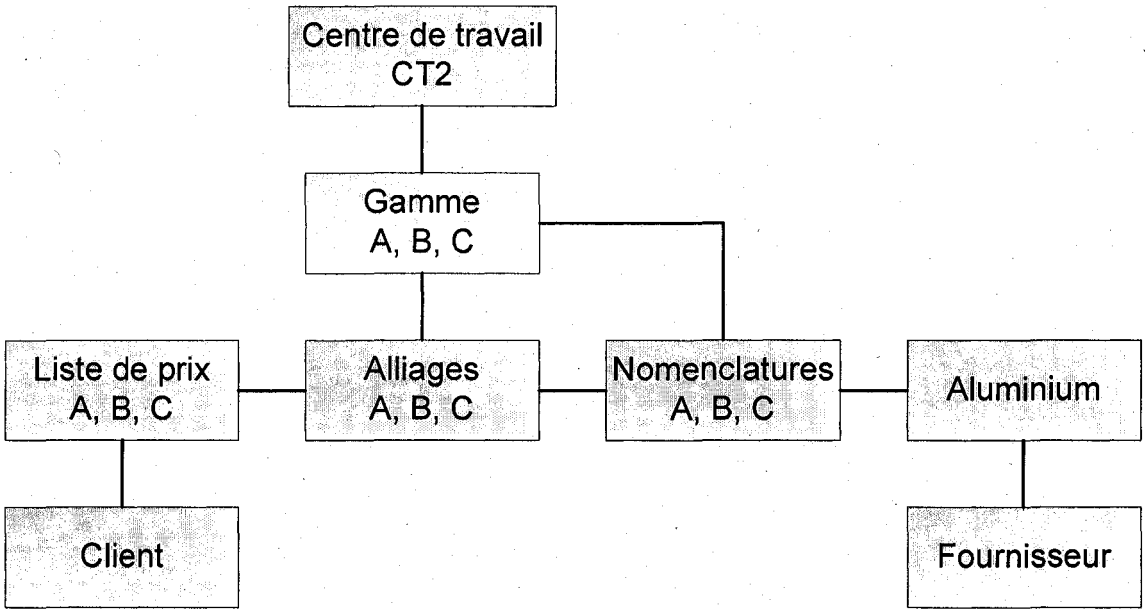


Figure 21 Données maîtresses de notre aluminerie

La Figure 21 présente les données maîtresses de notre aluminerie. Bien que ce scénario n'ait qu'un but démonstratif, il doit permettre de démontrer la viabilité du concept de processus adaptatif tout en restant simple dans sa configuration. Comme nous l'avons présenté au Chapitre 2, le processus d'ordonnancement sera réactif à l'apparition de nouveaux ordres de fabrication et à la sortie d'un ordre du centre de coulée. Pour cette

raison nous avons décidé que notre aluminerie fabriquerait 3 alliages, notés A, B, C. Par souci de simplicité, la gamme et la nomenclature de ces alliages sont réduites au strict minimum. Ils sont tous fabriqués à partir d'une même matière première en une seule étape sur un centre de travail commun représentant le centre de coulée. L'aluminium est acheté à un unique fournisseur et les alliages sont vendus à un unique client.

De cette manière, la structure organisationnelle et les données maîtresses restent simples tout en gardant une certaine diversité au niveau des produits finis. Nous allons maintenant voir les composants que nous avons dû développer pour intégrer notre ERP au reste de l'architecture.

3.3 Intégration et développements spécifiques

3.3.1 Confirmation des ordres de fabrication

Les BAPI sont un ensemble d'interfaces pour les méthodes de programmation orientées objet qui permettent à un programmeur d'intégrer un logiciel d'un fournisseur tiers aux produits SAP [123]. Les BAPI sont implémentés et stockés dans SAP R/3 comme des modules que l'on peut appeler de l'extérieur (RFC). SAP est responsable de leur développement et garanti leur maintenance. Le code source de ces méthodes est visible, mais n'est pas modifiable, tout comme les structures de données qu'elles utilisent. Le plus simple pour présenter ces outils est de montrer leur fonctionnement au travers d'un exemple qui nous intéresse : la confirmation des ordres de production.

Les BAPI permettent d'automatiser de nombreuses tâches aussi bien dans le domaine financier que de la distribution ou des ressources humaines. Il faut bien avouer que la version 46C de SAP R/3 n'est pas très riche en BAPI touchant le domaine de la planification et le contrôle de la production. Toutefois, il existe un objet, nommé *ProdOrdConfirmation*, dédié à la confirmation des ordres de production. Cet objet propose différentes méthodes pour confirmer les ordres ou les opérations.

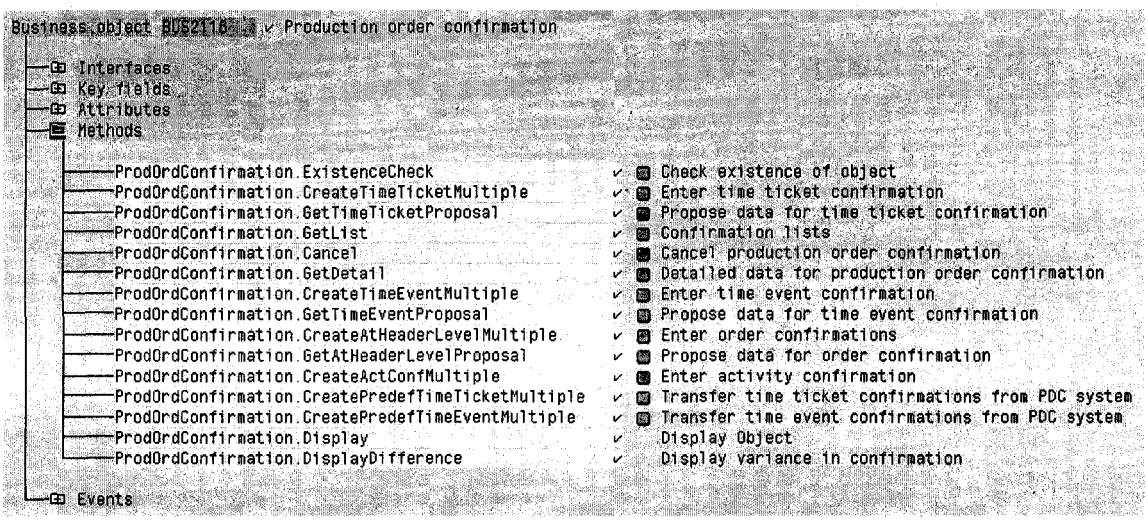


Figure 22 Méthode de l'objet ProdOrdConfirmation

Les BAPI sont répertoriés dans SAP dans l'explorateur de BAPI (transaction : BAPI). Ils sont organisés hiérarchiquement dans un ordre semblable à celui des menus de l'application. Ainsi pour trouver le BAPI de confirmation des ordres il faut descendre la hiérarchie suivant : *Production Planing and Control* ➔ *Production Orders* ➔ *ProdOrd-Confirmation*. Il est ensuite possible d'afficher les méthodes liées à cet objet (Figure 22). Il est alors possible de visualiser les paramètres nécessaires au fonctionnement de la méthode, de la tester pour s'assurer qu'elle fait bien ce qu'on en attend. La Figure 23 présente les paramètres de la méthode *CreateAtHeaderLevelMultiple* qui est celle qui nous intéresse ici. Une distinction est faite entre les éléments de base et les tables. Les tables marquées comme optionnelles sont généralement utilisées pour les valeurs de retour même si parfois ce sont uniquement des paramètres d'import optionnel (comme la table *GoodMovements* de la Figure 23). La façon la plus simple pour connaître la nature, la désignation des paramètres ainsi que le fonctionnement du BAPI est de se rendre dans le répertoire d'interface de SAP à l'adresse suivante http://ifr.sap.com/catalog/query.asp. Il est alors possible d'accéder à la description de la méthode présentée dans l'Annexe 3.

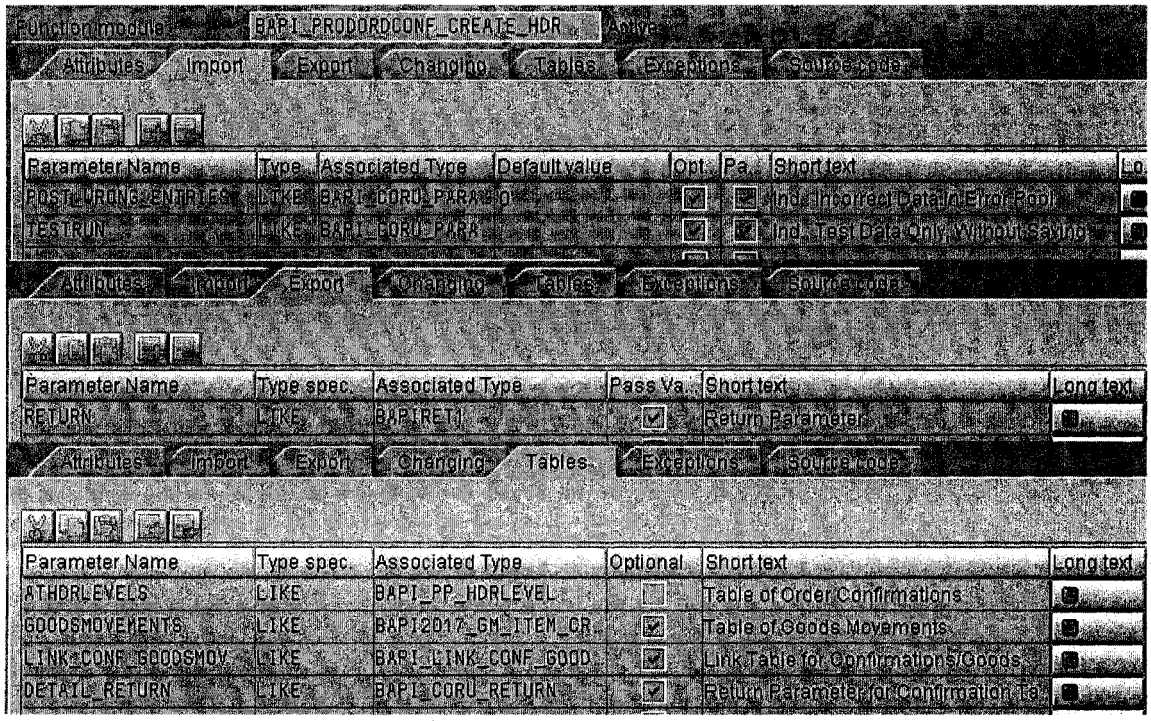


Figure 23 Paramètres de la méthode CreateAtHeaderLevelMultiple

Toute la difficulté réside ici à trouver le bon BAPI et à lui passer les paramètres nécessaires à son bon fonctionnement. Mais une fois le bon BAPI repéré et testé, il faut encore pouvoir l'utiliser depuis un système externe. C'est ce que nous verrons dans une section ultérieure portant sur les connecteurs.

3.3.2 Liste des ordres de fabrication

Les BAPI ont un grand avantage, ils limitent l'effort de développement et de maintenance. Malheureusement, et bien qu'il y ait de très nombreux BAPI, il arrive toujours un moment où nous avons besoin d'une fonction bien spécifique qui n'a pas été encore prévue par SAP. Dans ce cas-là, il n'y a pas d'autre moyen que de développer soi-même une fonction dans SAP répondant à nos besoins. C'est que nous avons dû faire

dans notre scénario. En effet, aucun BAPI n'existe pour lister les ordres de production en fonction de leurs statuts.

SAP R/3 peut être considéré comme un énorme système de gestion de base de données. La base de données supportant tous ces processus contient plus de 60 000 tables. Bien souvent, le développement de fonctions en ABAP, le langage de programmation de SAP, consiste à interroger un ensemble de tables et de procéder à un traitement des données. Toute la difficulté est alors de trouver les bonnes tables à interroger. Trouver ces tables est primordial pour 2 raisons :

- obtenir les informations pertinentes répondant au problème posé ;
- trouver les bons types de données d'entrée et de sortie de la fonction que nous allons écrire.

À notre sens, les meilleures sources d'information restent les informations techniques accessibles depuis l'aide de SAP R/3 et surtout les forums d'entraide. Là encore, il n'est pas rare que nous ne soyons pas les premiers à être confrontés au problème.

Dans le cadre de notre scénario, nous souhaitons réagir à l'arrivée d'un nouvel ordre ou à la sortie d'un ordre du centre de coulée. À l'occurrence de l'un de ces événements, le modèle d'optimisation sera lancé afin d'ordonnancer la production. Il est donc nécessaire que le module d'optimisation récupère la liste des ordres de production relâchés, mais pas encore confirmés. C'est à dire tous les ordres que notre fonderie a encore à produire. Le Tableau IV décrit succinctement des tables SAP en relation avec ces ordres. Dans notre cas, nous aurons donc besoin des informations de 2 tables :

- la table AFPO qui contient les en-têtes des ordres de fabrication ;
- la table JEST qui contient les statuts des objets (dont ceux des ordres).

Tableau IV

Tables SAP en relation avec les ordres de production

TABLES	DESCRIPTION
AFFL	Séquence des ordres de production – un numéro par opération
AFKO	En-tête des ordres de production (identique à CAUFV sans les noms et les dates)
AFPO	**En-tête des ordres de production / résumé**
AFVC	Détails des operations
AFVU	Structure de données sur les « USER FIELD" pour chaque opération
AFVV	Structure de données sur les qté/dates/valeurs sur les opération
AUFK	En-tête des ordres de production et des ordres internes - COSTING
AUFM	Mouvement des matières pour les ordres
AUPO	Ordres de production - Achat
CAUFVD	En-tête des ordres de production
JCDS	Date de changement de Statuts pour chaque type de statut (utilisateur et system)
JEST	**Statuts des ordres de production et des operations - détail**
PLAF	Ordres planifiés
S021	Structure des ordres de production (LIS)
S022	Structure des ordres de production avec les opérations (LIS)
S027	Informations sur le coût des produits (LIS)
T0030	Type d'ordres (liée à AFPO-AUFART)
TJ02	Description des statuts
TJ30T	Statuts définis par l'utilisateur

Afin de mener à bien l'ordonnancement, le module d'optimisation va avoir besoin de connaître un certain nombre d'éléments qui se trouvent dans la table AFPO (Figure 24) :

- le numéro d'ordre de production ;
- l'alliage à produire ;
- la quantité à produire ;
- l'unité de mesure ;
- la date de fin.

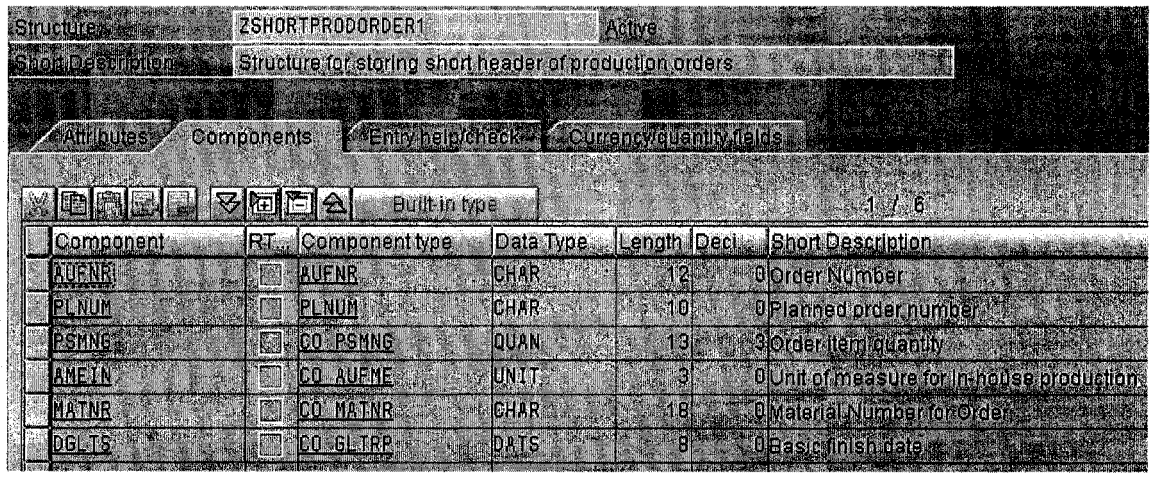


Figure 24 Structure des données renvoyée par la fonction

Les types des champs de cette structure ont une grande importance. Ils doivent être en accord avec la donnée qu'ils vont recevoir. Autrement dit, ils doivent avoir le même type que la source de données, la table AFPO dans notre cas. Il en va de même pour les paramètres d'entrée. La fonction que nous avons développée, *ZShortProdOrder_GetList*, recherchera, pour une usine et un alliage donnés, tous les ordres de production ayant le statut relâché et n'ayant pas le statut confirmé. Les paramètres d'entrée et de sortie sont ceux de la Figure 25. La structure Return sert à renvoyer un indicateur de succès ou d'échec et dans ce dernier cas, des informations sur la cause de l'échec. Grâce à cette

fonction, nous allons récupérer un certain nombre d'ordres de production. Pour cette raison, la fonction renvoie une table basée sur la structure de la Figure 24.

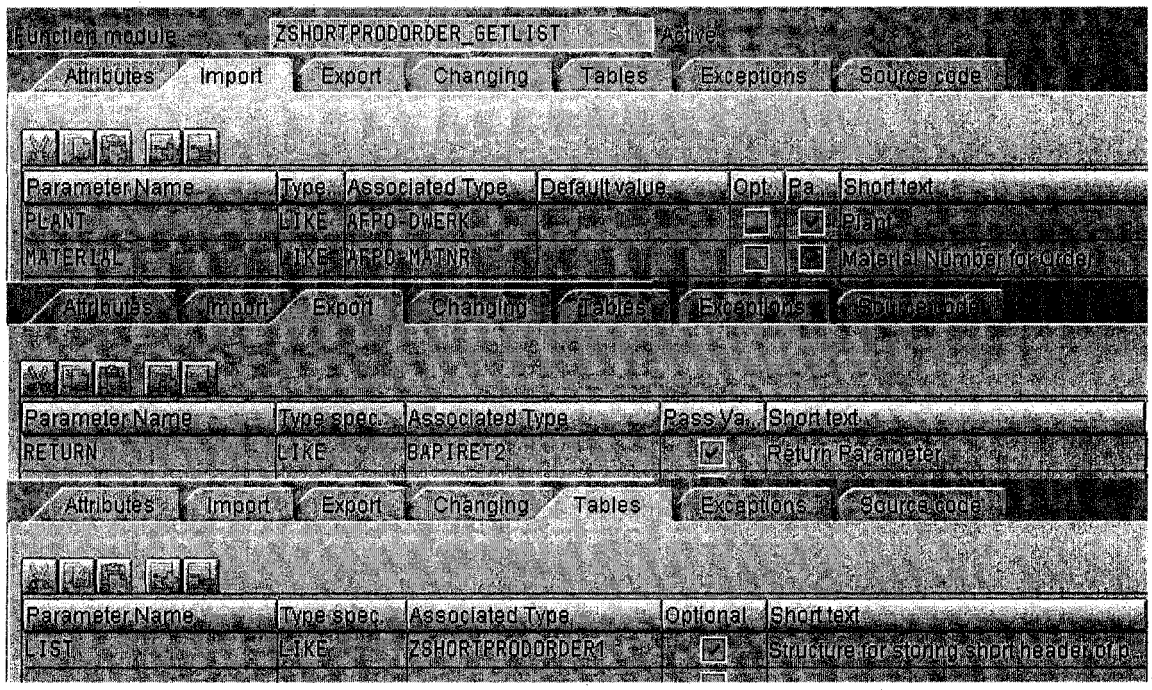


Figure 25 Paramètre d'entrée et de sortie de la fonction ZShortProdOrder_GetList

La table AFPO ne contient que les en-têtes des ordres et ne contient malheureusement pas leurs statuts. La table JEST contient uniquement les statuts des objets du système. Les ordres étant considérés comme des objets, nous pourrons y trouver l'information manquante. Reste un dernier écueil avant de passer à la logique de la fonction. En observant l'interface de SAP R/3 (Figure 26), les statuts sont affichés sous la forme d'un code alphabétique relativement déchiffrable et dépendant de la langue de l'utilisateur. Or dans la table JEST, ils sont codés sous une forme totalement différente. Un passage par la table TJ02T nous permettra de repérer les statuts à rechercher :

- le statut relâché (REL dans l'interface utilisateur) correspond au statut I0002 dans la table JEST;

- le statut confirmé (CNF dans l'interface utilisateur) correspond au statut I0009 dans la table JEST;

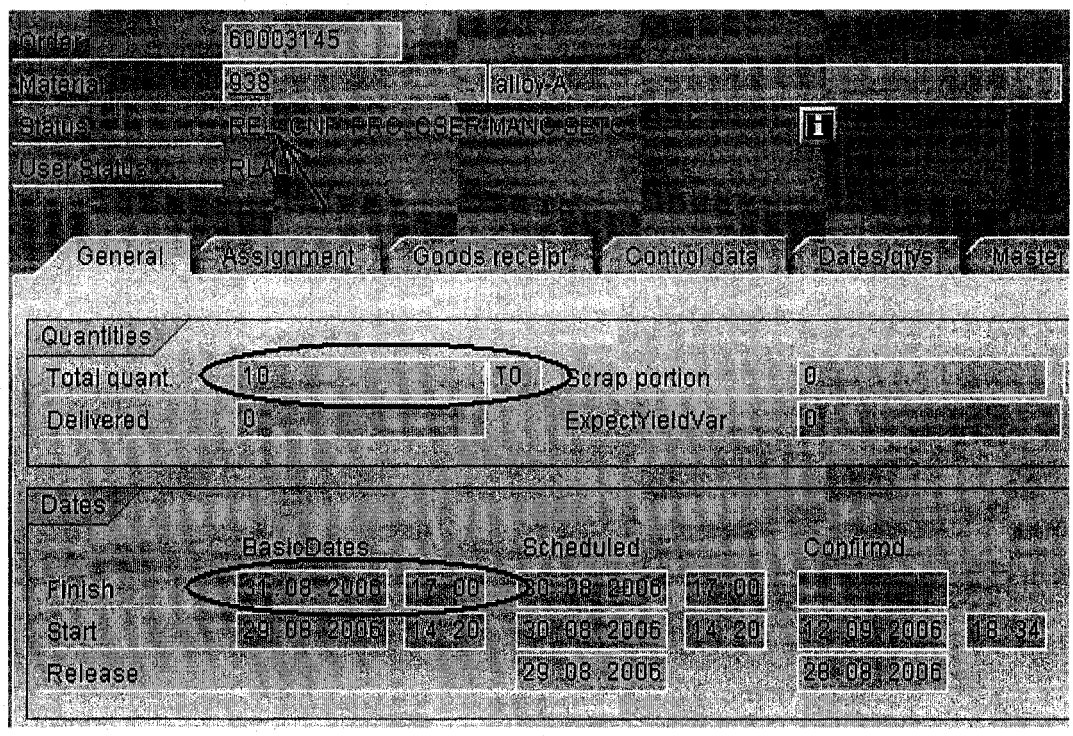


Figure 26 Informations désirées vues dans l'interface utilisateur

La fonction créée commence par rechercher tous les ordres de production correspondant à la bonne usine et au bon alliage. Si à ce stade, la requête reste sans résultat, une alerte est levée et l'information est passée grâce à la structure *Return*. Pour chacun des résultats, le numéro d'objet est reconstitué et les statuts de l'objet sont recherchés. On vérifie la présence du statut I0002 (ordre relâché). S'il n'est pas là, on passe au traitement de l'ordre suivant. S'il est présent, on vérifie l'absence du statut I0009 (ordre confirmé). S'il est là, on passe au traitement de l'ordre suivant. Sinon, l'ordre rencontre tous les critères de sélection et est copié dans la structure de sortie. Une fois tous les ordres traités, si aucun n'a été ajouté à la structure de sortie, une alerte est levée et

l'information est passée grâce à la structure *Return*. La Figure 27 représente la logique de la fonction. Le code est dans l'annexe 4. Le lecteur trouvera aussi en Annexe 5 un support présentant tous les aspects de la création de BAPI dans SAP R/3 46C.

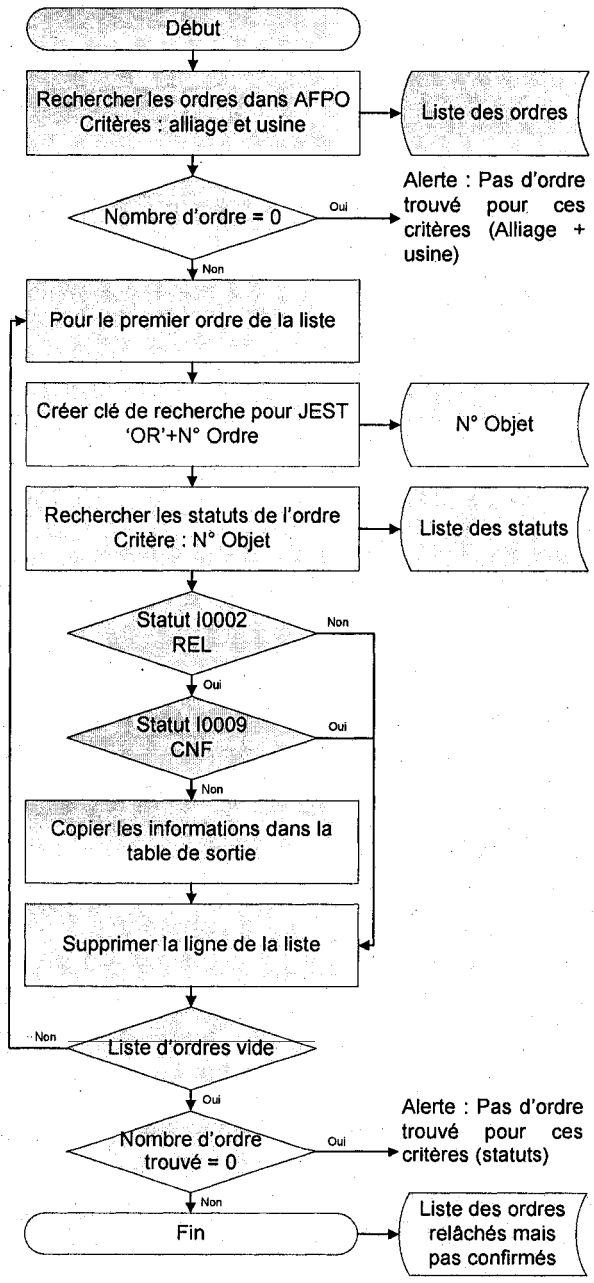


Figure 27 Logique de la fonction ZShortProdOrder_GetList

Une fois la fonction testée, il ne reste plus qu'à indiquer à SAP que cette fonction est accessible par un système tiers. Il faut aussi l'activer. L'activation a pour effet de rendre inaccessible à la modification les structures de données utilisées dans la fonction. Ce mécanisme protège des dysfonctionnements qui apparaîtraient si un individu venait à modifier une des structures liées. Il est aussi possible de le voir comme un contrat : le développeur s'engage à fournir les informations sous cette forme. Bien entendu rien n'empêche de modifier le code source de la fonction.

Là encore, tout le travail s'est effectué dans SAP R/3. Maintenant il faut pouvoir utiliser notre travail depuis un système externe. C'est ce que nous allons présenter ci-dessous.

3.3.3 Connecteur Java (JCo)

Pour faciliter l'intégration de SAP dans les environnements hétérogènes, il existe des connecteurs dans les principaux langages de programmation (.NET, PHP, JAVA). SAP met à disposition des développeurs un connecteur en JAVA baptisé JCo pour *Java Connector*.

JCo peut facilement s'intégrer dans une application tierce et permettre ainsi d'exécuter des fonctions dans SAP et de récupérer les données souhaitées, et ce, même si les deux applications ne sont pas installées sur le même serveur ou dans le même réseau. En utilisant vos paramètres de connexion, JCo va ouvrir une session dans SAP R/3 et récupérer le dictionnaire des fonctions utilisables ainsi que leurs paramètres d'entrée et de sortie. Il permettra ensuite d'appeler ces fonctions et de les exécuter. Les données reçues en retour pourront être exploitées par la suite pour alimenter le module d'optimisation par exemple.

Ce connecteur a pour principal intérêt d'utiliser le dictionnaire de fonction de SAP qu'il demande à chaque ouverture de session. Il est ainsi complètement générique et tant que

les fonctions appelées ne changent pas, il n'y a pas besoin de maintenance. De plus, il est très facile d'utilisation. L'annexe 6 présente le code appelant la fonction *ZShortProdOrder_GetList* utilisée dans notre scénario.

3.4 Conclusion

Dans ce chapitre nous avons vu les éléments de configuration du système ERP en vue de son utilisation dans notre scénario. L'accent a été mis sur la simplification des processus qui n'interviennent pas directement dans le processus adaptatif, par exemple les processus financiers. La suite a été consacrée aux éléments d'intégration de l'ERP dans l'architecture hétérogène proposée. Ces éléments d'intégration peuvent être standards, comme les BAPI, ou bien développés par nos soins pour répondre à un besoin plus spécifique, comme la fonction *ZShortProdOrder_GetList*. Ces deux éléments n'auraient que peu d'intérêt s'il était impossible de les utiliser depuis un système tiers. Ce dernier écueil est levé par l'utilisation de JCo, un connecteur Java permettant l'exécution de fonctions dans SAP depuis un système externe. Des exemples fonctionnels tirés de notre modèle ont été présentés.

Grâce à ces différents éléments il est désormais possible d'intégrer l'ERP au MES et au module d'optimisation en temps réel. Nous allons maintenant voir comment il est possible de connecter le module d'optimisation ou le MES au processus manufacturier pour récupérer les statuts de la fonderie afin d'initialiser le modèle d'optimisation.

CHAPITRE 4

SCÉNARIO D'ORDONNANCEMENT RÉACTIF – MES – ACQUISITION DE DONNÉES EN TEMPS RÉEL

4.1 Introduction

L'ordinateur, utilisé comme composant d'intégration, a apporté des changements en matière d'automatisation. Les standards du web descendent maintenant jusque dans l'atelier et pour l'utilisateur final, l'utilisation de standards ouverts performants est devenue un gage de pérennité. La baisse des coûts de l'équipement informatique, l'augmentation des capacités de traitement, l'ouverture des standards en font un outil efficace pour l'acquisition et le traitement des données des processus de production ainsi que pour le contrôle de ces processus. C'est un outil complémentaire des traditionnels PLC et autres terminaux et dans certains cas il les remplace même.

L'utilisation d'interfaces standardisées par de nombreux fabricants permet aux composants de l'architecture d'être plus flexible et plus interopérable. Il devient ainsi plus aisé d'intégrer les composants de vendeurs différents. Le standard OPC est maintenant accepté comme l'un des standards industriels les plus populaires auprès des utilisateurs et des développeurs. De nombreux fabricants d'IHM (*Human Machine Interface*), SCADA et DCS, aussi bien que les fabricants de logiciels PLC offrent des clients et/ou des serveurs OPC avec leurs produits. Il en va de même avec les fournisseurs de services et de cartes d'interface. Dans les dernières années, les serveurs OPC ont largement remplacé les serveurs DDE (échange dynamique de données ou *Dynamic Data Exchange*) et les protocoles propriétaires.

Nous présentons ici le standard OPC utilisé pour l'échange de données en temps réel et le développement d'un client OPC. Nous avons fait le choix de développer un client OPC pour pouvoir l'intégrer au besoin au processus d'optimisation. En effet, il serait

alors possible de court-circuiter le MES dans le processus d'acquisition des données du processus manufacturier.

4.2 Standards OPC

4.2.1 Contexte

OPC est un standard industriel public pour l'interconnectivité des systèmes. Son but est de fournir une infrastructure pour l'échange de données de contrôle reposant sur un standard. Il utilise les technologies COM et DCOM de Microsoft pour permettre aux applications d'échanger des données entre un ou plusieurs ordinateurs en utilisant une architecture client/serveur. L'Annexe 7 introduit succinctement les concepts COM et DCOM. OPC définit un ensemble commun d'interfaces de telle sorte que les applications reçoivent des données aux mêmes formats, et ce, sans se préoccuper de savoir si la source de ces données est un PLC, un DCS, une jauge, un analyseur, un autre logiciel ou tout autre type de source. Ainsi, OPC est une solution de communication directement utilisable après son installation. Pour illustrer cela, considérons une entreprise qui dispose de différentes sources de données telles que des PLC, des DCS, des bases de données, des jauges et bien d'autres encore. Ces données sont disponibles via différentes connexions telles que des connexions série, par Ethernet ou même par ondes radio. Différents systèmes d'exploitation comme Windows, UNIX ou DOS sont aussi utilisés par les nombreuses applications de contrôle du processus de production.

Dans le passé, il fallait recueillir ces données grâce à des applications dédiées utilisant leurs propres jeux d'interfaces. Les données devaient être conservées dans un format propriétaire ce qui signifie que vous ne pouviez accéder à vos données qu'en utilisant des outils provenant du même vendeur, celui qui a verrouillé à l'origine vos données. Vous étiez alors forcé de vous adresser à ce même vendeur à chaque fois que vous aviez besoin de changer ou d'étendre votre système.

Contrairement à tout cela, OPC standardise une technologie plutôt qu'un produit. En utilisant cet ensemble de normes, les données peuvent être transmises de n'importe quelle source vers n'importe quelle application compatible OPC telles qu'une IHM, des archives de données, un ERP, ... OPC est un standard de communication qui fournit une véritable interopérabilité et une réelle adaptabilité. Cela nous permet de visualiser, d'analyser, de rapporter ou de faire ce que l'on souhaite avec les applications de n'importe quel vendeur utilisant une ou plusieurs spécifications OPC. Une telle interopérabilité réduit les coûts de développement et est aussi facilement adaptable aux développements futurs.

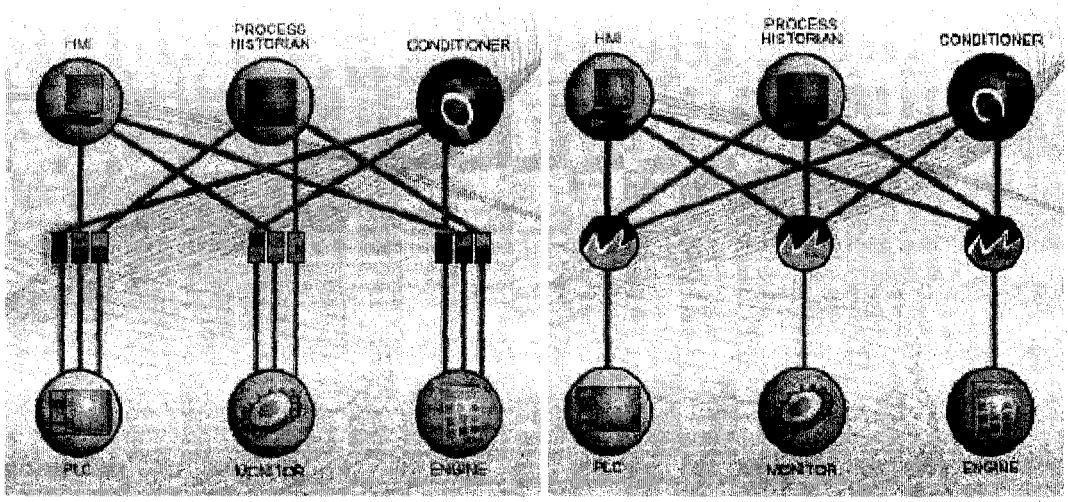


Figure 28 Exemple d'intégration avec une solution propriétaire (à gauche) et par le standard OPC (à droite) (source : MatrikonOPC)

La Figure 28 présente un exemple simple où nous retrouvons 3 sources de données (un PLC, un système de surveillance des vibrations et un calculateur) et 3 utilisateurs de ces données (une IHM pour la visualisation, un gestionnaire d'historique pour le stockage des données et une application surveillant les conditions d'évolution de la machine). Une connectivité propriétaire demanderait que chaque application possède un pilote pour

chaque source de données. Au final, il faudrait développer 9 pilotes. En utilisant le standard OPC, il faudrait équiper chaque source de données avec un serveur OPC, et chaque application utilisatrice d'un client OPC si elles ne sont pas déjà compatibles. Les serveurs et clients OPC étant standardisé il n'est pas nécessaire de tout redévelopper, il est même possible d'utiliser des produits commerciaux. L'effort de développement est réduit par 3 et la maintenance est grandement facilitée.

4.2.2 Spécifications OPC

Étant un standard industriel ouvert pour l'échange de données en temps réel, les spécifications OPC sont disponibles pour toute personne intéressée par le développement de ses propres produits OPC. Les principales spécifications disponibles sont les suivantes :

- **OPC Data Access**, ou OPC DA, fournit un accès en temps réel aux données du processus de production. En utilisant OPC DA, il est possible de demander au serveur les plus récentes valeurs du flux, des pressions, des niveaux, des températures, des densités, ...
- **OPC Historical Data Access**, ou OPC HDA, est utilisé pour rapatrier et analyser les données historiées du processus de production qui sont typiquement stockées dans une archive des données de production ou une base de données.
- **OPC Alarms and Events**, ou OPC A&E, est utilisé pour échanger des événements et des alarmes en provenance du processus de production.
- **OPC Data eXchange**, ou OPC DX, définit comment les serveurs OPC échangent des données avec d'autres serveurs OPC.
- **OPC eXtensible Markup Language**, ou OPC XML, encapsule les données de contrôle et les rends disponibles pour tous les systèmes d'exploitation.

D'autres spécifications, telles que les spécifications OPC Batch et OPC Security, sont plus confidentielles et plus rarement utilisées. La Fondation OPC continue de mettre à jour les spécifications OPC existantes et travaille aussi sur de nouvelles spécifications comme celle pour les données complexes (OPC for Complex Data) ou encore pour les systèmes ERP (OPC for ERP Systems). Bien choisir la spécification pour notre application est important. Par exemple, OPC DA et OPC HDA sont deux spécifications distinctes utilisées pour des applications tout aussi différentes. De plus, chaque spécification OPC a différents numéros de version comme OPC DA version 1,2 et 3. Il faut savoir que tous les types et toutes les versions ne sont pas supportés par tous les produits vendus.

4.2.3 OPC DA – Accès en temps réel aux données

L'une des spécifications OPC les plus populaires est OPC Data Access ou OPC DA. Elle fournit une méthode standard pour accéder en temps réel aux données en provenance des logiciels ou matériels de contrôle de l'atelier. Avec elle, la communication de toutes les applications utilisatrices de données est relayée par un serveur OPC DA vers les PLC, DCS ou toutes autres sources de données.

OPC DA est uniquement utilisé pour écrire et lire les données en temps réel. Pour accéder à d'anciennes valeurs, il faut utiliser OPC HDA. Pour comprendre OPC DA, considérons un capteur mesurant le débit d'une pompe. Dans cet exemple le capteur est connecté à un PLC qui échange des données avec une application client de type IHM. OPC DA fournit l'accès à des items appelés points. Le contrôleur de débit aura plusieurs points, dont un point d'assignation (CD101.PA) et un point pour la valeur du débit (CD101.VP). OPC DA traite chacun de ces points comme s'ils étaient des points séparés. Chaque point inclut trois attributs : une valeur, une qualité, et une date. OPC DA transmet la valeur du point, la fiabilité de cette valeur et l'heure de lecture de la valeur, que ce point vienne d'un PLC, d'un DCS ou d'une application. Par exemple, une

lecture du débit peut avoir pour valeur 12.85 centimètres cubes par minute, avec une qualité « Bonne » et datée du 25 septembre 2006 à 13 :30 et 17.358 secondes.

La question typique d'un client OPC posée à un serveur OPC est : Quel est le débit actuel de CD101, quelle est la fiabilité de la lecture et à quelle heure la valeur a-t-elle été lue ? OPC spécifie qu'un horodatage doit être fourni pour chaque point, mais il ne dit pas d'où l'horodatage doit provenir. Parfois l'horodatage n'est pas fourni par la source. Par exemple, Modbus ne fournit pas un horodatage en provenance de l'automate. Dans un tel cas, le serveur fournit son propre horodatage. D'autres sources sont en mesure de le fournir et quand le serveur OPC reçoit une lecture, il reçoit aussi un horodatage à transmettre au client. Un serveur OPC peut être conçu pour ignorer les horodatages quand il en reçoit un. Certains vendeurs de serveurs OPC choisissent d'ignorer délibérément l'horodatage de la source et si cet aspect a une importance, il faut toujours se poser la question de la provenance de l'horodatage.

OPC DA fournit un standard pour l'accès en temps réel aux données qui va permettre une connectivité ouverte. Les systèmes OPC bénéficient d'une réelle interopérabilité et adaptabilité utilisable pour surveiller et superviser les processus. OPC permet aux applications d'échanger, c'est-à-dire de lire et écrire, les dernières valeurs et non pas les valeurs passées (pour cela il faut utiliser OPC HDA).

4.3 Client OPC DA

4.3.1 Interfaces

OPC est suffisamment flexible pour être utilisé à différents niveaux dans l'architecture du système et il est spécialement conçu pour pouvoir travailler en réseau. Pour ces raisons il utilise DCOM, un modèle objet de composants distribués qui permet de créer des API, créé par Microsoft dans les années 90. L'efficacité d'OPC est assurée par le fait que, contrairement à DDE (mécanisme d'échange de données entre les applications de

Windows), OPC peut englober plusieurs centaines d'items à chaque transaction. Les performances sont ainsi garanties par la multitude d'items traités à chaque transaction, mais aussi grâce à la possibilité fournie par COM de créer des serveurs sous différentes formes : *inprocess* (contenu dans une DLL), local ou distant. Les serveurs *inprocess* fournissent les plus grandes performances en traitant des millions de transactions par seconde. Les serveurs installés sur une machine locale sont ainsi capables de traiter des milliers de transactions par seconde et les serveurs distants, des centaines de transactions par seconde.

Les serveurs OPC sont accessibles aux clients qui peuvent être écrits dans différents langages de programmation, incluant le C++ et le Visual Basic. OPC distingue deux sortes d'interface : la *Custom Interface* et l'*Automation Interface* (Figure 29). La *Custom Interface* est utilisée par les langages de programmation qui supportent le concept de pointeur (C ou C++). Pour les clients écrits dans un langage ne comportant pas ces concepts, l'*Automation Interface* a été introduite. Cette interface utilise une interface standard où les méthodes ne sont pas appelées par des pointeurs, mais par leur nom.

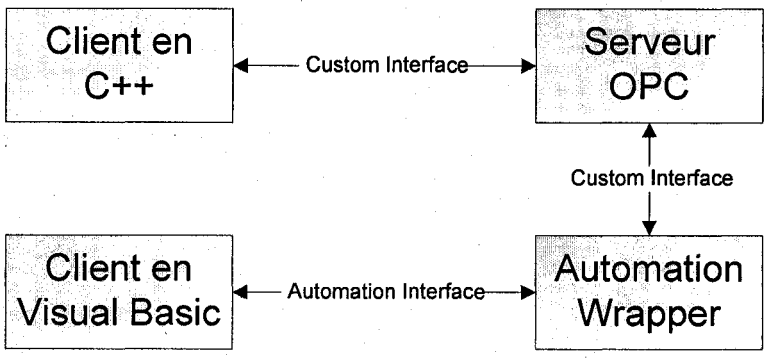


Figure 29 Accès au serveur OPC en utilisant la custom ou l'automation interface

Les serveurs OPC sont obligés d'implémenter la *Custom Interface* et peuvent, s'ils le souhaitent, fournir l'*Automation Interface*. La Fondation OPC fournit une *Automation*

Interface standard sous la forme d'une DLL. L'interface est standard, mais dépendante de la version du langage utilisé pour développer le client. Par exemple, la DLL n'est pas la même suivant que le code soit en VB6 ou en VB.net. Pour éviter tout problème quant à l'existence incertaine d'une *Automation Interface* fournie avec le serveur, notre client est développé en C++, langage pour lequel la notion de pointeur est naturelle. De plus, l'utilisation du langage C++ nous permet de gagner en rapidité d'exécution.

4.3.2 Structure logique

4.3.2.1 Espace des noms

L'espace de noms contient les références de toutes les sources de données disponibles via le serveur. Il peut avoir une structure en arborescence sur plusieurs niveaux (espace de noms hiérarchique) ou une structure plate (espace de noms plat). Dans un espace de nom plat, tous les items sont dans un unique niveau. Il n'y a pas de nœud. Dans un espace de noms hiérarchique, les nœuds peuvent être utilisés dans un but structurel ; ils peuvent par exemple représenter un département de l'atelier. Les items sont disponibles à partir des nœuds et représentent les sources des données.

Pour l'application qui va utiliser les données, les sources ne sont pas les seuls points d'importance, les informations sur les types de données le sont tout autant. Pour cette raison, des attributs, aussi appelés propriétés, sont alloués aux nœuds et aux items. Le nom du fabricant de la source peut être considéré comme une de ces propriétés. En plus des propriétés, des chemins d'accès peuvent être disponibles pour un item. Ils peuvent être utilisés pour décrire la voix de communication entre le serveur OPC DA et le service (par exemple : COM1, 9600kBits/s). Le chemin d'accès est optionnel et peut être omis.

4.3.2.2 Modèle logique

Pour être utilisable par le plus large public, OPC se base sur quelques observations générales :

- Les applications clientes ne sont intéressées que par un sous-ensemble de données (items) parmi toutes celles mises à disposition.
- Les applications clientes sont intéressées par différents sous-ensembles d'items à différents instants et par des résolutions différentes.
- Les applications clientes veulent être indépendantes des objets du processus manufacturier.

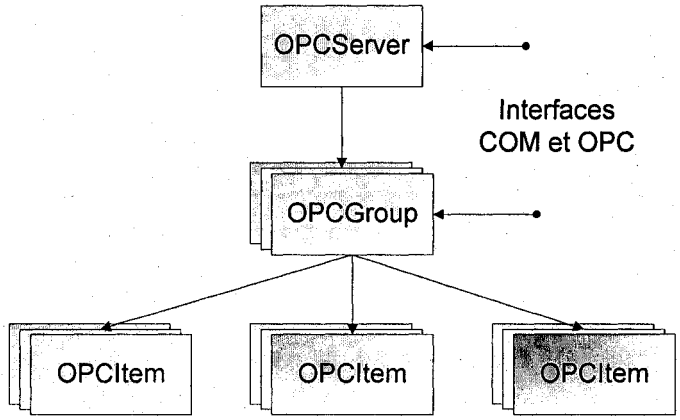


Figure 30 Modèle logique des clients OPC

En prenant en compte ces différentes observations, il est alors possible de représenter le modèle logique de la Figure 30. Au sommet de la hiérarchie se trouve l'objet serveur ou OPCServer. C'est l'objet COM auquel il se connectera en premier l'application. Le plus bas niveau est formé des objets OPCItem représentant une valeur ou une propriété d'une des sources disponibles. Au niveau intermédiaire, les objets OPCGroup sont utilisés pour structurer les objets OPCItem. Cette structuration peut se faire en suivant un aspect logique (toutes les variables d'un département) ou suivant un aspect dynamique (les

variables du processus ayant un comportement temporel semblable ou proche) ou encore suivant l'inspiration de l'utilisateur. Les objets OPCGroup sont gérés par l'objet OPCServer, de la même façon les objets OPCItem sont gérés au niveau de l'objet OPCGroup. Le nombre de niveaux hiérarchiques pour les objets OPC du client est fixe. La Figure 31 met en vis-à-vis la structure logique du serveur et du client. L'espace de noms est unique et est le même pour tous les clients. Le client représenté ici a créé deux objets OPCGroup et pour chaque objet OPCGRoup, deux objets OPCItem.

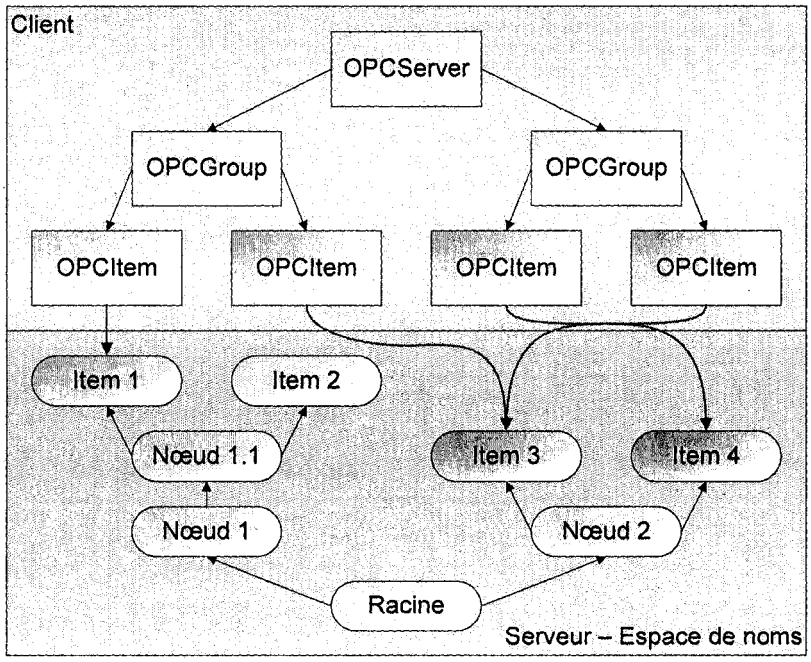


Figure 31 Hiérarchies du serveur et du client

Seuls deux des objets du modèle hiérarchique du client sont de vrais objets DCOM : les objets OPCServer et OPCGroup. Pour ces deux objets, des interfaces, des méthodes et des paramètres ont été définis. L'objet OPCItem n'a pas d'interfaces. Dans les applications clientes, il peut être intéressant de pouvoir lire et écrire plusieurs items en

même temps. Il est d'ailleurs plus efficace d'accéder à plusieurs objets OPCItem par un seul appel. Ces appels sont réalisés via les méthodes et les interfaces de l'objet OPCGroup.

4.3.3 Fonctionnement séquentiel

Un client OPC suit une séquence de démarrage bien précise présentée à la Figure 32. Chaque étape de cette séquence se focalise sur un objet OPC particulier. Cette séquence à aussi été utilisée pour le développement du client. La première étape consiste à rechercher les serveurs OPC DA disponibles par l'intermédiaire de la base de registre. Une fois le serveur identifié, vient l'étape de la connexion. Cette connexion se matérialise par la création d'un objet OPCServer. Une fois la connexion établie, il est possible de créer les OPCGroups pour organiser l'espace de travail. Les étapes suivantes vont consister à créer les items que l'application va vouloir lire ou modifier. Il faut tout d'abord explorer l'espace de noms du serveur pour finalement créer l'objet OPCItem.

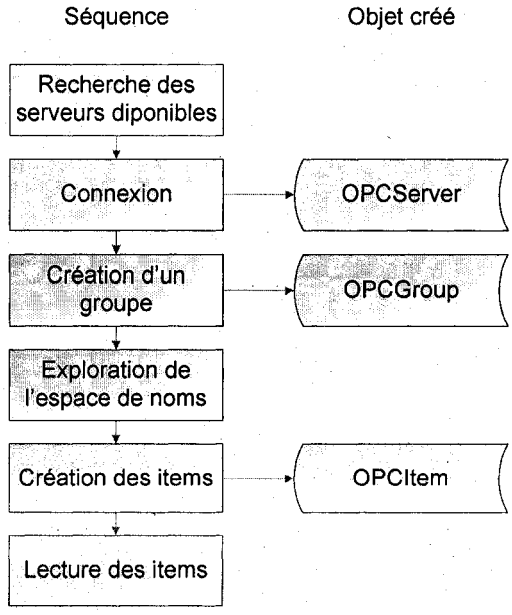


Figure 32 Séquence de fonctionnement d'un client OPC DA

4.3.3.1 Recherche des serveurs disponibles

Le but de cette recherche est de pouvoir lister les serveurs déjà installés et faciliter ainsi le travail de configuration à l'utilisateur. Il est toujours possible pour l'utilisateur de configurer manuellement un serveur pour s'y connecter.

Le client a besoin de l'identifiant de classe (CLSID) du serveur pour pouvoir fonctionner et instancier correctement les objets DCOM. La fondation OPC oblige donc les serveurs OPC à faire 3 entrées dans le registre de Windows :

- L'identification du programme (ProgID) contient une chaîne de caractères lisible décrivant le composant. La structure de la chaîne est prédéfinie : <Fabricant>.<Nom du serveur>. Par exemple, on pourra trouver Matrikon.Simulation_OPC_DA (la chaîne de caractère ne devant contenir aucun espace).
- L'identification de classe (ClassID ou encore CLSID) est une clé unique sur 128 bits qui identifie le serveur. La valeur de cette clé n'est allouée qu'à un seul serveur et ce serveur n'est lié qu'à cette CLSID. Si des changements sont effectués dans la structure du serveur (ajout de nouveaux objets ou de nouvelles interfaces), une nouvelle clé devra être utilisée.
- L'identification de l'application (AppID) contient une autre description du serveur, mais sa structure et son contenu ne sont pas prédéfinis.

Comme plusieurs spécifications et plusieurs versions de chaque spécification peuvent être disponibles, ces seules informations ne suffisent pas. En effet, un client *Data Access* ne souhaite communiquer qu'avec les serveurs *Data Access* et pas avec les serveurs *Alarms and Events*. L'identification de catégorie (CATID) permet de distinguer parmi tous les produits OPC ceux qui nous intéressent. La CATID est située dans une sous-clé

de la CLSID. Elle est définie de manière unique pour chaque spécification. Les CATID représentant les différentes spécifications OPC DA sont celles de la

Tableau V

CATID des versions de la spécification OPC DA

Spécification Data Access 1.0A	{63D5F430-CFE4-11D1-B2C8-0060083BA1FB}
Spécification Data Access 2.0	{63D5F432-CFE4-11D1-B2C8-0060083BA1FB}
Spécification Data Access 3.0	{CC603642-66D7-48F1-B69A-B625E73652D7}

4.3.3.2 Connexion au serveur OPC DA

La connexion d'un client OPC à un serveur revient pour le client à créer une instance de l'objet OPCServer qui permettra alors de créer et gérer les objets OPCGroup. L'objet OPCServer va mémoriser les pointeurs vers les interfaces implémentées par le serveur. Ces différentes interfaces sont présentées dans la Figure 33.

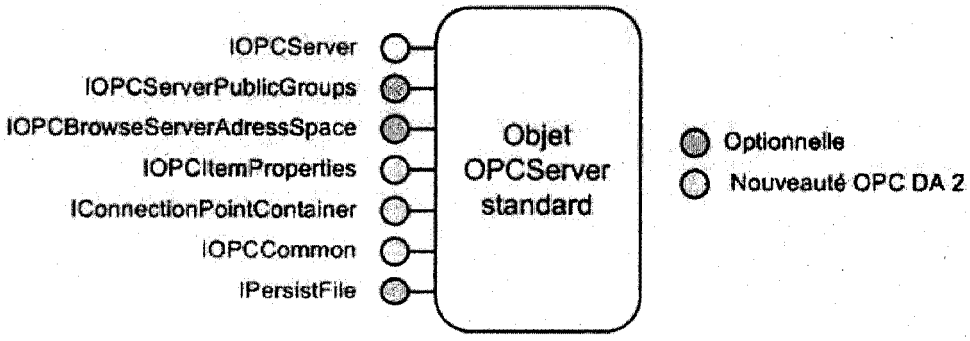


Figure 33 Interfaces de l'objet OPCServer

Sans entrer dans les détails, ces différentes interfaces vont permettre de créer ou de supprimer des groupes, d'explorer l'espace de noms du serveur, de transcrire les codes d'erreurs en texte plus explicite et d'obtenir des informations sur le statut du serveur.

La connexion au serveur va consister à instancier l'interface IOPCServer et à mémoriser les paramètres suivants :

- **ProgID** : une chaîne de caractères représentant l'identification du programme.
- **RemoteMachine** : une chaîne de caractères représentant le nom de la machine distante.
- **Connected** : un booléen représentant l'état de la connexion avec le serveur.
- **pIUnknow** : un pointeur vers l'interface IUnknown.
- **pIOPCCommon** : un pointeur vers l'interface IOPCCommon.
- **pIOPCServer**: un pointeur vers l'interface IOPCServer.

L'objet développé contient en plus de cela un constructeur permettant de l'initialiser ainsi que les fonctions permettant de lire et d'assigner les variables membres. Elle contient en plus des méthodes permettant d'obtenir l'identification de classe du serveur (CLSID), de se connecter, de se déconnecter et de tester l'état de la connexion et de gérer les groupes.

D'un point de vue fonctionnel, après la recherche dans le registre, l'utilisateur accède à la liste des différents serveurs compatibles. Il sélectionne le serveur qui l'intéresse et se connecte. L'objet OPCServer est alors instancié et la méthode de connexion est appelée. Un moyen simple de tester la connexion est de demander le statut du serveur. La Figure 34 présente cette partie de l'interface utilisateur.

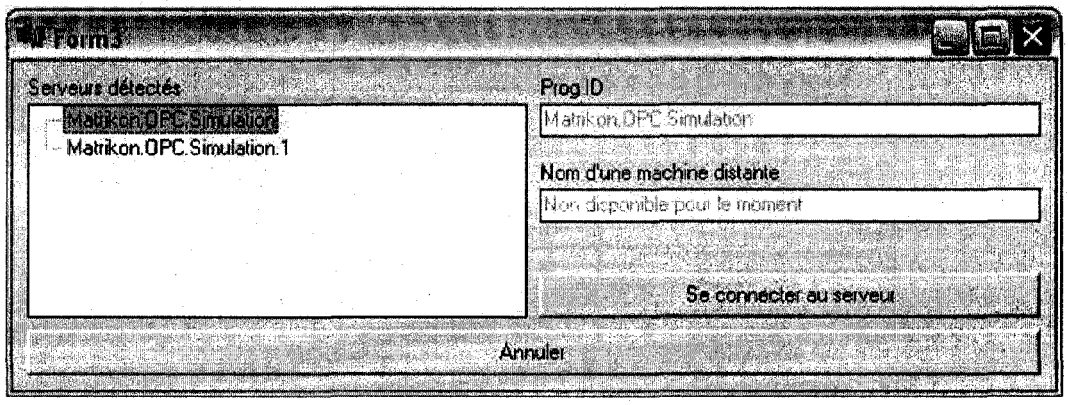


Figure 34 Sélection d'un serveur OPC

4.3.3.3 Création de groupes

L'étape suivante consiste à organiser l'espace de travail du client en créant des objets OPCGroup. Un groupe peut être actif ou inactif. Ce paramètre indique au serveur s'il doit maintenir ou non les données concernant le groupe dans sa mémoire virtuelle, optimisant ainsi l'utilisation des ressources de la machine. Cette fonctionnalité permet aussi d'éviter la suppression de groupe dont on n'a pas momentanément l'utilité.

C'est aussi au niveau du groupe que l'utilisateur va spécifier le taux de rafraîchissement des données qu'il souhaite visualiser. Cette propriété signifie que le client demandera les valeurs des items contenus dans ce groupe avec cet intervalle de temps. Cet intervalle sera le même pour tous les items du groupe. Dans certains cas, les services fournissant les valeurs peuvent se situer dans un autre fuseau horaire. Il est possible au niveau du groupe de prendre en compte ce décalage horaire.

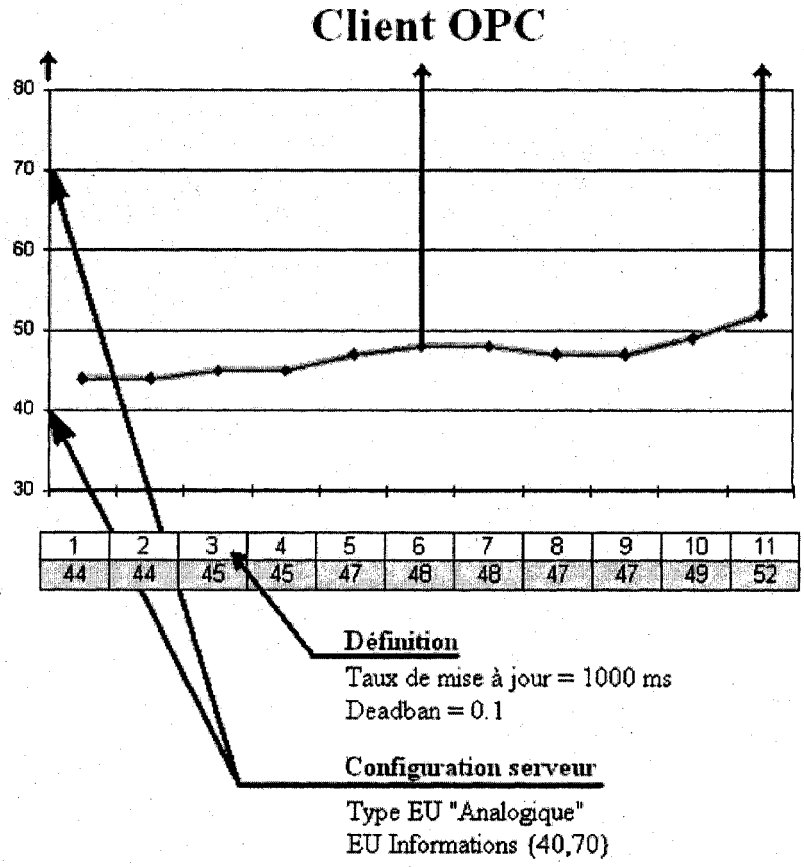


Figure 35 Utilisation et paramétrage de la Deadband

Au moment de la définition de l'espace de noms, il est possible de configurer des types d'unité (appelé type *Engineering Unit* ou type EU) et les informations les concernant. Ce paramètre a une importance dans le processus d'échange de données entre le client et le serveur. En effet, à la création d'un objet OPCGroup, le client spécifie la valeur d'un pourcentage appelé bande morte (*DeadBand*). Si à la définition de l'espace de noms, la variable a été enregistrée avec un type « analogique » et que les informations correspondantes ont été saisies (voir Figure 35), le serveur suit le processus suivant. Le serveur calcul la valeur absolue de la différence entre les bornes de EU-Informations (ici elle vaut 30) puis il multiplie le résultat par la valeur de *Deadband* de l'objet OPCGroup (ici 0.1, le résultat vaut alors 3). L'intervalle entre les lectures est déterminé par le taux

de mise à jour (ici 1000 ms). La valeur absolue entre la dernière valeur envoyée et la valeur actuelle est calculée cycliquement. Si cette différence est supérieure au résultat calculé juste avant (avec les unités de mesure et le *Deadband*) alors la valeur actuelle est envoyée au client. Ce processus est uniquement valable pour les variables dont le type EU est « Analogique » et n'est pas applicable pour les variables de type Chaîne de caractère.

L'objet OPCGroup permet à un client de créer et de manipuler les objets OPCItem. L'objet OPCGroup va mémoriser les pointeurs vers les interfaces implémentées par le serveur. Ces différentes interfaces sont présentées dans la Figure 36. Ces interfaces permettent au client de créer et de supprimer des items des groupes, de gérer le rafraîchissement des variables (taux de rafraîchissement) et de lire et d'écrire un ou plusieurs items du groupe.

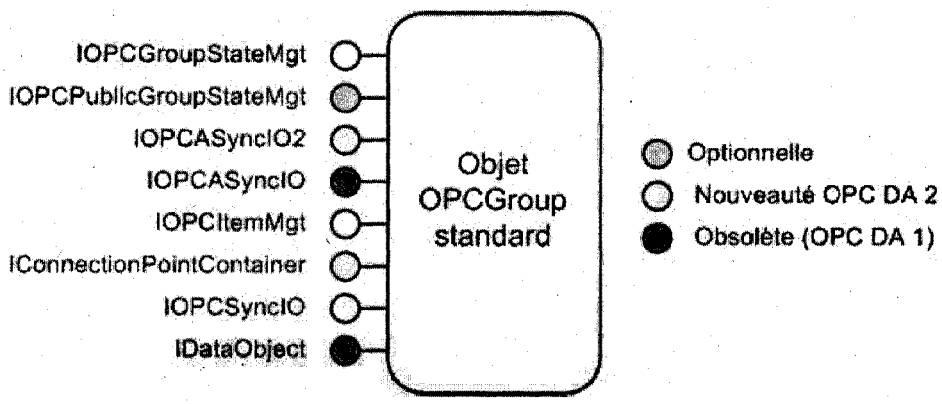


Figure 36 Interfaces de l'objet OPCGroup

La création d'un groupe va consister à instancier l'interface *IOPCGroupStateMgt* et à mémoriser les paramètres suivants :

- **Name** : une chaîne de caractères représentant le nom du groupe.

- **UpdateRate** : un entier non signé représentant le taux de rafraîchissement en ms.
- **LanguageID** : identification locale (LCID) représentant la langue utilisée pour cette session client/serveur.
- **Active** : booléen représentant l'état du groupe.
- **TimeBias** : entier long représentant le décalage horaire entre les sources de données et l'utilisateur.
- **Deadband** : réel représentant le pourcentage de la bande morte.
- **hServer** : entier non signé représentant le handle utilisé par le serveur pour ce groupe.
- **Valid** : booléen représentant le succès de l'ajout du groupe au serveur.
- **pIGroupStateMgt** : pointeur d'interface IGroupStateMgt.
- **pICPContainer** : pointeur d'interface *IConnectionPointContainer*.
- **pServer** : pointeur d'objet OPCServer pointant sur l'objet parent de ce groupe.

L'objet développé contient en plus de cela un constructeur permettant de l'initialiser ainsi que les fonctions permettant de lire et d'assigner les variables membres. Elle contient en plus une fonction permettant d'initialiser l'objet OPCGroup à l'aide de l'état de ce groupe demandé au serveur.

L'ajout d'un groupe est assez simple, l'interface utilisateur est présentée à la Figure 37. Il suffit de renseigner les différents champs ou de laisser les paramètres par défaut. Pour tester le bon fonctionnement du groupe, il est possible de demander le statut du groupe au serveur.

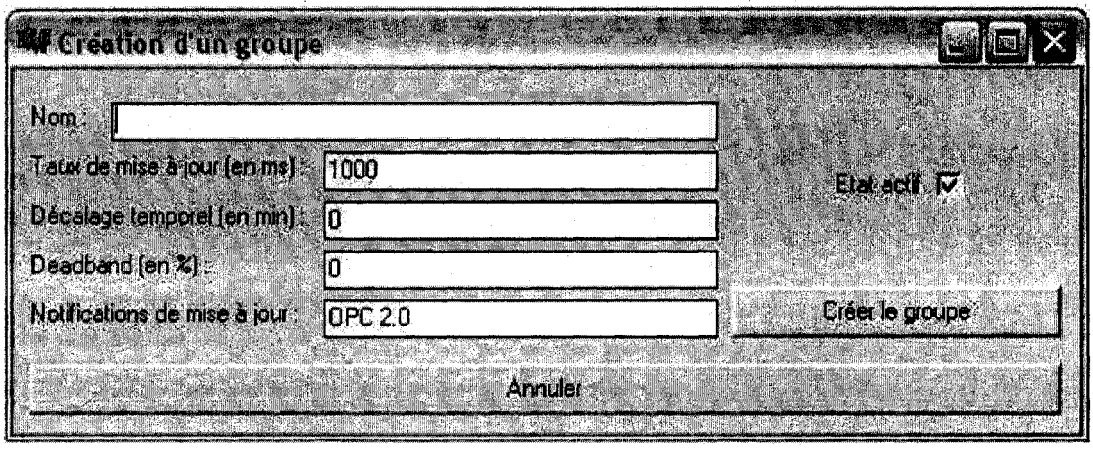


Figure 37 Création d'un groupe

4.3.3.4 Exploration d'un espace de noms et création d'un item

La prochaine étape logique consiste à créer les items. L'item du client est en lien direct avec celui du serveur. Il faut donc d'abord être en mesure d'explorer l'espace de nom du serveur. L'espace de noms et son exploration ont évolué avec la spécification OPC DA. Même s'il existe 2 types d'espaces de noms, leurs explorations reposent sur une interface commune, l'interface *IOPCBrowseServerAddressSpace*. Un espace de noms hiérarchique est constitué de nœuds ou branches utilisés dans un but structurel et d'item ou feuille représentant les sources de données. Les évolutions de la spécification ont apporté des fonctions permettant d'être plus efficace dans l'exploration de l'espace de noms du serveur. Mais pour rester compatible avec la version 1.0 de OPC DA nous avons gardé les méthodes définies lors de cette première version de la spécification. Les serveurs OPC DA 1 nous obligent, lors de la navigation de branche en branche, de repasser très souvent par la racine (Figure 38).

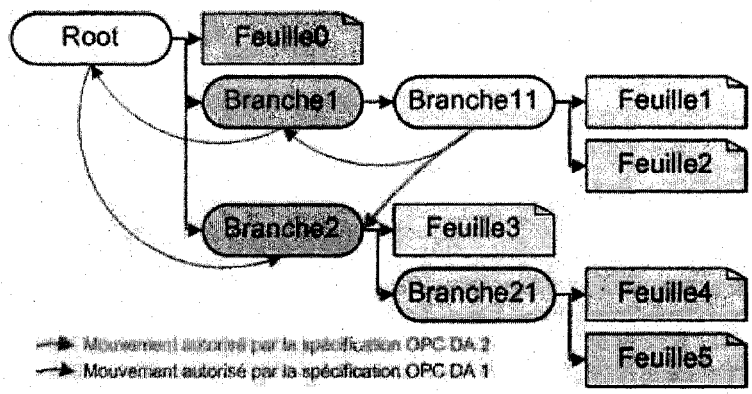


Figure 38 Exploration dans la spécification OPC DA 1 et 2

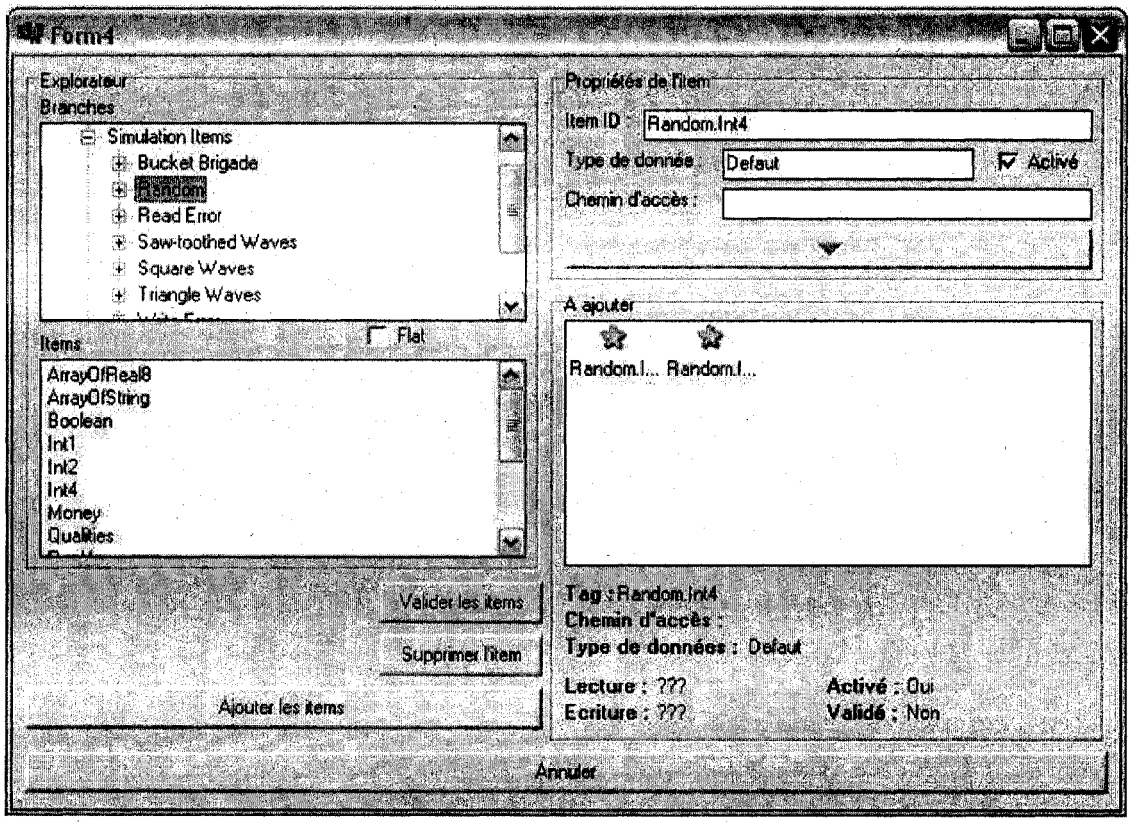


Figure 39 Exploration de l'espace de noms et ajout d'un item

Pour rendre le chargement plus rapide et minimiser l'utilisation de la mémoire, l'exploration se fait pas à pas, suivant les orientations de l'utilisateur, les branches découvertes étant représentées dans une vue arborescente et les items dans une vue de type liste. L'interface utilisateur est présentée à la Figure 39.

La vérification de la bonne création d'un item est relativement simple, il faut être en mesure de le lire.

4.3.3.5 Lecture d'un item

OPC DA propose plusieurs types d'échange de données (Tableau VI). Les lectures synchrones ou asynchrones peuvent se référer à la mémoire cache ou directement à la source de données. La lecture de la mémoire cache n'est possible que pour les objets OPCGroup et OPCItem actifs. Dans ce cas, la mémoire cache est mise à jour régulièrement en fonction de l'intervalle défini pour l'objet OPCGroup. Avec une lecture synchrone, le client appelle la méthode de lecture et attend que le serveur lui renvoie la valeur. Cette méthode n'est donc à utiliser que si le retour des valeurs ne prend pas beaucoup de temps sinon le client peut rester bloqué un certain temps. Avec une lecture asynchrone, le client appelle la méthode de lecture et continue de fonctionner. Sa requête entre dans la file d'attente du serveur. Après un certain temps, dépendant du type d'accès aux données et du nombre de tâches dans la file d'attente, le client obtient la valeur désirée par l'intermédiaire d'une interface que le client doit mettre à disposition du serveur. Les lectures asynchrones sont à utiliser si l'enregistrement des données par le serveur prend un certain temps. Avec le rafraîchissement, le client lit tous les objets OPCItem actifs d'un objet OPCGroup actif. Pour ces trois types de lecture, le client doit prendre l'initiative. Ce n'est pas le cas si le client veut recevoir les valeurs en fonction des possibles changements de ces dernières. Il est parfois important de transférer une valeur depuis le serveur vers le client si la valeur a changé. Pour cette raison il existe un quatrième type d'échange, la soumission.

Le serveur lit les valeurs à intervalle régulier déterminé par le taux de mise à jour et les passe au client si elles ont changé en valeur ou en statut. La Figure 40 présente l'interface utilisateur permettant de visualiser la lecture des données.

Tableau VI

Échanges de données autorisés

	Cache	Device
Lecture synchrone	x	x
Lecture asynchrone	x	x
Rafraîchissement	x	x
Soumission		

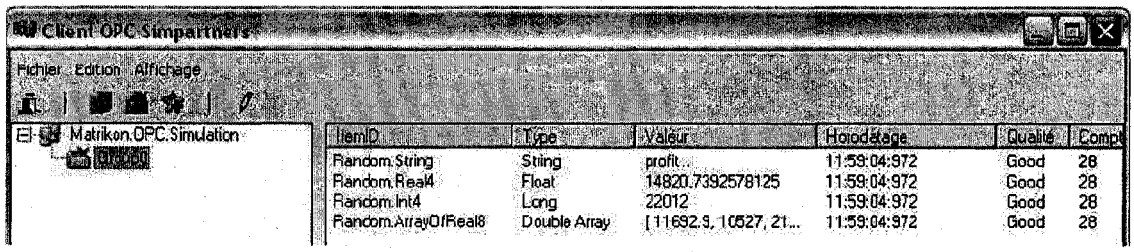


Figure 40 Lecture des items

4.4 Conclusion

Le standard OPC permet l'acquisition de données en temps réel. Il est de plus en plus intégré nativement dans les applications utilisatrices des données du processus manufacturier. Nous avons vu les différentes spécifications maintenues par la Fondation OPC et nous avons identifié la spécification Data Access comme la plus appropriée pour

accéder aux données en temps réel. Au travers du développement d'un client OPC DA nous en avons profité pour présenter les particularités du fonctionnement des serveurs et des clients OPC DA. Pour finir, nous avertissons que le client OPC développé ici a été intégré au simulateur SDBuilder et que pour des raisons de confidentialité, le code du client n'a pas été mis en annexe de ce mémoire.

CHAPITRE 5

SCÉNARIO D'ORDONNANCEMENT RÉACTIF – MODULE D'OPTIMISATION EN TEMPS RÉEL

5.1 Introduction

Le module d'optimisation en temps réel est une des pierres angulaires de l'architecture. C'est en parti grâce à lui que l'architecture proposée peut supporter les processus adaptatifs et c'est un élément critique du système. Il est en effet très dépendant des données des autres systèmes et il se doit d'être suffisamment rapide pour supporter la prise de décision dans un laps de temps en accord avec la vitesse du processus manufacturier. Il a besoin d'interroger l'ERP et le MES pour travailler avec les données les plus actuelles possible et dans certains cas il devra retourner les résultats de ses investigations à l'un ou l'autre de ces systèmes, voire les deux.

Dans le scénario démonstratif présenté ici, nous désirons être en mesure d'avoir un ordonnancement réactif. Les creusets d'aluminium se présentent devant les fours de mélange dans une séquence incontrôlée, ce qui fait que nous évoluons dans un environnement très incertain. Le module d'optimisation présenté ici est très simple et n'est sûrement pas le plus efficace qui soit. Notre volonté est de présenter la viabilité de notre architecture et c'est pour cela que nous avons fait le choix de simplicité. Dans un premier temps, nous vous présenterons la logique de notre algorithme. Nous présenterons ensuite le modèle de simulation utilisé pour évaluer les critères de performance.

5.2 Modèle d'optimisation

Le modèle d'optimisation se focalise sur le centre de coulée, car c'est le goulot d'étranglement de l'atelier. La fonderie modélisée ne dispose que de 2 fours de mélange

qui alimentent une unique lingotière. La séquence que va produire le modèle d'optimisation est destinée aux fours de mélange. La Figure 41 présente la logique du module d'optimisation.

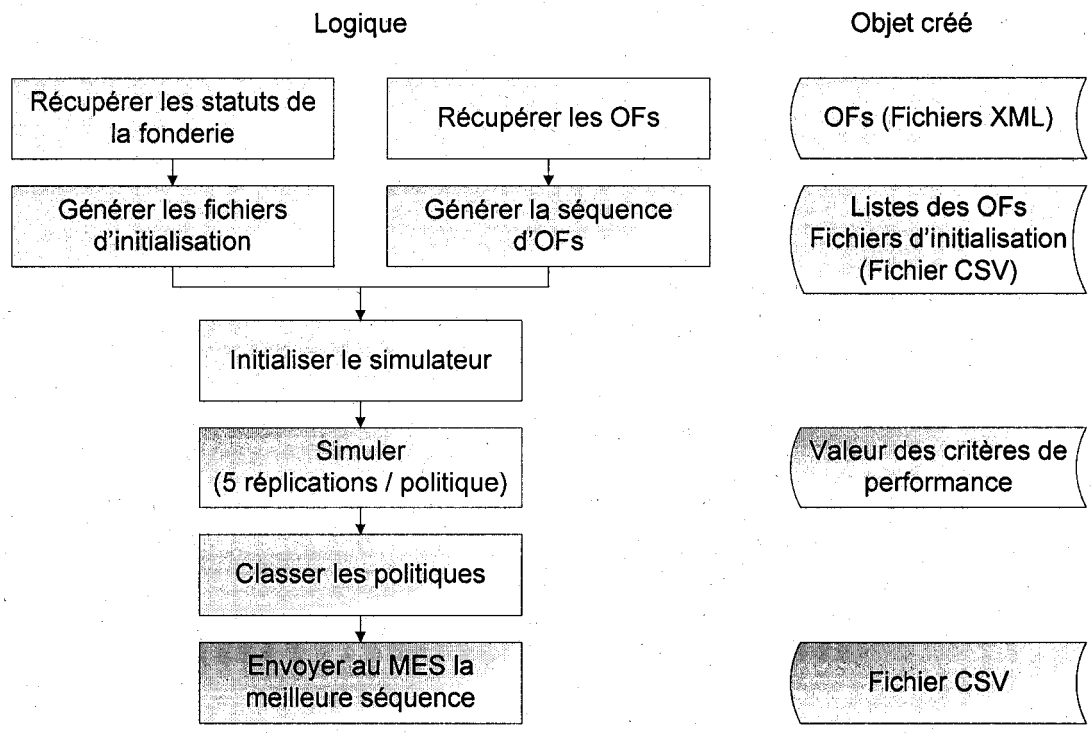


Figure 41 Algorithme d'optimisation

L'algorithme d'optimisation peut être décomposé en 3 sous-processus. La première étape va conduire à l'initialisation du modèle de simulation. La seconde étape correspond à la simulation en elle-même. Nous nous limiterons à 5 réplications par politique de gestion. Ce nombre est suffisant pour démontrer la viabilité de notre architecture, mais rien n'empêche de l'augmenter. La dernière étape permet de choisir la meilleure séquence de production testée.

5.2.1 En amont de la simulation

Le module d'optimisation va commencer par récupérer un certain nombre de données. Il demande les statuts de la fonderie au MES en utilisant le client OPC présenté au Chapitre 5. En utilisant ce client, il pourrait tout autant directement interroger le processus manufacturier. Si un MES est disponible, il est aussi possible d'interroger directement sa base de données. Avec les données récupérées, nous générons un ensemble de fichiers CSV destinés à l'initialisation du modèle de simulation.

Après l'obtention des statuts de l'atelier, le module d'optimisation demande la liste des ordres de production de l'ERP en utilisant la fonction présentée au Chapitre 4. Il récupère alors 3 fichiers XML (1 par alliage) contenant pour chacun des ordres le numéro d'ordre, l'alliage, la quantité Q_d demandée et la date de besoin T_b. Ces fichiers sont chargés par l'application, et l'ordre en cours de fabrication sur l'autre four est écarté de la liste. À partir des données recueillies, nous allons évaluer les valeurs suivantes en unité de temps du simulateur (une unité de temps représente une heure) :

- le temps de production dans le centre de coulée, D_p^{sim}, correspondant au temps nécessaire au mélange de l'alliage augmenté du temps de coulée (Formule 6.1) ;
- la durée nous séparant de la date de besoin, T_r^{sim}, exprimée en heure. La fonction f calcule le nombre de jours entre deux dates (Formule 6.2) ;
- la marge de temps nous restant pour produire, M_p^{sim} (Formule 6.3).

$$D_p^{sim} = D_{mélange}^{sim} + Q_d \times 0{,}25 \tag{5.1}$$

$$T_r^{sim} = f\left(T_{now}, T_b\right) \times 24 \tag{5.2}$$

$$M_p^{sim} = D_p^{sim} - T_r^{sim} \tag{5.3}$$

À partir de cet instant, nous avons tous les éléments en main pour générer les séquences d'ordres en appliquant les politiques retenues. Nous utilisons ici 5 politiques :

- EDD (*Earlier Due Date*) : les ordres sont classés par ordre croissant de leur date de besoin *Tb* ;
- Marge : les ordres sont classés par ordre croissant de leur marge M_p^{sim} ;
- SPT (*Shortest Processing Time*) : les ordres sont classés par ordre croissant de leur temps de production D_p^{sim} ;
- LPT (*Largest Processing Time*) : les ordres sont classés par ordre décroissant de leur temps de production D_p^{sim} ;
- NOF (Numéro d'ordre de fabrication) : les ordres sont classés par ordre décroissant de leur numéro de production. Cela correspond à l'ordre de leur génération.

Pour chacune de ces politiques, nous générons un fichier CSV (*Comma-Separated Values*) contenant la séquence des ordres qui sera utilisée par le simulateur.

Quand toutes ces étapes sont achevées, la simulation peut débuter. Nous verrons le modèle de simulation plus en détail par la suite. Ce modèle va procéder à 5 réplications pour chaque politique. À chaque réplication, les valeurs des critères d'évaluation de nos politiques de gestion sont sauvegardées. Pour rappel, les critères d'évaluation sont les suivants :

- le pourcentage de volume d'aluminium déclassé (à minimiser) ;
- l'écart entre la quantité produite et la quantité prévue initialement (à minimiser) ;
- le temps nécessaire pour exécuter la séquence des ordres (à minimiser).

5.2.2 En aval de la simulation

Les résultats de chaque réplication sont obtenus sous 2 formes. La trace de tous les événements et de tous les changements d'état des instances des objets de la simulation est sauvegardée dans le format CSV et reste à disposition du module d'optimisation ou d'une tierce application si besoin est. Les valeurs des critères d'évaluation des politiques sont envoyées par message à l'application qui les garde en mémoire.

Quand toutes les réplications sont achevées, il est possible de classer les politiques en fonction de leurs résultats. Là encore, la simplicité est de mise, mais rien n'empêche de raffiner le modèle. Les politiques sont classées pour chacun des critères. Les classements se basent sur la moyenne des résultats de chaque réplication. En fonction du classement obtenu, la politique obtient un certain nombre de points. Les points sont calculés comme suit. Soit n le nombre de politiques et p la position de la politique dans le classement. La politique est alors créditée de $n+1-p$ points. Dans notre cas, nous avons 5 politiques. Une politique terminant à la deuxième place est créditée de 4 points (5+1-2). Tous les critères ont le même poids, ils sont jugés de même importance. Une politique sera alors considérée comme meilleure que les autres si elle accumule plus de points que les autres politiques.

La séquence établie avec la meilleure politique est finalement envoyée au MES qui sera chargée de l'appliquer. Ce transfert peut se faire soit par fichier, soit par message, soit par requête sur sa base de données.

Le système manufacturier étant assez complexe, nous utilisons un modèle de simulation pour évaluer les performances des politiques de gestion. Nous allons maintenant présenter ce modèle.

5.3 Modèle de simulation

SDBuilder est le simulateur que nous avons utilisé dans notre scénario. C'est un simulateur très souple qui fonctionne à base d'objets et de règles déterminant les changements d'état des instances de ces objets. Nous allons commencer par présenter les objets du modèle de simulation, leurs paramètres et leur utilité. Nous verrons aussi les hypothèses de modélisation que nous avons faites. Nous parlerons ensuite des règles qui régissent le fonctionnement du modèle.

5.3.1 Rappel et notations

Nous rappellerons au lecteur que, suite à la configuration de l'ERP, notre aluminerie fabrique trois alliages A, B et C de qualité décroissante. Suivant la qualité de l'aluminium produit dans les cuves d'électrolyse, l'utilisation qui en est faite est différente. Un aluminium de grande qualité peut être utilisé pour fabriquer l'alliage A de haute qualité. Il peut aussi être utilisé pour fabriquer les autres alliages, mais c'est alors un cas de déclassement. L'aluminium de qualité moyenne est utilisé pour fabriquer l'alliage B et peut être déclassé pour fabriquer de l'alliage C. Enfin l'aluminium de basse qualité ne peut être utilisé que pour l'alliage C. La qualité de l'aluminium sera représentée par les lettres A, B et C faisant ainsi référence au meilleur alliage qu'il puisse fabriquer.

Comme nous l'avons précisé précédemment, une unité de temps dans le simulateur représente une heure dans la vie courante.

Dans la suite du chapitre le caractère $ est utilisé comme indice de l'ensemble {A, B, C}. Son utilisation est la suivante. Quand nous décrivons le paramètre *A_faire_$* comme la quantité d'alliage $ produite, cela signifie qu'il existe en fait 3 paramètres *A_faire_A*, *A_faire_B* et *A_faire_C*, un pour chacun des alliages.

5.3.2 Objets

SDBuilder simule le fonctionnement des objets. Ces objets sont totalement personnalisable que ce soit dans les paramètres qu'ils contiennent, les règles où ils interviennent, la représentation graphique qu'ils utilisent. Dans l'ensemble des objets utilisés, il y en a un qui est particulier, qui représente le système simulé. Les autres objets vont servir à composer ce système. C'est aussi à cet objet que seront attachées les règles de fonctionnement.

5.3.2.1 Objet système une_usine

L'objet une_usine représente le système de production. Il sera composé des autres objets et contiendra les règles que nous présenterons plus tard. Cet objet possède ses propres paramètres (Figure 42) qui ont deux utilités distinctes. Il y a les paramètres qui servent à évaluer les critères de performance et il y a les paramètres généraux du système. Ces derniers paramètrent le nombre de cuves que peut contenir un creuset (*Nb_cuves_dans_creuset*), le nombre de creusets que peut contenir un four (*Nb_creuset_dans_four*) et les tolérances acceptées sur l'heure de vidange d'une cuve dans le but de maximiser l'utilisation des creusets (*Tolérance_avant* et *Tolérance_après*).

Les paramètres *A_faire_$* représentent les quantités produites pour chaque alliage. *Tonnes_A_dans_B*, *Tonnes_A_dans_C* et *Tonnes_B_dans_C* représentent les tonnes d'alliage déclassées. Les paramètres *Tonnes_$_avant_four* ont été introduits pour évaluer plus finement le pourcentage d'aluminium déclassé. Nous en reparlerons après. Les paramètres *Scrap_$* comptabilisent les tonnes d'aluminium qui ont dû être jetées faute d'avoir pu être utilisées à temps.

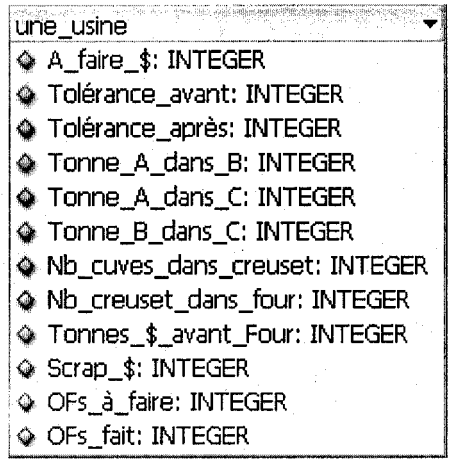


Figure 42 Paramètres d'une fonderie

Enfin, les deux paramètres *OFs_à_faire* et *OFs_fait* ont été introduits pour conditionner la fin de la simulation.

5.3.2.2 Objet une_cuve

Cet objet représente les cuves électrolytiques. Ses paramètres sont ceux de la Figure 43. Ils sont au nombre de 240. Les différentes instances sont identifiées par le paramètre *id_cuve*. Les cuves doivent être siphonnées toutes les 24heures. L'heure de siphonnage d'une cuve est stockée dans le paramètre *Heure_siph*. L'aluminium qu'elles contiennent a certaines propriétés chimiques et pourra être utilisé dans la composition de l'alliage du paramètre *Produit*. Une cuve peut être dans 3 états qui représentent son cycle de vie : en électrolyse, à siphonner et siphonnée. Une fois siphonnée, la cuve va être réutilisée pour l'électrolyse. Le nouveau contenu est déterminé grâce à l'histogramme *Get_property_cuve*. Dans 50% des cas, la cuve contiendra de l'aluminium de qualité A, dans 30% des cas elle contiendra du B et dans 20% des cas ce sera du C. Le paramètre *Destination* permet de retracer le creuset dans lequel la cuve a été vidangée.

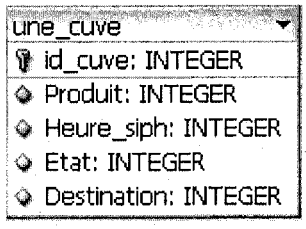


Figure 43 Paramètres d'une cuve

5.3.2.3 Objet un_creuset

Cet objet représente les creusets servant à transférer le métal en fusion des cuves jusqu'au four. Ses paramètres sont ceux de la Figure 44. Ils sont au nombre de 20. Les différentes instances sont identifiées par le paramètre *Numéro*. Un creuset peut être dans 3 états qui représentent son cycle de vie : libre, en siphonnage, à vider. Un creuset libre est en attente de cuves à siphonner. Un creuset ne peut siphonner que des cuves ayant le même type d'aluminium. L'aluminium n'est donc pas déclassé à cette étape. La première cuve siphonnée assigne le paramètre *Contenu* du creuset. La quantité de cuves que peut contenir un creuset est définie au niveau du système et le paramètre *Qté_en_cuve* représente son niveau de remplissage. Bien que les creusets ne soient pas chauffés, ils disposent d'une marge de temps pour siphonner d'autres cuves ou pour attendre d'utiliser l'aluminium qu'ils contiennent, c'est l'utilité des paramètres *Heure_min* et *Heure_max*. Une fois plein, ou dans la plage d'utilisation de son contenu, il va être utilisé pour répondre à la demande. Le déclassement est alors permis. Les paramètres *Destination_type* et *Destination_num* servent à retracer le four qui reçoit son contenu. S'il ne peut être utilisé pour produire un alliage, il peut être vidé dans le four d'attente. Si ce dernier est plein, l'aluminium est comptabilisé comme un rebut. Le paramètre *MAJ* est utilisé à l'initialisation des creusets. Nous en reparlerons après.

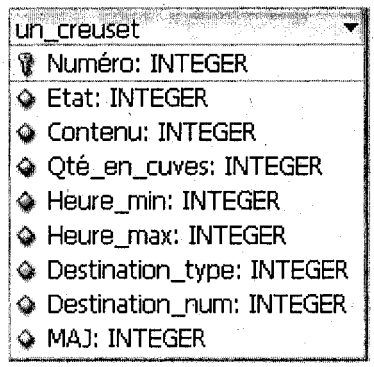


Figure 44 Paramètres d'un creuset

5.3.2.4 Objet un_four

Cet objet représente deux types de four (paramètre *Catégorie*). Dans notre modèle il existe deux fours servant à mélanger les alliages avant de couler les lingots. Il y a aussi un four d'attente qui permet de vider les creusets non utilisables en production pour le moment. Les paramètres sont ceux de la Figure 45. Les différentes instances sont identifiées par le paramètre *Numéro*. Les paramètres *Contenu*, *Qté_à_faire* et *Num_lot* font référence à l'alliage, à la quantité à produire et au numéro de l'OF que le four de mélange est en train de traiter. Le paramètre *Contenu* sert aussi à identifier le contenu du four d'attente. Le paramètre *Etat* n'est utile qu'aux fours de mélange. Avant d'être alloué à un OF, ces fours sont dans l'état libre. Ils passent alors en phase de remplissage puis de mélange pendant *Mélange_durée* unités de temps. Le mélange est de durée fixe quelque soit la taille de l'ordre. Le fait d'avoir cette durée en paramètre permet d'initialiser un four dans cet état au début de la simulation. Quand le mélange est terminé, le four passe en état mélangé et attend que le centre de coulée se libère. Quand le centre de coulée est libre, le four commence la coulée. Il retourne après dans l'état libre. Le paramètre *Aux_qté* est un intermédiaire de calcul au moment du remplissage. Il permet d'évaluer la quantité manquante pour que le four complète son OF. Le paramètre *MAJ* est utilisé à l'initialisation des fours. Nous en reparlerons après.

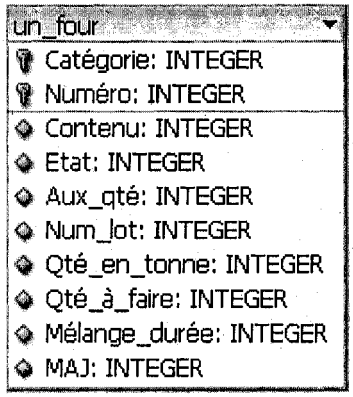


Figure 45 Paramètres d'un four

5.3.2.5 Objet un_OF

Cet objet représente les ordres de fabrication. Ses paramètres sont ceux de la Figure 46. Leur nombre est variable. Les différentes instances sont identifiées par le paramètre *Numéro* qui correspond au numéro d'OF attribué par l'ERP. Les paramètres *Contenu*, *Qté_en_tonnes* font référence à l'alliage, à la quantité à produire. Les paramètres *Date_Début* et *Date_fin* servent à retracer le temps de production de l'OF, de l'allocation à un four à la coulée de l'alliage. Le paramètre *IndexOF* représente la place de l'OF dans la séquence de production que les fours doivent suivre.

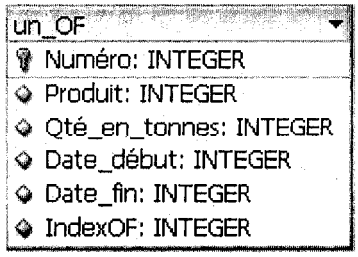


Figure 46 Paramètres d'un OF

5.3.3 Règles

Un certain nombre d'instances de chacun des objets vont être créées. La vie de ces instances est rythmée par les règles du modèle de simulation. Ces règles définissent les types de ressources sur lesquelles elles vont agir, la manière de choisir la ressource qui va être affectée et enfin les modifications qui vont être apportées à la ressource. Ces règles sont toutes décrites en détail dans l'Annexe 8. Les cycles de vie des différentes instantes sont représentés au travers des différentes règles du système dans la Figure 47.

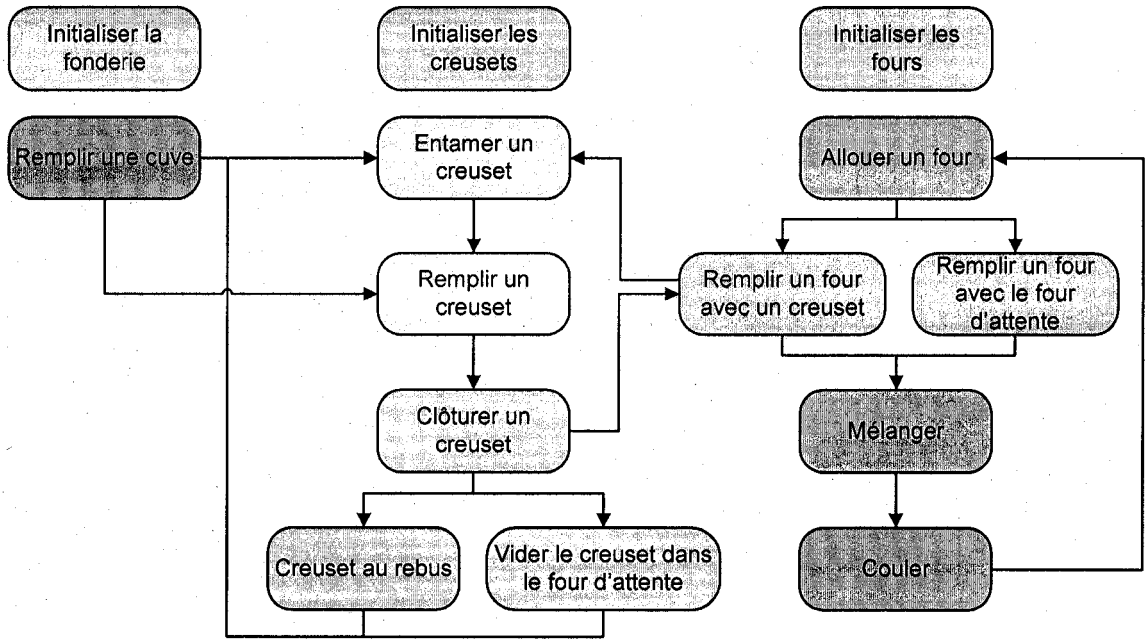


Figure 47 Règle de fonctionnement du modèle de simulation

5.3.3.1 Initialisation du modèle

L'ERP fournit la liste des OFs à produire, le MES fournit les statuts de la fonderie et le modèle de simulation doit tenir compte de tous ces éléments. Les creusets et les fours

sont facilement inutilisables grâce à leurs divers paramètres. La séquence de production fournie par le module d'optimisation ne doit pas omettre les deux ordres en cours de traitement par les fours de mélange. La fonderie est simplement initialisée avec le nombre d'OFs à traiter.

La difficulté de l'initialisation des fours et des creusets est liée au logiciel et à l'interface graphique présentée à l'utilisateur. En effet dans SDBuilder, les instances des objets ayant une représentation graphique doivent être initialisées au moment de la construction du modèle alors que les instances non visibles peuvent être initialisées au début de la simulation grâce aux fichiers CSV. Pour cette raison nous avons créé des instances miroirs de toutes les instances visibles. Ces instances miroirs ne sont pas visibles et peuvent donc être initialisées uniquement au moment de la simulation. Au tout début de la simulation, les règles d'initialisation sont appliqués et copient simplement les paramètres des instances miroirs dans les instances visibles et passent le paramètre *MAJ* de ces instances à 1.

5.3.3.2 Cycle de vie d'une cuve

La vie des cuves est rythmée par leur heure de siphonnage. Quand cette heure arrive, la cuve est vidée dans un creuset vide (entame du creuset) ou dans un creuset contenant de l'aluminium de même qualité. Une fois la cuve siphonnée, il sert à nouveau pour récupérer l'aluminium. La fonction *Get_property_cuve* est appliqué pour déterminer le contenu et une nouvelle heure de siphonnage est évaluée en ajoutant 24 heures à la précédente.

5.3.3.3 Cycle de vie d'un creuset

Un creuset libre est entamé par une cuve prête à être siphonnée. Le creuset peut alors uniquement siphonner des cuves de contenu identique, et ce, dans un laps de temps limité. Dans ce laps de temps, le creuset peut être directement utilisé dans les fours de

mélange pour répondre à la demande. Si ce laps de temps s'écoule et que le creuset n'est pas utilisé, il peut être vidé dans le four d'attente. Si ce dernier est plein, le contenu du four est envoyé au rebut. L'aluminium du creuset peut être déclassé lors de son utilisation dans les fours de mélange ou d'attente.

5.3.3.4 Cycle de vie d'un four

Le four d'attente reçoit les creusets arrivant à la limite temporelle de leur utilisation. Le contenu du four est du type de l'aluminium de qualité la plus basse qui a été versé dedans et sert tant que possible à répondre à la demande des fours de mélange.

Un four de mélange libre va être alloué à un OF. Les creusets ou le four d'attente à utiliser vont être déterminés de manière à minimiser la fonction *Pénalité* tout en restant positive (Tableau VII). Quand le four contient la quantité indiquée par l'OF, l'alliage est mélangé puis coulé.

Tableau VII

Fonction Pénalité

	en_creuset = A	en_creuset = B	en_creuset = C
en_four = A	0	-1	-1
en_four = B	5	0	-1
en_four = C	10	5	0

5.3.3.5 Évaluation du déclassement de l'aluminium

Les déclassements d'aluminium sont au nombre de 3 : de A vers B, de A vers C et de B vers C. Dans notre modèle de simulation, nous avons identifié 2 types de déclassement. Nous les qualifierons de déclassements directs et indirects. Le déclassement direct est illustré par la Figure 48. Le cas est simple, 33,3% de l'aluminium dans l'état A est déclassé en B et en C et 33,3% de l'aluminium dans l'état B est déclassé en C.

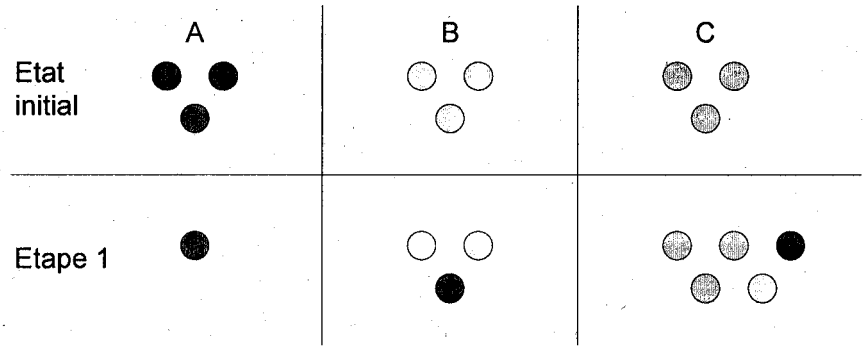


Figure 48 Déclassement direct

La Figure 49 présente un cas de déclassement indirect. La première étape reste inchangée par rapport au premier cas, mais elle est suivie d'une seconde étape de déclassement dans laquelle l'aluminium déclassé en B est alors encore déclassé. Le cas se présente par exemple quand un creuset d'aluminium A est vidé dans le four d'attente contenant de l'aluminium B et est finalement utilisé pour répondre à un OF de qualité C. La question qui se pose est alors la suivante : quels sont les pourcentages de A et de B déclassés en C ?

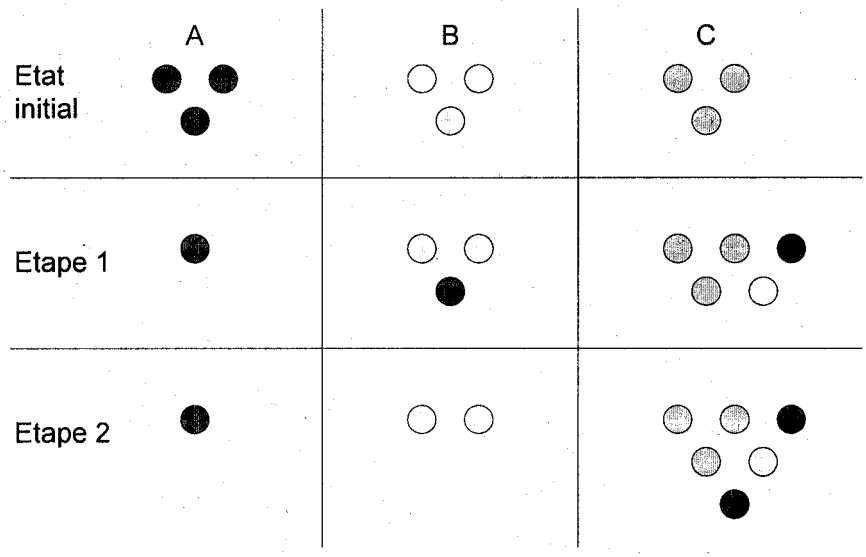


Figure 49 Déclassement indirect

Pour répondre à cette question, nous avons modélisé ce problème comme un pseudo modèle de flux (Figure 50). Le flux entrant en A, B et C valent respectivement 3, 4 et 6. Les flux sortants de A vers B, de A vers C et de B vers C valent respectivement 1, 1 et 2. Le pourcentage de déclassement de X en Y est défini comme étant le rapport du flux sortant de X vers Y sur le flux entrant en X. Les pourcentages de A déclassé en B, de A déclassé en C et de B déclassé en C sont alors respectivement de 33,3% (1/3), 33,3% (1/3) et 50% (2/4). Les paramètres *Tonnes_$_avant_four* évalue les flux entrants tandis que les paramètres *Tonnes_A_dans_B*, *Tonnes_A_dans_C* et *Tonnes_B_dans_C* mesurent les flux sortants.

La présente une capture du modèle de simulation en cours d'utilisation. On y retrouve les différentes instances des objets du modèle présenté ici.

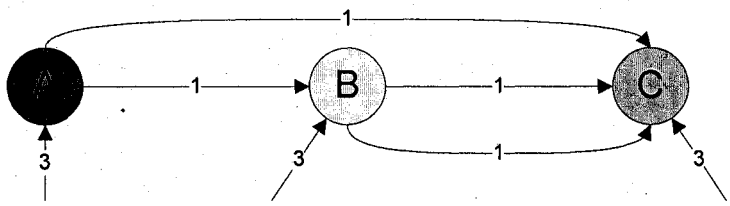


Figure 50 Le déclassement sous forme de flux entrants et sortants

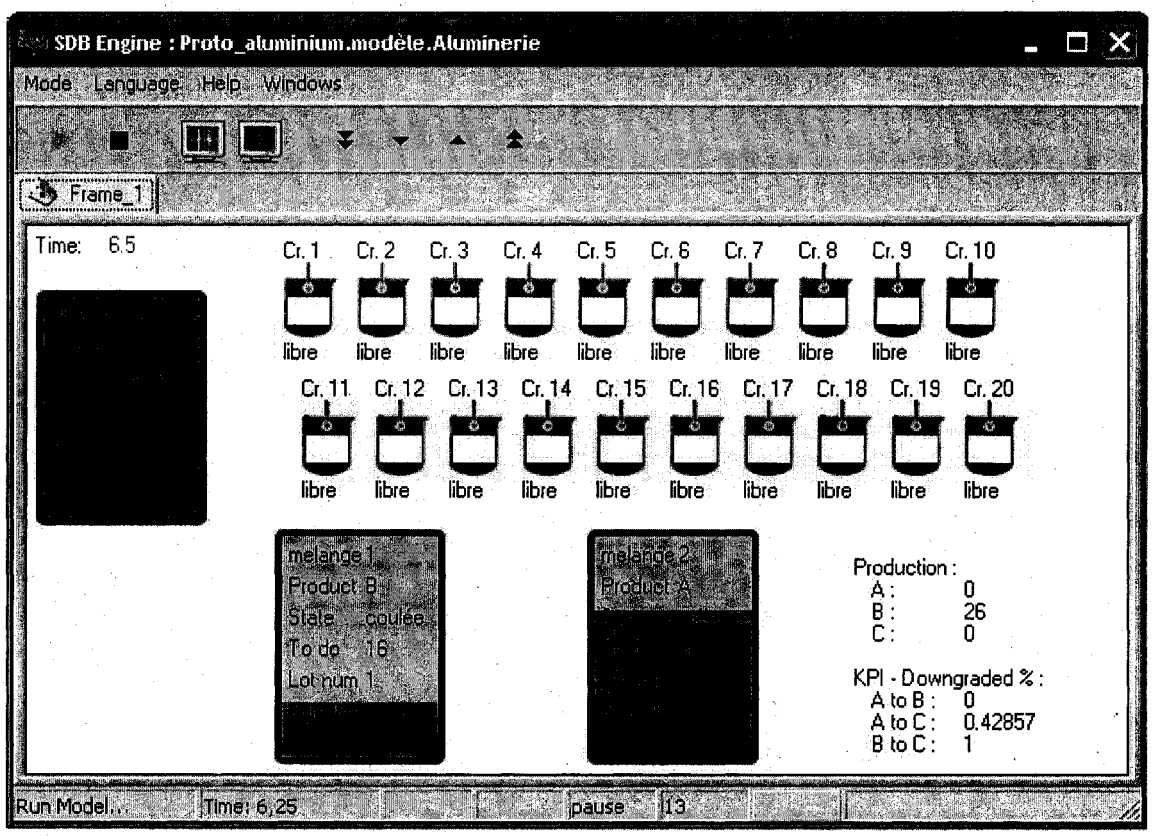


Figure 51 Modèle de simulation de l'aluminerie

5.4 Conclusion

Dans ce chapitre nous avons présenté le module d'optimisation utilisé dans notre scénario démonstratif. Ce module utilise 5 politiques de gestion pour créer des séquences de production. La simulation est utilisée pour évaluer les performances de chaque séquence. Nous procédons à 5 réplications pour tenir compte des phénomènes aléatoires. Les résultats sont compilés et les politiques de gestion sont classées pour chaque critère. Suivant leur classement elles sont créditées d'un certain nombre de points. La politique retenue est celle qui cumule le plus de points et la séquence de production correspondante est alors envoyée au MES.

Le module d'optimisation utilisé est volontairement simpliste. Le but de ce mémoire est de proposer une architecture permettant de supporter les processus adaptatifs et de la tester au travers d'un scénario démonstratif. Ce module d'optimisation est juste là pour démontrer la faisabilité de notre démarche et mettre en lumière le développement inhérent à l'intégration d'un tel module.

Au terme de ce chapitre, nous avons détaillé les différents modules de notre architecture et les composants que nous avons dû développer. Le prochain chapitre est dédié au dispositif d'expérimentation.

CHAPITRE 6

EXPÉRIMENTATION, ANALYSE ET DISCUSSIONS

6.1 Introduction

Au terme du chapitre précédent, nous avons détaillé les différents modules de notre architecture. Nous avons aussi présenté les différents composants d'intégration qui ont dû être développés. Ces composants sont spécifiques à notre scénario et en dehors du client OPC il est plus qu'improbable que nous puissions les réutiliser. Ces différents éléments ont fait l'objet de présentations séparées ce qui peut donner une vision parcellaire de l'architecture globale. Il est temps de prendre un peu de recul et de présenter le dispositif d'expérimentation mis en place pour tester notre architecture.

Dans un premier temps, nous présenterons l'architecture du prototype. Dans cette architecture nous avons dû pallier à l'absence d'un MES. Les conditions du test sont ensuite présentées et le fonctionnement pas à pas du processus adaptatif est alors détaillé.

6.2 Architecture du prototype

L'architecture mise en place pour l'expérimentation est répartie sur plusieurs ordinateurs. Malheureusement, nous n'avons pas complètement pu tester notre architecture telle que nous l'avions conçue faute d'avoir un MES à disposition. Nous avons donc essayé de pallier ce manque en modifiant un peu le prototype (Figure 52). Le premier ordinateur contient le système ERP. Le module de prise de décision en temps réel est sur le deuxième ordinateur. Ce module est composé de l'algorithme décisionnel, d'un client OPC pour dialoguer avec le MES et du JCo, le connecteur Java, pour

dialoguer avec SAP. Le troisième ordinateur simule le processus manufacturier et contient un client et un serveur OPC pour dialoguer avec le reste de l'architecture.

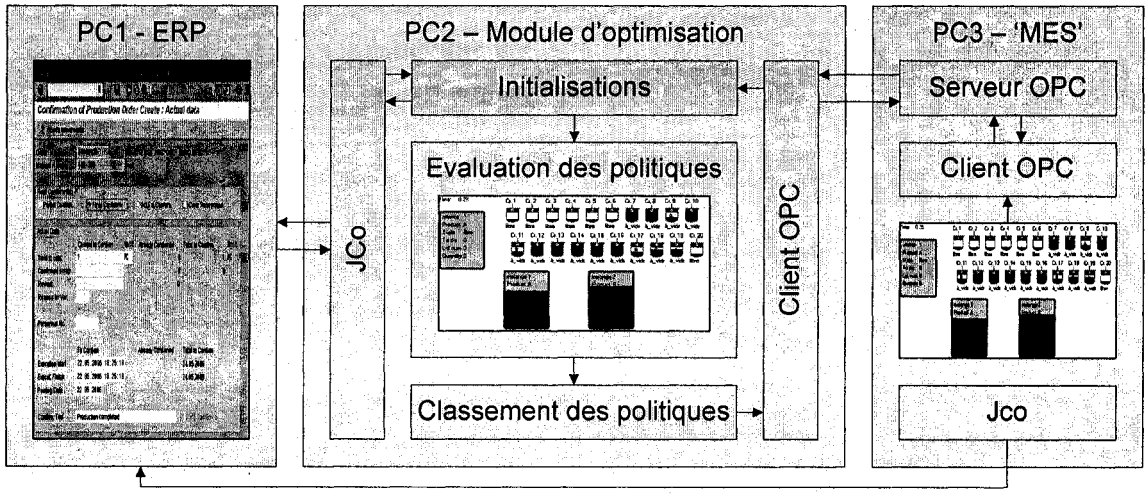


Figure 52 Architecture du prototype d'expérimentation

Le modèle utilisé pour simuler le processus manufacturier est sensiblement le même que celui utilisé dans le module décisionnel. L'échelle temporelle a été étirée. Il a aussi été modifié pour communiquer avec le serveur OPC. Le processus mis en œuvre est décrit à la Figure 53. Quand un ordre de production sort du centre de coulée, le modèle de simulation envoie un message au client OPC indiquant qu'un ordre est achevé. L'ordre est alors confirmé dans SAP et le serveur OPC du MES indique alors qu'un ordre a été confirmé. Le client OPC du module décisionnel est à l'écoute de cette variable et quand sa valeur est modifiée, le processus décisionnel est enclenché. Les ordres de production sont demandés à l'ERP via le connecteur JAVA, les statuts de l'atelier sont demandés via le client OPC. Le modèle de simulation est initialisé à l'aide ces données et le simulateur est lancé. Quand toutes les réplications sont achevées, le module décisionnel classe les politiques de gestion et détermine la plus efficace. La séquence est alors renvoyée au MES pour être appliquée.

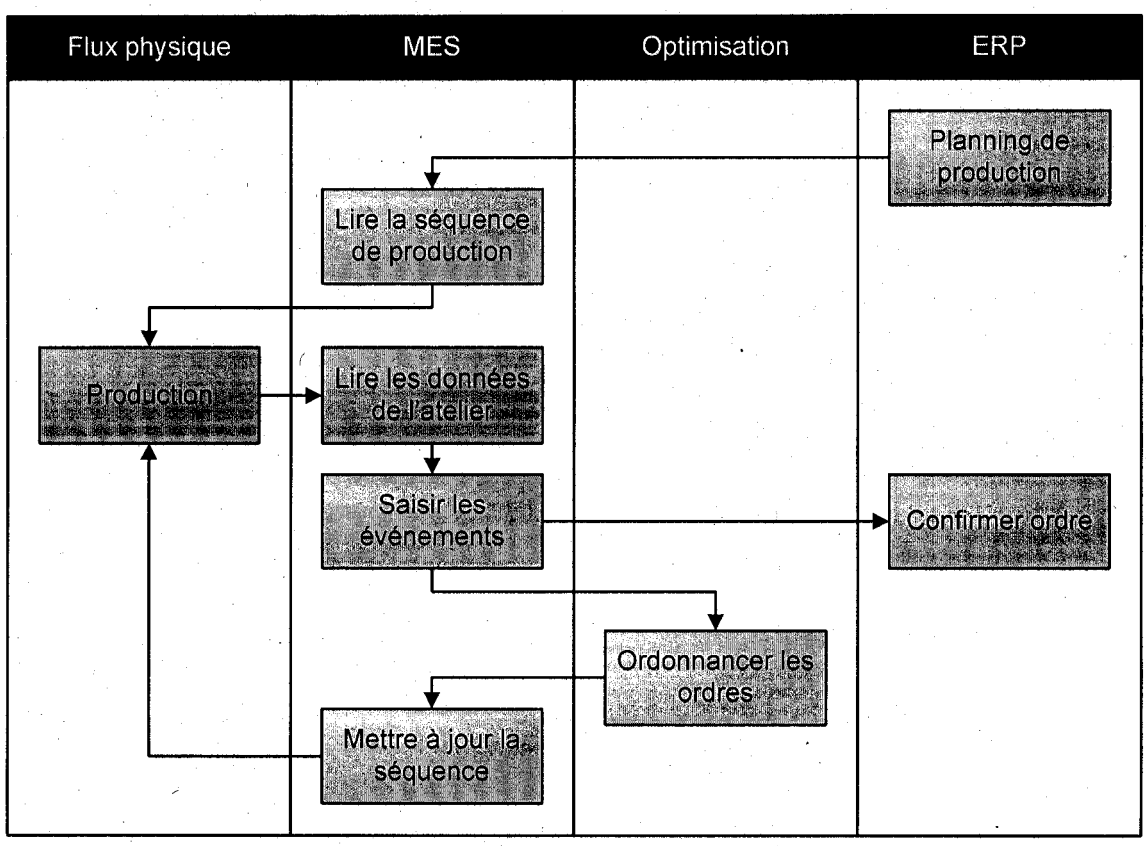


Figure 53 Processus mis en œuvre par le prototype

6.3 Données de test

6.3.1 Les ordres de production

Pour tester le prototype, nous avons créé une quinzaine d'ordres de production dans SAP, 5 pour chaque alliage. Ces ordres ont des dates de besoin différentes et portent sur des quantités différentes aussi. Le Tableau VIII présente les paramètres des ordres de production. La date de besoin est exprimée ici en heure. Par exemple, l'ordre de fabrication n°60003165 doit être terminé dans 105 heures.

Tableau VIII

Liste des ordres de production

Numéro d'OF	Produit	Quantité	Date de Besoin
60003165	A	16	105
60003166	A	14	35
60003167	A	24	85
60003168	A	20	140
60003169	A	8	10
60003170	B	23	45
60003171	B	20	55
60003172	B	17	110
60003173	B	14	80
60003174	B	16	145
60003175	C	24	90
60003176	C	20	150
60003177	C	16	15
60003178	C	18	50
60003179	C	22	115

6.3.2 État initial du processus manufacturier

Nous avons mis à profit les capacités d'initialisation de notre modèle de simulation pour faire en sorte que le prototype ne démarre pas à vide. Tout se passe comme si nous arrivions un instant t de la vie de notre fonderie (Figure 54). Les cuves d'électrolyse sont en train de recueillir de l'aluminium, certains creusets sont vides, d'autres sont pleins et prêts à être utilisés. Le four d'attente est vide, un des fours de mélange est en train de mélanger un alliage et l'autre four est en train de couler son alliage et sera libéré dans 4

unités de temps. Quand cet ordre sera terminé, le processus de prise de décision sera enclenché.

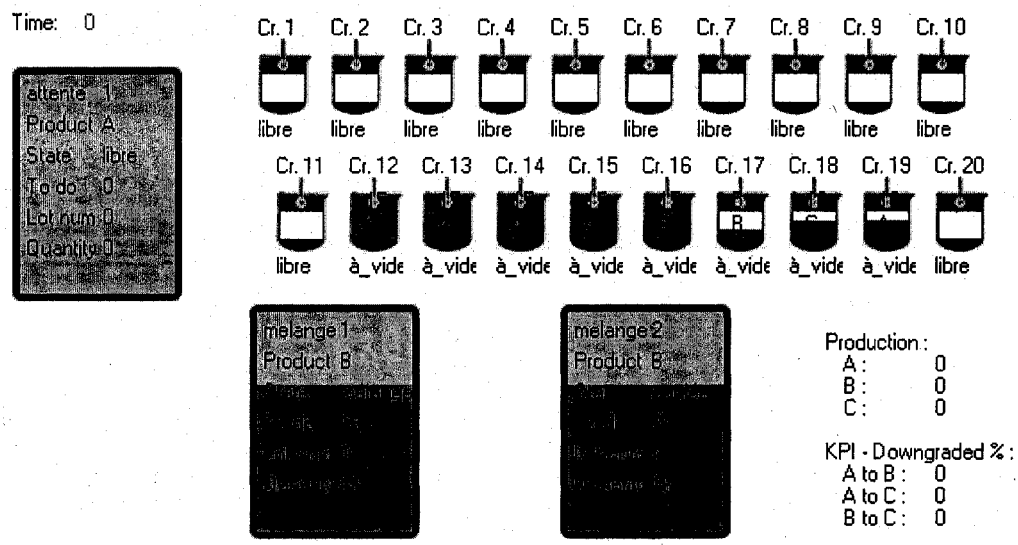


Figure 54 État du processus manufacturier à t=0

6.4 Processus décisionnel pas à pas

Comme nous le disions, le four de mélange n° 2 va se libérer au bout de 4 unités de temps. Un message est envoyé au client OPC signalant la libération du four, le numéro de l'ordre de fabrication complété et la quantité fabriquée. Le connecteur Java utilise ces données pour confirmer l'ordre de production à SAP et en retour signaler la confirmation d'un ordre. C'est le signal pour le module décisionnel de se mettre en action. Il commence par récupérer les statuts de la fonderie et construit les fichiers d'initialisation des cuves, des fours et des creusets. Une fois cette étape achevée il récupère les ordres de fabrication de l'ERP et élimine celui en cours de traitement sur l'autre four de mélange. Pour chaque politique de gestion, il génère la séquence de production et lance 5 réplications. Quand toutes les réplications sont terminées, les résultats sont compilés (Tableau IX) et les politiques sont classées (Tableau X). On

remarquera que le critère évaluant les différences entre les quantités produites et les quantités prévues sont toujours nulles. Cela est dû au modèle de simulation qui ne commence le mélange que si la quantité dans le four correspond à la quantité à produire.

La séquence de la meilleure politique de gestion est envoyée au MES qui la met en application. Justement, ici nous avons deux politiques à égalité, les politiques EDD et LPT. Pour les départager, nous proposons d'ordonner les critères d'évaluation. On regardera d'abord le délai de production puis le déclassement de A en C, le déclassement de A dans B et finalement le déclassement de B dans C. Si elles sont toujours à égalité, c'est que les politiques sont équivalentes. Et on choisira indifféremment l'une ou l'autre. Suivant ce critère, la politique la plus efficace est la politique LPT. Sa séquence de production est transmise au MES par l'entremise du client OPC. Le processus manufacturier, qui attendait de savoir quel ordre produire dans le four de mélange libre, reçoit la nouvelle séquence. Le premier ordre de la séquence est alloué au four et la production débute. Le processus s'itère tant que des ordres sont à produire.

Tableau IX

Résultat des politiques de gestion

	% de A → B	% de A → C	% de B → C	Écart de production	Délai de production
EDD	10,4	46,9	32,2	0	150
Marge	25,3	29,3	54,7	0	138
SPT	11,2	50,6	34,1	0	156
LPT	23,7	31,7	49,8	0	134
NOF	26,6	42,1	65,4	0	155

Tableau X

Classement des politiques de gestion

	Politique	Points
1ère	EDD	20
1ère	LPT	20
3ème	Marge	18
4ème	SPT	15
5ème	NOF	12

6.5 Analyse et discussion

Le scénario démonstratif a démontré la viabilité de l'architecture générique proposée. Les méthodes utilisées pour rendre ce scénario fonctionnel nous ont portés à nous interroger sur certains aspects du prototype ou de l'architecture générique. Nous présentons ici l'état de nos réflexions.

6.5.1 Sur le prototype

Pour la construction du prototype, nous avons dû nous adapter à l'absence de MES. Nous avons modifié le modèle de simulation présenté au chapitre 6. Le modèle modifié a pris la place du processus manufacturier. Le serveur OPC prend alors la place du MES. Il est en liaison avec le processus manufacturier via un client OPC. Les statuts de l'atelier ainsi simulé sont ainsi connus en temps réel.

L'absence du MES a compliqué la mise en œuvre du prototype. En effet, les MES et les ERP s'intègrent bien aujourd'hui. Bénéficier d'un MES aurait grandement facilité les opérations de confirmation des ordres et de gestion des événements du processus manufacturier.

Le module décisionnel est relativement simple dans son implémentation. Les difficultés que nous avons rencontrées sont dûes à la synchronisation des tâches. Un événement va lancer le démarrage du module et à partir de cet instant le fonctionnement est séquentiel. Ce fonctionnement séquentiel ne poserait pas de problème si tout se déroulait au sein d'une seule et même application. Or, le module décisionnel met en œuvre trois sous-processus dans des applications distinctes : une application de manipulation des données et le simulateur qui est une application commerciale. Le simulateur ne doit pas démarrer avant que les fichiers d'initialisation soient construits. De même, le dernier sous-processus de compilation des résultats de la simulation ne doit pas démarrer avant que toutes les réplications soient achevées.

Pour arriver à surmonter cet écueil, il a été nécessaire d'utiliser un chef d'orchestre qui est à l'écoute de ce qui se passe dans son module. Quand le processus décisionnel n'est pas en fonctionnement, cette application est à l'écoute du MES pour détecter l'occurrence de l'événement déclencheur du processus de réordonnancement. C'est encore lui qui donne le coup d'envoi de l'initialisation du modèle de simulation. Il prend en charge le lancement des réplications et surveille leur bonne exécution. Quand tout est fini, il lance le dépouillement des résultats. Et finalement, il envoie la nouvelle séquence au MES.

6.5.2 Sur l'architecture générique

6.5.2.1 Instabilité des décisions prises

Le concept de processus adaptatif permet au processus manufacturier de s'adapter plus rapidement à l'apparition de certains événements dans l'atelier ou dans l'entreprise. Suivant les cas, il faut être prudent sur la fréquence d'itération du processus de réaction. Prenons un exemple. À un instant t un ensemble de décisions est mis en application dans l'atelier en réaction à l'occurrence d'un événement perturbateur. Peu de temps après, le

processus adaptatif est à nouveau déclenché et conduit à remettre en cause les décisions fraîchement prises. Si cela se produit trop souvent ou à une fréquence trop élevée, les décisions émanant de ce système n'auront plus aucun sens pour le personnel de l'atelier. En effet, à quoi bon suivre ces directives, nous savons qu'elles vont être remises en cause sous peu ? De plus, si les nouvelles décisions remettent profondément en cause les anciennes, il faudra peut-être envisager de gérer la transition vers les nouvelles décisions plutôt que de faire face à un changement brusque et coûteux.

Ce type d'instabilité n'est pas une nouveauté. Il est inhérent à tous les outils d'optimisation qui permettent de calculer, d'évaluer un nouvel ensemble de décisions dans un laps de temps relativement court. Les APS sont confrontés au même problème.

6.5.2.2 Intérêt de l'approche orientée services

Comme nous l'avons évoqué plus tôt, la difficulté de synchronisation des activités est présente dans toute l'architecture et pas uniquement dans le module décisionnel. Si, à notre échelle, nous y avons été confronté, je vous laisse imaginer la taille du problème dans un système comme SAP R/3.

L'approche orientée service nous semble être très intéressante dans notre cas et simplifierait grandement la synchronisation des tâches. Dans une telle architecture, chaque application implémente un certain nombre de services. Pour chaque service, elle dispose d'un contrat spécifiant les entrants nécessaires au bon fonctionnement du service et les extrants que le mandant peut en attendre. Dans cette approche (Figure 55), un gestionnaire de service prend en charge le rôle de chef d'orchestre. Il a connaissance de tous les services proposés dans l'architecture. Quand une application a besoin d'appeler un service externe (le mandant), elle envoie sa demande au gestionnaire qui se charge de la faire traiter par la bonne application et de lui retourner le résultat. Pour le mandant tout est transparent. Peu importe qui implémente réellement le service, du moment qu'il

respecte le contrat. L'appel des services peut se faire de manière synchrone (le mandant attend la réponse avant de poursuivre) ou de manière asynchrone (le mandant appelle le service et continue de faire ce qu'il a à faire).

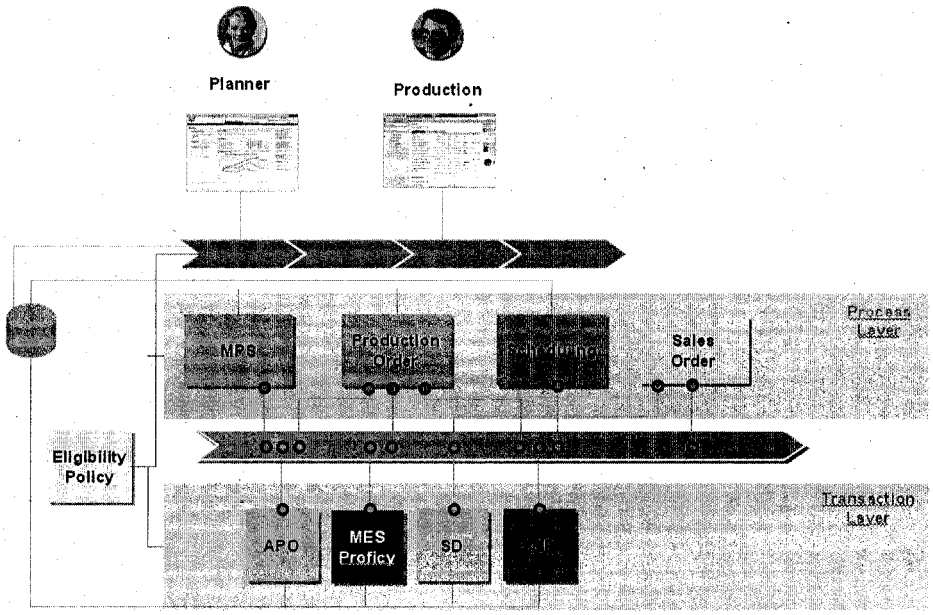


Figure 55 Architecture générique orientée service

Une telle approche, bien que très intéressante, pose plusieurs problèmes. Le premier n'est pas des moindres. Comme nous l'avons évoqué dans le Chapitre 1, cette approche nécessite de réécrire les applications actuelles sous forme de service. La tâche est immense, mais certains grands acteurs tels que SAP s'y sont déjà attelés. Le second problème est plus terre-à-terre. En effet, pour que le mandant soit indépendant de l'application implémentant le service, il faut que toutes les applications implémentant ce service se mettent d'accord sur un seul et unique contrat. C'est dans l'intérêt du mandant, mais très peu dans celui du fournisseur de service. En effet, le mandant pourrait alors sans aucune contrainte changer de fournisseur de service. De plus, nous vous laissons imaginer les luttes d'influence qui auraient lieu au moment de définir le

contrat du service. Ce serait digne de la guerre qui fit rage entre Microsoft, Netscape et les autres fournisseurs de fureteur dans les années 1990 au moment de définir les standards du HTML. Il est à craindre que le même sort soit réservé aux petits fournisseurs de services.

6.6 Conclusion

Pour la construction du prototype, nous avons dû nous adapter à l'absence de MES. Grâce à l'utilisation du modèle de simulation modifié et à un serveur OPC, cette absence est quasiment compensée. Dans les tests que nous avons menés, l'ERP comporte une quinzaine d'ordres à produire et le système manufacturier n'est pas initialement vide. Une itération du processus décisionnel est détaillée. Bien que ce processus décisionnel est simple dans sa mise en œuvre, il est suffisant pour démontrer la viabilité de notre architecture générique d'exécution.

L'analyse menée met en lumière les simplifications que l'utilisation d'un MES aurait entraîner au niveau de la confirmation des ordres et de l'émission des alertes à l'occurrence des événements déclencheurs du processus de réodonnancement. Elle met aussi le doigt sur une difficulté majeure. Il sera plus que courant que le module décisionnel soit constitué d'au moins deux applications : une application implémentant l'heuristique d'optimisation et le simulateur évaluant les performances des décisions prises. Il devient alors critique de correctement synchroniser l'usage de ces applications pour que le processus de décision opère comme prévu.

Nous avons ensuite abordé les risques de l'instabilité qui peut être engendrée par une réoptimisation trop fréquente. Le processus adaptatif ne doit pas se faire au détriment de sa robustesse. Enfin, nous avons évoqué l'intérêt de l'approche orientée service, les simplifications qu'elle entraînerait dans notre architecture générique, mais aussi les difficultés de sa mise en œuvre.

CONCLUSION

L'architecture systémique des entreprises est en perpétuelle adaptation et reflète les profonds changements qui s'opèrent dans la gestion de ces entreprises. Aujourd'hui, le besoin de réviser les priorités de fabrication pour compenser des conditions changeantes sur la demande dans l'optique de respecter les objectifs financiers a poussé un certain nombre d'entreprises à passer d'une mentalité de « réaction aux changements sur les prévisions » à un raisonnement à base de « processus adaptatifs ». Dans ce contexte, les entreprises ont besoin de détecter les exceptions de façon proactive, de résoudre ces difficultés en collaborant avec leurs fournisseurs et leurs sous-traitants, et d'en tirer les leçons qui permettront l'amélioration des procédés en temps réel. Les organisations manufacturières adaptatives ont aussi besoin d'une plateforme d'exécution qui connecte les procédés de fabrication au reste de l'entreprise et de la chaîne logistique et qui fournisse une aide à la décision au personnel de l'atelier, leur permettant de livrer tout en respectant les objectifs de performance.

Les ERP sont conçus pour intégrer tous les processus internes d'affaire à l'entreprise. Malgré des coûts de maintenance et d'intégration élevés, le taux d'adoption des ERP continue d'augmenter. À tel point qu'aujourd'hui 80% des entreprises figurant dans le classement de Fortune 500 utilisent un ERP pour gérer leurs opérations et qu'un nombre croissant d'organismes publics et de petites et moyennes entreprises sont en train d'adopter la même stratégie. Néanmoins, les ERP présentent toujours un certain nombre de limitations dans le domaine de la gestion de production que nous avons évoqué. En plus de cela, les ERP ont besoin de systèmes externes additionnels pour collecter et contrôler les données en temps réel. Les ERP sont conçus pour collecter et retracer des événements, mais ne les exploitent dans le cadre d'un processus d'aide à la prise de décisions.

Le concept de MES est né de la forte demande des industriels pour répondre aux exigences des marchés dans une optique de réactivité, de qualité, de respect des normes, de diminution des coûts et des délais. Historiquement bien implanté dans les industries de *process* (les industries pharmaceutiques, agro-alimentaires et des semi-conducteurs, en particulier) où il est une très bonne réponse aux exigences de traçabilité et d'archivage imposées par les réglementations, ce système s'ouvre de plus en plus à tous les secteurs d'activités, notamment aux entreprises manufacturières.

La valeur ajoutée des entreprises manufacturières se créant dans l'atelier, le MES occupe une position particulière dans l'architecture des systèmes d'information. Il fait office de pont entre les différents systèmes d'information, notamment entre le contrôle de l'atelier et les ERP. Mais tous les modèles d'intégration existants ne sont faits que pour répondre à des problématiques ou des produits commerciaux bien spécifiques. Tous ces systèmes incorporent un nombre plus ou moins important de méthodes d'optimisation. Mais leur rigidité les rend inefficaces à reproduire l'incertitude intrinsèque aux ateliers. La capacité du MES à travailler avec les données en temps réel en fait une source inestimable d'information nécessaire à l'optimisation.

La relative facilité de modélisation de la complexité inhérente des ateliers et des organisations actuelles par les outils de simulation en font de redoutables outils d'analyse des performances. C'est donc naturellement que les heuristiques et la simulation ont été intégrées pour permettre d'optimiser en temps réel les performances de ces systèmes complexes.

Jusqu'ici, la littérature scientifique a proposé différents modèles d'intégration entre les systèmes ERP et MES. Alors que ces modèles sont faits pour résoudre certains problèmes d'interopérabilité, ils n'ajoutent pas d'outils d'aide à la décision à l'un ou l'autre des systèmes. L'architecture que nous proposons, en plus d'intégrer classiquement ces deux systèmes, leur adjoint un module d'aide à la décision en temps

réel. Cette architecture est générique et ouverte. Elle est indépendante des applications commerciales qui seront réellement implantées. La définition des interfaces entre les systèmes s'appuie sur des standards tels que la norme S95 ou bien encore le standard OPC.

L'architecture proposée a fait l'objet d'une implantation démonstrative basée sur un scénario de production de l'industrie de l'aluminium. Dans l'étude préliminaire, le processus manufacturier actuel est étudié et les événements perturbateurs auxquels l'entreprise souhaite s'adapter sont identifiés. Par la suite, un processus adaptatif est proposé pour répondre aux occurrences de ces événements. Le processus ayant été reconçu, il est normal que de nouveaux indicateurs de performance soient mis en place. Quand le nouveau processus adaptatif est connu, il est possible de proposer l'architecture système qui le supportera et d'affecter les tâches au système le plus apte à la mener à bien. Dans le cadre de ce scénario, les solutions logicielles retenues sont :

- SAP R/3 version 4.6c comme ERP;
- Proficy comme MES;
- SDBuilder comme simulateur.

À partir de cet instant, tous les composants développés sont dépendant des solutions logicielles retenues. Pour concrétiser cet exemple démonstratif, il a été nécessaire de mettre en œuvre des éléments de connectivité déjà existants tels que le connecteur Java de SAP. Nous avons aussi été obligés de développer des fonctions dans SAP répondant aux besoins spécifiques du scénario. Nous avons aussi développé un client OPC qui a été intégré au simulateur SDBuilder. Le simulateur est alors doté de moyen de communication qui nous permet de l'utiliser directement dans un environnement manufacturier. Nous avons aussi développé un module décisionnel évaluant les performances de 5 règles de gestion. La règle la plus performante sera appliquée dans l'atelier.

Une fois tous les éléments implémentés, ils ont été assemblés pour construire le prototype. L'absence du MES a pu être palliée grâce aux capacités de communication de SDBuilder. Notre but n'était pas de développer un modèle d'optimisation performant, mais bien de démontrer la viabilité du concept générique. En ce sens, le scénario démonstratif nous permet de conclure par la positive. Et même si les fonctions développées sont spécifiques au scénario, les méthodes utilisées restent génériques dans la mesure où les systèmes retenus sont les même que les nôtres.

L'analyse menée à la suite des expérimentations met en lumière les simplifications que l'utilisation d'un MES aurait entraîner au niveau de la confirmation des ordres et de l'émission des alertes à l'occurrence des événements déclencheur du processus de réodonnancement. Elle met aussi le doigt sur une difficulté majeure. Il sera plus que courant que le module décisionnel soit constitué d'au moins deux applications : une application implémentant l'heuristique d'optimisation et le simulateur évaluant les performances des décisions prises. Il devient alors critique de correctement synchroniser l'usage de ces applications pour que le processus de décision opère comme prévu.

Nous avons ensuite abordé les risques de l'instabilité qui peut être engendrée par une réoptimisation trop fréquente. Le processus adaptatif ne doit pas se faire au détriment de sa robustesse. Ce type d'instabilité n'est pas une nouveauté. Il est inhérent à tous les outils d'optimisation qui permettent de calculer, d'évaluer un nouvel ensemble de décisions dans un laps de temps relativement court. Les APS sont confrontés au même problème.

Enfin, nous avons évoqué l'intérêt de l'approche orientée service, les simplifications qu'elle entraînerait dans notre architecture générique et dans la synchronisation des différentes tâches. Mais il y a le revers de la médaille qui ne doit pas être occulté par tous ces attraits. Cette approche nécessiterait un gros effort de reconception des

applications existantes et suppose aussi que les acteurs d'un même marché se mettent d'accord sur les contrats des différents services.

RECOMMANDATIONS

Nous ferons deux recommandations. L'une porte sur un développement de l'architecture générique proposée, l'autre sur son utilisation.

Nous pensons qu'il serait intéressant d'approfondir l'approche orientée service. SAP est en train de se lancer dans cette direction et nous pensons qu'elle simplifierait grandement l'intégration de systèmes hétérogènes. De plus, si la pression se fait de plus en plus forte sur les vendeurs de logiciels, ils seront fortement tentés d'investir ce marché.

Le scénario démonstratif utilise un module décisionnel simple et pas nécessairement efficace. L'architecture générique lève les barrières entre les systèmes opérationnels, d'affaires et le noyau d'aide à la prise de décision en temps réel. Il faut que les méthodes d'aide à la décision en fassent autant et qu'elles brisent les paradigmes qui les enchaînent pour investir le champ de l'optimisation en temps réel.

ANNEXE 1

FICHES PRODUIT DE L'ÉTUDE DE MARCHÉ DES MES

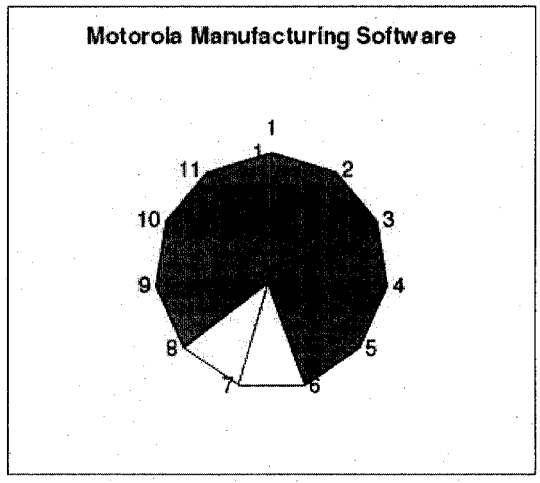
FICHES PRODUIT DE L'ÉTUDE DE MARCHÉ DES MES

Nom du produit : **Motorola Manufacturing Software**

Lien principal : http://www.manufacturingpulse.com/products/main.htm

Fonctionnalité MESA

1 Collecte et acquisition des données
2 Analyse des performances
3 Traçabilité et généalogie
4 Ordonnancement
5 Gestion des documents
6 Cheminement des des lots
7 Gestion du personnel
8 Gestion de la qualité
9 Gestion du procédé
10 Gestion de la maintenance
11 Gestion des ressources



Autres fonctions mises en avant : -

Apports :

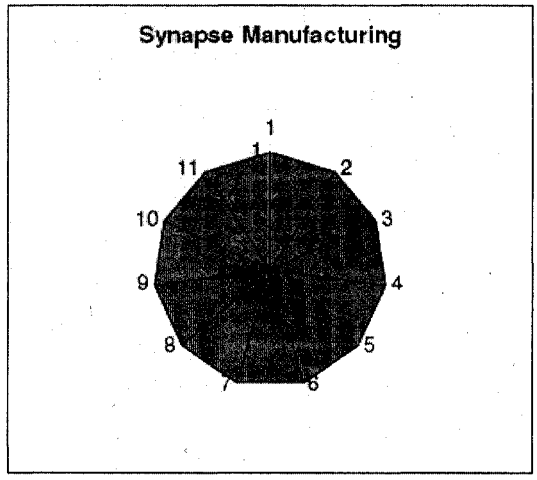
Optimisation de la productivité
Amélioration de la qualité (contrôle statistique du procédé)
Réduction du rebut
Promouvoir les contrôles ISO

Nom du produit : **Synapse Manufacturing**

Lien principal : http://www.ibss.net/manf.htm

Fonctionnalité MESA

1 Collecte et acquisition des données
2 Analyse des performances
3 Traçabilité et généalogie
4 Ordonnancement
5 Gestion des documents
6 Cheminement des des lots
7 Gestion du personnel
8 Gestion de la qualité
9 Gestion du procédé
10 Gestion de la maintenance
11 Gestion des ressources



Autres fonctions mises en avant : Management des stocks
Management de l'exécution

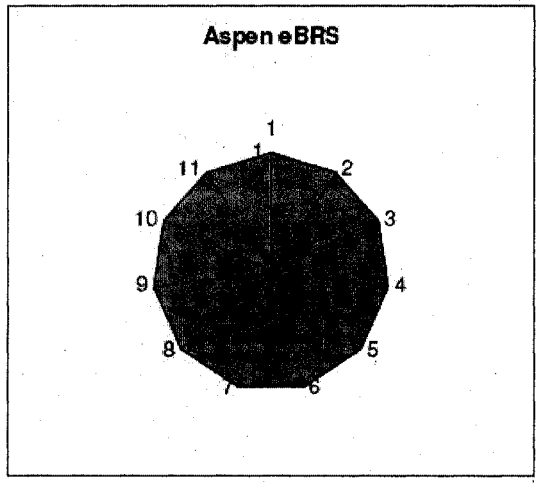
Apports : Non précisés

Nom du produit : **Aspen eBRS**

Lien principal : http://www.aspentech.com

Fonctionnalité MESA

1 Collecte et acquisition des données
2 Analyse des performances
3 Traçabilité et généalogie
4 Ordonnancement
5 Gestion des documents
6 Cheminement des des lots
7 Gestion du personnel
8 Gestion de la qualité
9 Gestion du procédé
10 Gestion de la maintenance
11 Gestion des ressources



Autres fonctions mises en avant : -

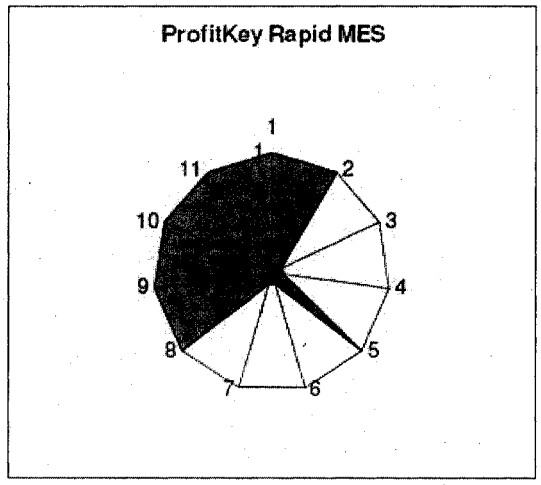
Apports : Réduction des coûts par l'utilisation de best practices
Optimisation la production
Réduction des délais

Nom du produit : **ProfitKey Rapid MES**

Lien principal : http://www.profitkey.com/MODULES.asp#mes

Fonctionnalité MESA

1 Collecte et acquisition des données
2 Analyse des performances
3 Traçabilité et généalogie
4 Ordonnancement
5 Gestion des documents
6 Cheminement des des lots
7 Gestion du personnel
8 Gestion de la qualité
9 Gestion du procédé
10 Gestion de la maintenance
11 Gestion des ressources



Autres fonctions mises en avant : Management des capitaux

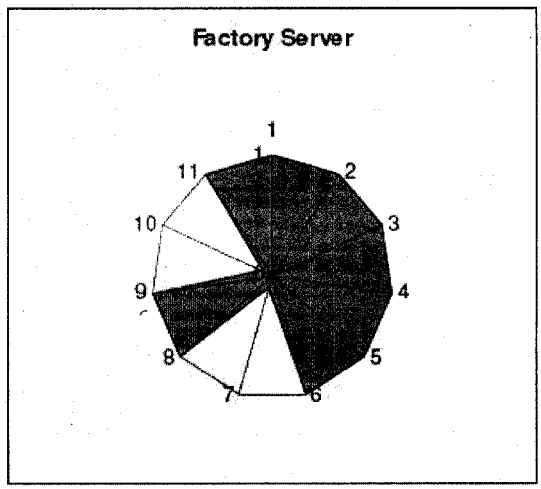
Apports : Compétitivité - expension du marché
Augmentation des revenus
Augmentation des compétences sans augmenter la taille des équipes

Nom du produit : **Factory Server**

Lien principal : http://www.massgroup.com/Products/MES.asp

Fonctionnalité MESA

1 Collecte et acquisition des données
2 Analyse des performances
3 Traçabilité et généalogie
4 Ordonnancement
5 Gestion des documents
6 Cheminement des des lots
7 Gestion du personnel
8 Gestion de la qualité
9 Gestion du procédé
10 Gestion de la maintenance
11 Gestion des ressources



Autres fonctions mises en avant : -

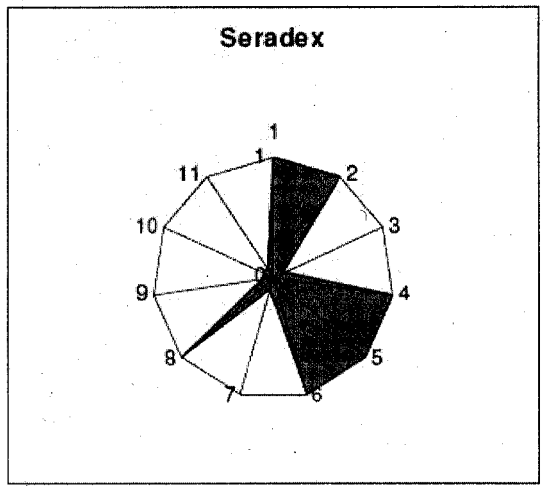
Apports : Réduction des temps de cycles et du Time to Market
Réduction des coûts
Valable pour différentes industries (Semiconductor Fabrication, LCD/Flat Panel Manufacturing, Electronics Packaging & Assembly, Medical, Device Manufacturing, Pharmaceuticals, Aerospace, Automotive, Food & Beverage)

Nom du produit : **Seradex**

Lien principal : http://www.seradex.com/MES_ERP.shtml

Fonctionnalité MESA

1 Collecte et acquisition des données
2 Analyse des performances
3 Traçabilité et généalogie
4 Ordonnancement
5 Gestion des documents
6 Cheminement des des lots
7 Gestion du personnel
8 Gestion de la qualité
9 Gestion du procédé
10 Gestion de la maintenance
11 Gestion des ressources



Autres fonctions mises en avant : modules d'un ERP

Apports :

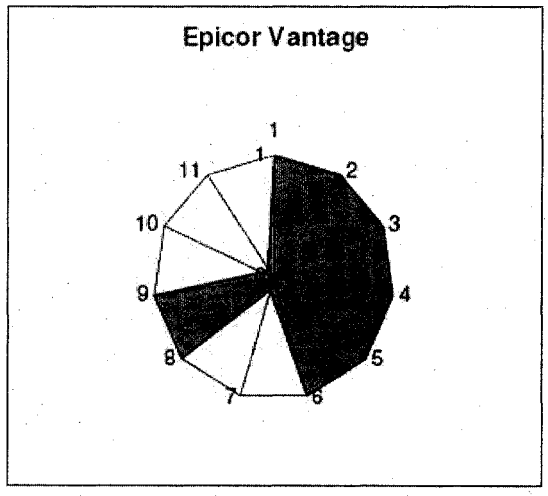
Réduction du coût de la collecte d'information
Augmentation de l'exactitude des informations
Meilleur prise de décision pour la production
Engagement plus précis auprès du client
Aide à un meilleur respect des temps de production
Connaissance en temps réel du statut des ordres
Recherche des coûts et des causes du rebus et des retouches
Préviens l'excès d'en-cours

Nom du produit : **Epicor Vantage**

Lien principal : http://msg.epicor.com/msg/Vantage/Modules/

Fonctionnalité MESA

1 Collecte et acquisition des données
2 Analyse des performances
3 Traçabilité et généalogie
4 Ordonnancement
5 Gestion des documents
6 Cheminement des des lots
7 Gestion du personnel
8 Gestion de la qualité
9 Gestion du procédé
10 Gestion de la maintenance
11 Gestion des ressources



Autres fonctions mises en avant : modules d'un ERP

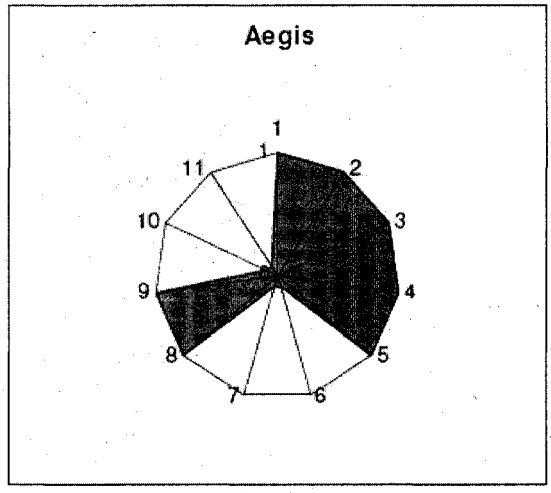
Apports : Non précisés

Nom du produit : **Aegis**

Lien principal : http://www.aiscorp.com/products_mes_overview.htm

Fonctionnalité MESA

1 Collecte et acquisition des données
2 Analyse des performances
3 Traçabilité et généalogie
4 Ordonnancement
5 Gestion des documents
6 Cheminement des des lots
7 Gestion du personnel
8 Gestion de la qualité
9 Gestion du procédé
10 Gestion de la maintenance
11 Gestion des ressources



Autres fonctions mises en avant : Canal de communication sur chaque poste
Communication de certaines données aux acheteurs/fournisseurs

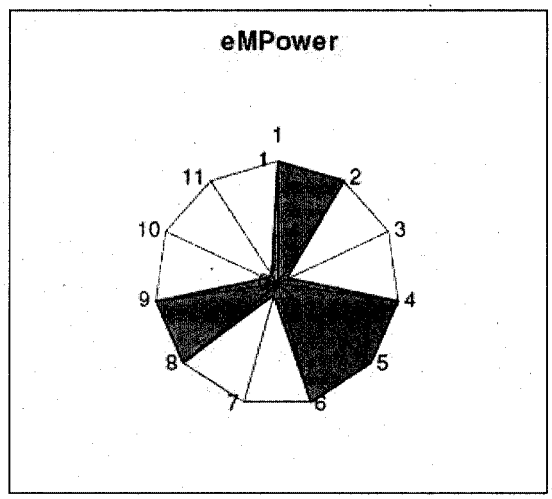
Apports : 100% accessible par le web
Robuste
Pour un large panel d'entreprise
Collecte des données : homme-machine, machine-machine

Nom du produit : **eMPower**

Lien principal : http://www.tecnomatix.com/showpage.asp?page=377

Fonctionnalité MESA

1 Collecte et acquisition des données
2 Analyse des performances
3 Traçabilité et généalogie
4 Ordonnancement
5 Gestion des documents
6 Cheminement des des lots
7 Gestion du personnel
8 Gestion de la qualité
9 Gestion du procédé
10 Gestion de la maintenance
11 Gestion des ressources



Autres fonctions mises en avant : -

Apports :

Pour l'industrie de l'électronique

Réduction de 25% du rebut

Aide à la mise en place de la méthode des 6 sigma

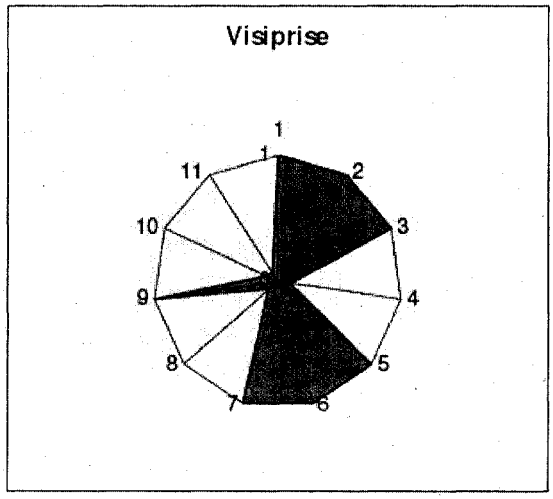
Sans papier (donc réduction des coûts)

Nom du produit : **Visiprise**

Lien principal : http://www.visiprise.com

Fonctionnalité MESA

1 Collecte et acquisition des données
2 Analyse des performances
3 Traçabilité et généalogie
4 Ordonnancement
5 Gestion des documents
6 Cheminement des des lots
7 Gestion du personnel
8 Gestion de la qualité
9 Gestion du procédé
10 Gestion de la maintenance
11 Gestion des ressources



Autres fonctions mises en avant : -

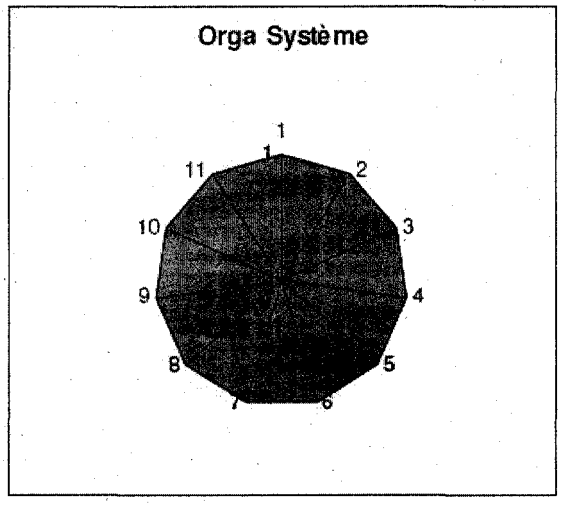
Apports : Pour production discrète

Nom du produit : **Orga Système**

Lien principal : http://www.osys.fr/mes.php?lang=fr

Fonctionnalité MESA

1 Collecte et acquisition des données
2 Analyse des performances
3 Traçabilité et généalogie
4 Ordonnancement
5 Gestion des documents
6 Cheminement des des lots
7 Gestion du personnel
8 Gestion de la qualité
9 Gestion du procédé
10 Gestion de la maintenance
11 Gestion des ressources



Autres fonctions mises en avant :

Messagerie intranet d'atelier temps réel
Driver OPC
Générateur d'application
Solution matérielle

Apports :

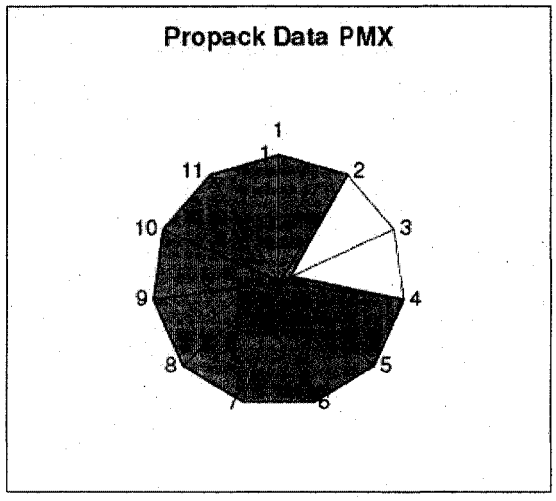
Réactivité
Traçabilité
Productivité

Nom du produit : **Propack Data PMX MES**

Lien principal : http://www.propackdata.com/html/manufacturing_execution_system.html

Fonctionnalité MESA

1 Collecte et acquisition des données
2 Analyse des performances
3 Traçabilité et généalogie
4 Ordonnancement
5 Gestion des documents
6 Cheminement des des lots
7 Gestion du personnel
8 Gestion de la qualité
9 Gestion du procédé
10 Gestion de la maintenance
11 Gestion des ressources



Autres fonctions mises en avant : Technologie certifiée SAP

Apports :

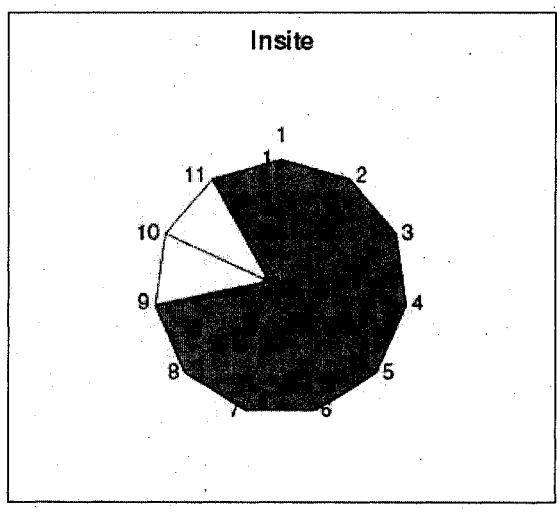
Produit pharmaceutique
Réduction des coûts du personnel et d'activité
Augmentation de la productivité et de l'efficacité
Augmentation de la motivation du personnel et des clients

Nom du produit : **Insite**

Lien principal : http://www.camstar.com/

Fonctionnalité MESA

1 Collecte et acquisition des données
2 Analyse des performances
3 Traçabilité et généalogie
4 Ordonnancement
5 Gestion des documents
6 Cheminement des des lots
7 Gestion du personnel
8 Gestion de la qualité
9 Gestion du procédé
10 Gestion de la maintenance
11 Gestion des ressources



Autres fonctions mises en avant : -

Apports :

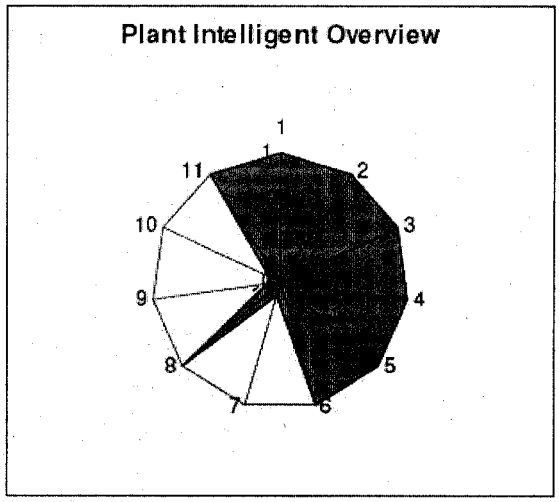
Management efficace des stocks
Traçabilité des ordres et personnalisation
Optimisation des rendements
Accélération des prises de décision
Réduction des cycles de production

Nom du produit : **Plant Intelligent Overview**

Lien principal : http://www.gefanucautomation.com

Fonctionnalité MESA

1 Collecte et acquisition des données
2 Analyse des performances
3 Traçabilité et généalogie
4 Ordonnancement
5 Gestion des documents
6 Cheminement des des lots
7 Gestion du personnel
8 Gestion de la qualité
9 Gestion du procédé
10 Gestion de la maintenance
11 Gestion des ressources



Autres fonctions mises en avant : -

Apports :

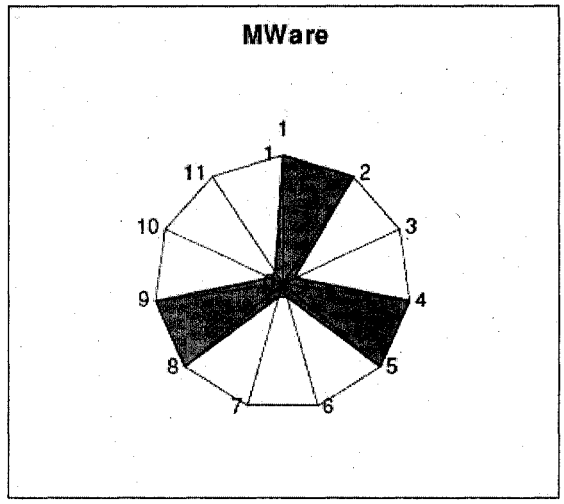
S'intégrer avec les systèmes existants
Partager les informations en temps réel
Réduire le temps de réponse
Augmenter la flexibilité
Visualiser les données importantes

Nom du produit : **MWare**

Lien principal : http://www.mware-mes.com/mware1/solu%20MES.htm

Fonctionnalité MESA

1 Collecte et acquisition des données
2 Analyse des performances
3 Traçabilité et généalogie
4 Ordonnancement
5 Gestion des documents
6 Cheminement des des lots
7 Gestion du personnel
8 Gestion de la qualité
9 Gestion du procédé
10 Gestion de la maintenance
11 Gestion des ressources



Autres fonctions mises en avant : -

Apports : Augmentation des parts de marché
Augmentation de la rentabilité
Amélioration de la compétitivité

ANNEXE 2

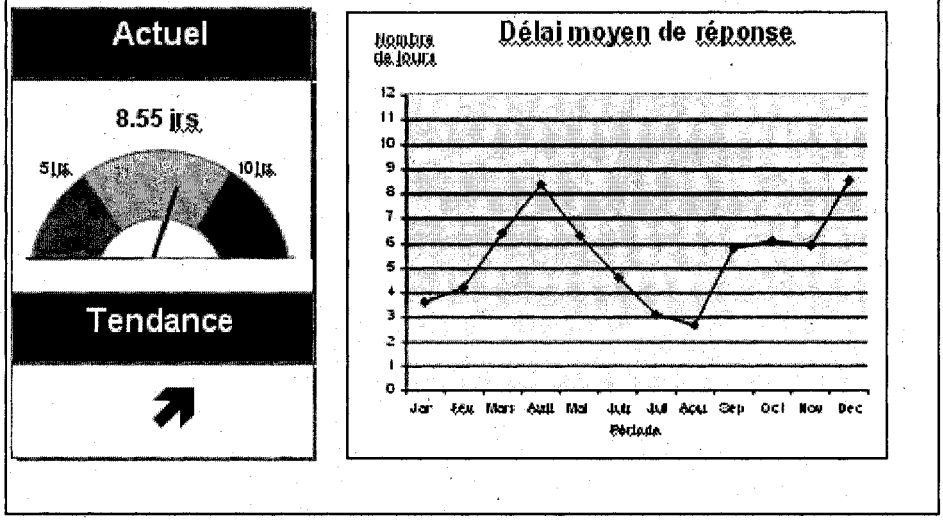
EXEMPLE DE FICHE D'INDICATEUR

EXEMPLE DE FICHE D'INDICATEUR

Fiche d'indicateur
Délai de traitement de remplacement de pièce (ECO)

SECTION 1	DEFINITION
TITRE	Temps de réponse ECO – pièce de remplacement
NUMERO	1.1
FORMULE	For all ECO completed in period: Σ (PREP status end date - PREP status start date) / # ECO
UNITE DE MESURE	Jours
PERIODICITE	Semaine, mois.
CHARACTERISTICS	Customer, material, start time, end time, production order, MRP controller.
FILTRES	PREP Status.
KEYS FIGURES	ECO number, material
SECTION 2	**PROPRIÉTAIRE**
NOM	Engineering support
ORGANISATION	Customer service
SECTION 3	**IMPLANTATION**
SYSTÈME	BW (infocube à développer)
INTERFACES	Nil.
SECTION 4	**OBJECTIFS**
VISION	2 jours
DATE CIBLE	2010
OBJECTIF EN VIGUEUR	4 jours
DATE DE VALIDITE	2007
AVERTISSEMENT	5 jours
NIVEAU D'ALERTE	10 jours
SECTION 5	**BENCHMARKS**
ATTENTE CLIENT	3 jours
MOYENNE DE L'INDUSTRIE	5 jours
BEST OF CLASS	2 jours
SOURCE	Internal report

SECTION 6	REPRÉSENTATION



ANNEXE 3

EXTRAIT DU SAP INTERFACE REPOSITORY

EXTRAIT DU SAP INTERFACE REPOSITORY

Class Method : ProdOrdConfirmation.CreateAtHeaderLevelMultiple

Enter order confirmations

Documentation

Functionality

You can use this method to enter **order confirmations** for production orders.

You can also transfer good movements, that are posted together with a confirmation. If no goods movements have been entered for a confirmation, they are determined using the standard logic for backflushing and automatic goods receipt for confirmations.

Notes

Authorization check

The authorization object **C_AFRU_AWK** is checked together with activity 01 (create), the order type and the work of the production order to be confirmed.

Parameters

- **PostWrongEntries**

  You use the **PostWrongEntries** parameter to control whether in an exception (incorrect data/ locked objects) the confirmations are accepted by the SAP system and placed in the pool of incorrect confirmations.

  Exceptions for confirmations are logged in the **DetailReturn** parameter table.

  The following characteristic values of the **PostWrongEntries** parameter are possible:

  - " ": Confirmations, in which no exception occurred, are posted. All the confirmations are not processed, they are not placed in the pool for incorrect confirmations.
  - "1": Confirmations, in which no exception occurred, are posted. All confirmations with incorrect data are placed in the

pool for incorrect confirmations. On the other hand, confirmations that are not posted due a lock situation are not processed, they are not placed in the pool.

- "2": Confirmations, in which no exception occurred, are posted. All confirmations with exceptions are placed in the pool.

- **Testrun**

  You use the **Testrun** parameter to specify that the transfered data are only checked to see if they are correct. Locking of objects such order and reservation as well as the updating of databases does not take place in this mode.

  You set the value "X" to activate the **Testrun** mode.

- **Return**

  If confirmations could not be processed, due to a serious error, you receive information about the error in the **Return** parameter.

- **AtHdrLevels**

  The confirmation data is transfered to the **AtHdrLevels** table.

- **Goodsmovements**

  If goods movements should be posted that differ from the standard logic, transfer these goods movements in the **Goodsmovements** table.

- **LinkConfGoodsmov**

  The goods movements are linked to a confirmation via the **LinkConfGoodsmov** table. There must be an entry in this table for every entry in the **Goodsmovements** table. The index in the **LinkConfGoodsmov-Index_Confirm** field refers to the lines in the corresponding confirmation in the **AtHdrLevels** table and the index in the **LinkConfGoodsmov-Index_Goodsmov** field refers the corresponding good movement in the **Goodsmovements** table.

  If you want to prevent a goods movement being posted during a confirmation according to the standard logic, make an entry in the **LinkConfGoodsmov** table as well as in the **AtHdrLevels** table. The

index in the **LinkConfGoodsmov-Index_Confirm** field refers to the lines of the corresponding confirmation in the **AtHdrLevels** table. Enter the initial value 0 in the **LinkConfGoodsmov-Index_Goodsmov** field. It is not necessary to make an entry in the **Goodsmovements** table, in this case.

If no entry exists in the **LinkConfGoodsmov** table for a confirmation, goods movements are determined using the standard logic for backflushing and automatic goods receipt for confirmations.

- **DetailReturn**

The **DetailReturn** table informs you for each confirmation to be entered, whether a confirmation could not be entered due to a lock conflict or which error occurred.

- If the confirmation could be enterd successfully, the key of the confirmation is logged in the fields **DetailReturn-Conf_No** and**DetailReturn-Conf_Cnt**.
- In the case of a lock conflict, the **DetailReturn-Flg_Locked** indicator is set.
- If the data is found to be incorrect, a corresponding error message is written in the relevant fileds of the **DetailReturn** table.

Properties	
Direction	inbound

xml		
Schema	Send	Response
Template	Send	Response

Parameter
Import

Athdrlevels	Table	Table of confirmations
Goodsmovements	Table	Table of goods movements
LinkConfGoodsmov	Table	Link table for confirmations/goods movement
PostWrongEntries	CHAR 1	Ind.: Incorrect data in error pool
Testrun	CHAR 1	Ind.: Test data only, without saving
Export		
DetailReturn	Table	Return parameter for confirmation table
Return	Structure	Return parameter

ANNEXE 4

CODE DE LA FONCTION ZSHORTPRODORDER_GETLIST

CODE DE LA FONCTION ZSHORTPRODORDER_GETLIST

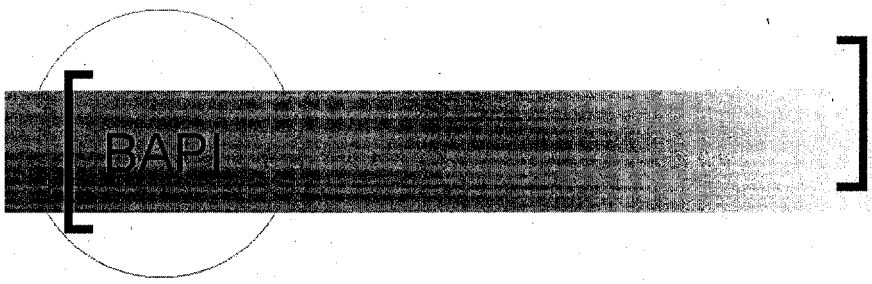
```
FUNCTION ZSHORTPRODORDER_GETLIST.
*"----------------------------------------------------------------------
*"*"Local interface:
*"  IMPORTING
*"     VALUE(PLANT) LIKE  AFPO-DWERK
*"     VALUE(MATERIAL) LIKE  AFPO-MATNR
*"  EXPORTING
*"     VALUE(RETURN) LIKE  BAPIRET2 STRUCTURE  BAPIRET2
*"  TABLES
*"      LIST STRUCTURE  ZSHORTPRODORDER1 OPTIONAL
*"----------------------------------------------------------------------

*"----------------------------------------------------------------------
*" First release
*"   Find all production orders even if they are confirmed
*"----------------------------------------------------------------------

*SELECT AUFNR PLNUM PSMNG AMEIN MATNR DGLTS from AFPO into TABLE list where
MATNR = MATERIAL and DWERK = PLANT.



*"----------------------------------------------------------------------
*" Second release
*"   Find all released and not confirmed production orders
*"----------------------------------------------------------------------


* Local declaration
* Data elements
DATA:
 l_aufnr LIKE afpo-aufnr,
 l_objname LIKE jest-objnr,
 l_flag_released TYPE C VALUE 'N',
 l_flag_confirmed TYPE C VALUE 'N'.
* Tables
DATA:
 l_total_list LIKE ZSHORTPRODORDER1 occurs 0,
 l_one_order LIKE ZSHORTPRODORDER1,
 l_status LIKE ZBAPI_STAT occurs 0,
 l_stat_rel LIKE ZBAPI_STAT,
 l_stat_conf LIKE ZBAPI_STAT.

* Initialize return-type
return-type = 'S'.



* Find all production orders
SELECT AUFNR PLNUM PSMNG AMEIN MATNR DGLTS from AFPO into TABLE l_total_list
where MATNR = MATERIAL and DWERK = PLANT.
IF sy-subrc <> 0.
   return-type = 'W'.
```

```
   CONCATENATE 'No production order for material ' material INTO return-
message.
ENDIF.

* For each production order
LOOP AT l_total_list INTO l_one_order.
* Initialization
   CLEAR l_status.
   l_flag_released = 'N'.
   l_flag_confirmed = 'N'.
* Create object name
   CONCATENATE 'OR' l_one_order-aufnr INTO l_objname.
* Find status for this production order
   SELECT objnr stat inact FROM jest INTO TABLE l_status WHERE objnr =
l_objname AND inact <> 'X'.
* Sort
   SORT l_status BY stat.
* Check if our production order is have a released and a confirmed statu
   READ TABLE l_status WITH KEY stat = 'I0002' BINARY SEARCH INTO l_stat_rel.
   IF sy-subrc = 0.
      l_flag_released = 'Y'.
   ENDIF.
   READ TABLE l_status WITH KEY stat = 'I0009' BINARY SEARCH INTO l_stat_conf.
   IF sy-subrc = 0.
      l_flag_confirmed = 'Y'.
   ENDIF.
* If our production order is released and not confirmed, copy in the output
table
   IF l_flag_released = 'Y'.
      IF l_flag_confirmed = 'N'.
      MOVE l_one_order TO list.
      APPEND list.
   ENDIF.
   ENDIF.


ENDLOOP.


ENDFUNCTION.
```

ANNEXE 5

TUTORIAL – CRÉATION DE BAPI

TUTORIAL – CRÉATION DE BAPI



Business Application
Programming Interface

SAP R/3 4.6C

benoit@s2u.fr

Compilé par B. Saenz de Ugarte, le 20/09/2006

Université du Québec
École de technologie supérieure

A propos de l'exemple

- Créer un BAPI qui, à partir du code du pays, renvoie la liste des vendeurs et leurs coordonnées.

A propos de l'exemple

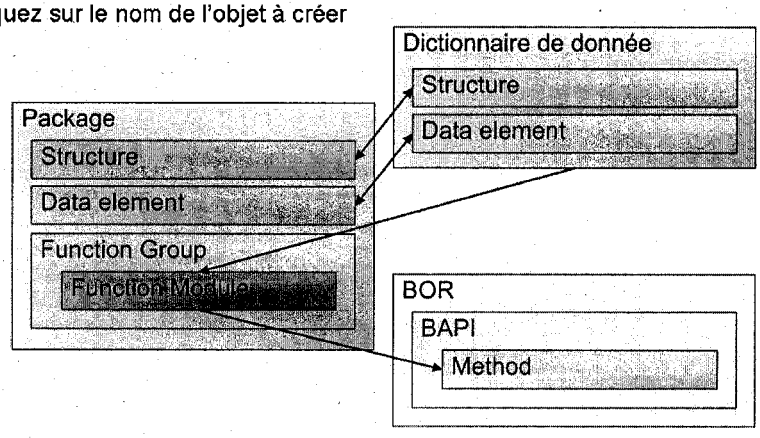
- Table interrogée : LFA1
- Structure crée : ZVEND

Convention

- Tout ce que vous allez créer devra porter un nom commençant par la lettre **Z**.

Hiérarchie

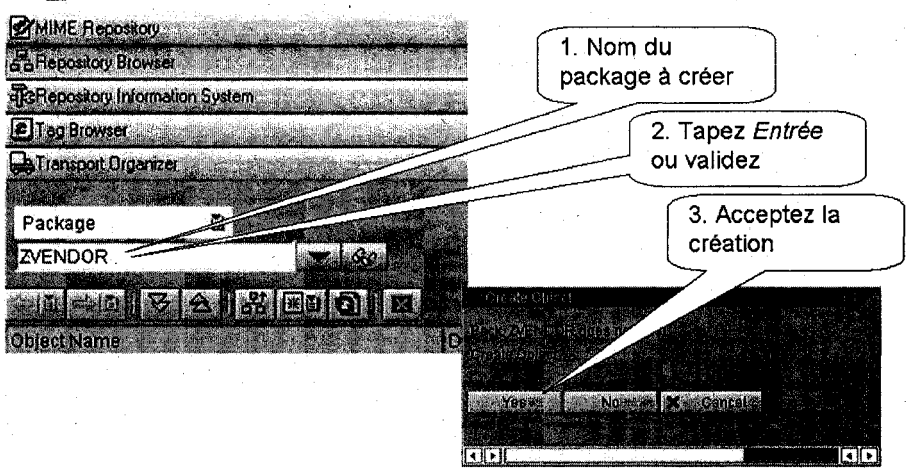
*Cliquez sur le nom de l'objet à créer



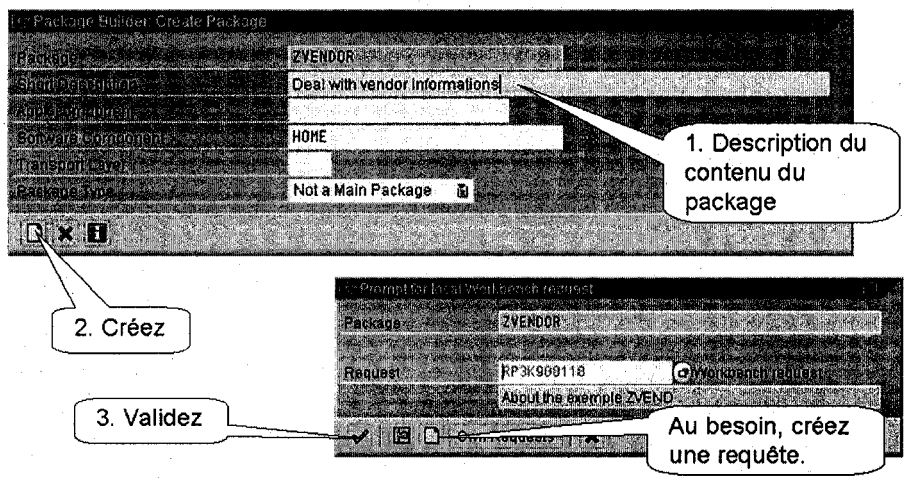
Créer un package (1)

- Un package regroupe les objets qui sont en liaison (remplace les classes de développement).
- Détermine les attributs de transport d'un objet.
- Tcode : **SE80**

Créer un package (2)



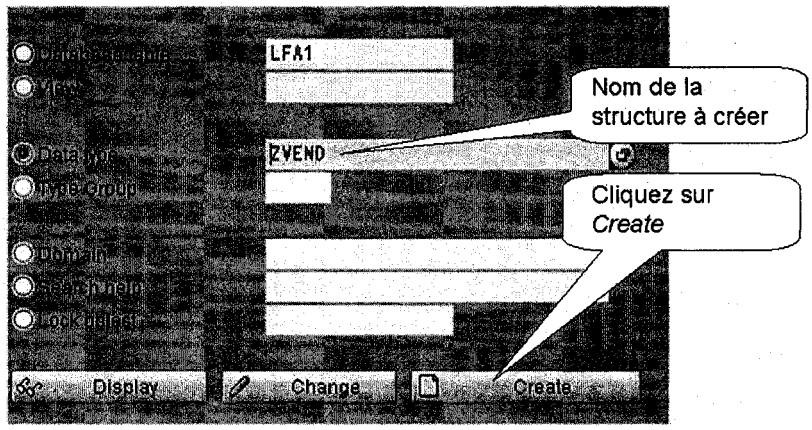
Créer un package (3)



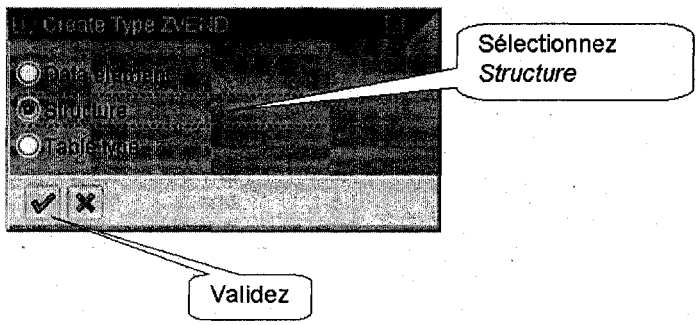
Créer une structure (1)

- Démarche à suivre pour créer une structure (ou un *data element*) servant dans les paramètres d'entrée, d'export ou les tables du BAPI
- Tous les paramètres des BAPI doivent être dans le dictionnaire de données. Des qu'un BAPI à le statu *released* tous les types de données en relation ne sont plus modifiable.
- Tcode : **SE11**

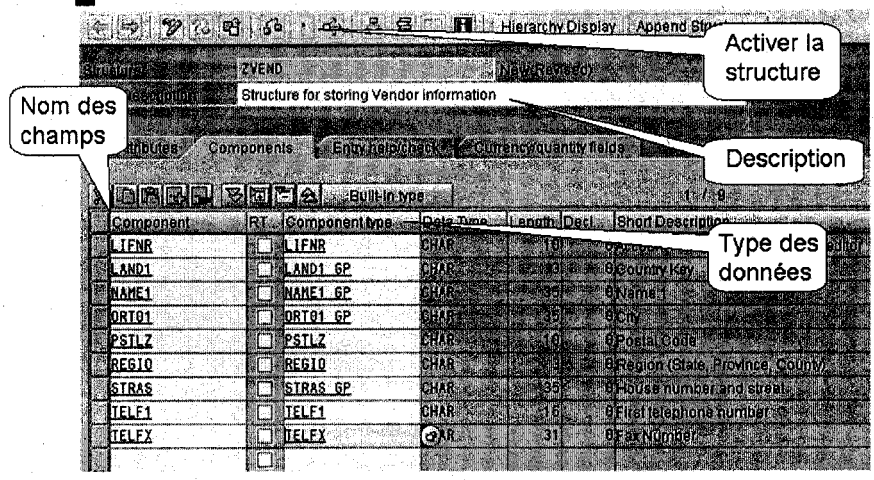
Créer une structure (2)



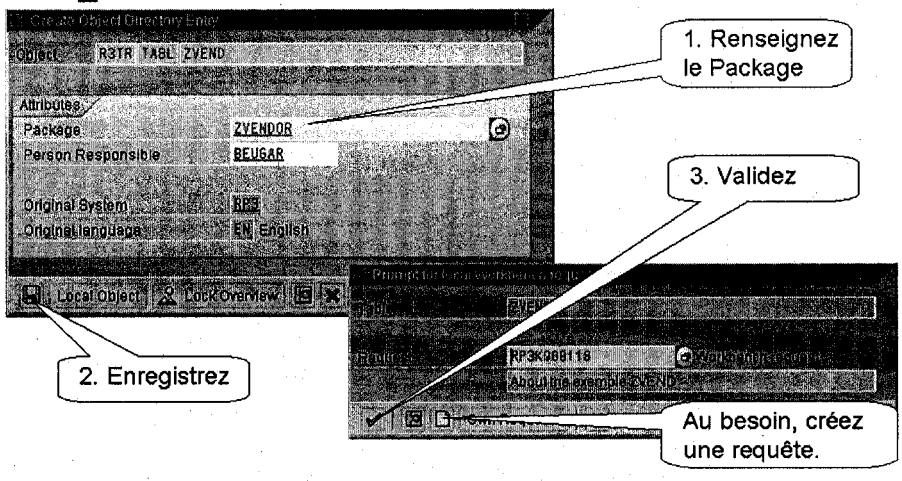
Créer une structure (3)



Créer une structure (4)



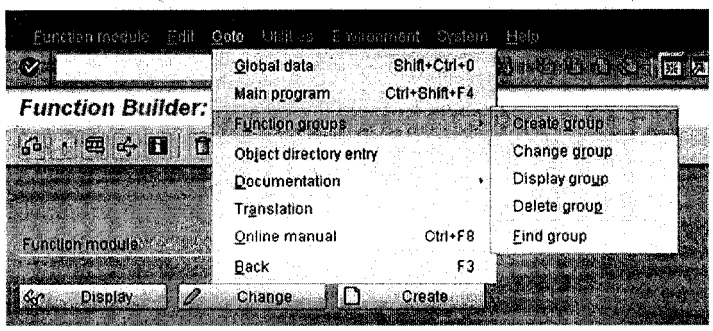
Créer une structure (5)



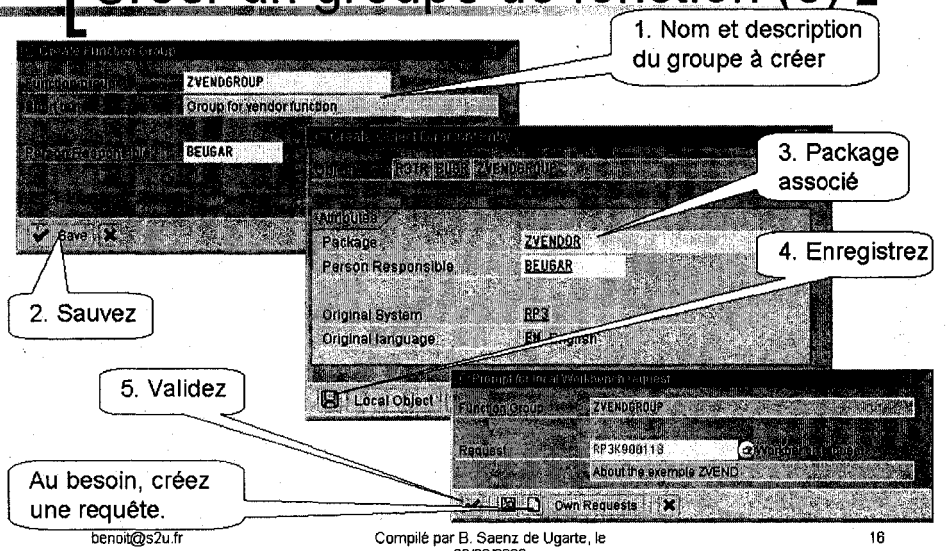
Créer un groupe de fonction (1)

- Chaque BAPI doit avoir son propre groupe de fonction
- Tcode : **SE37**

Créer un groupe de fonction (2)

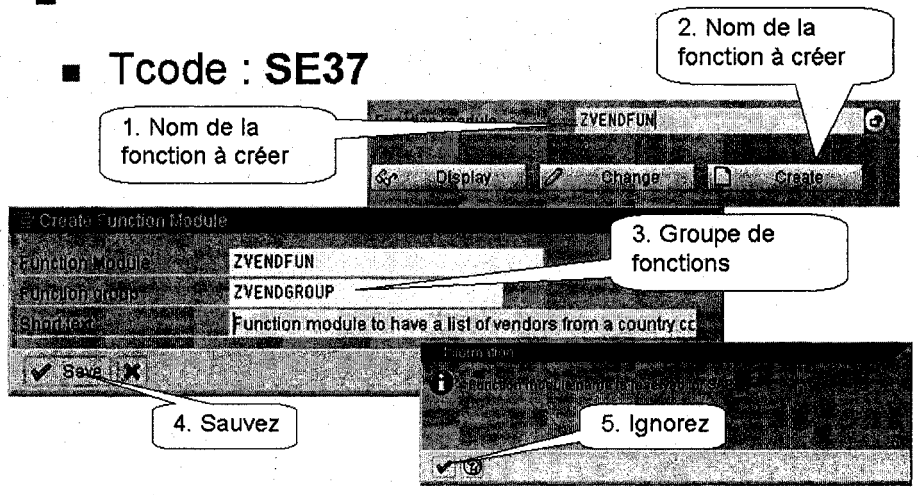


Créer un groupe de fonction (3)

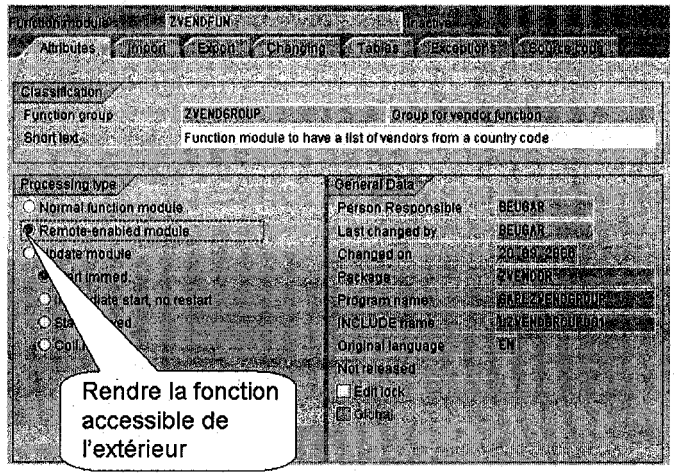


Créer une fonction (1)

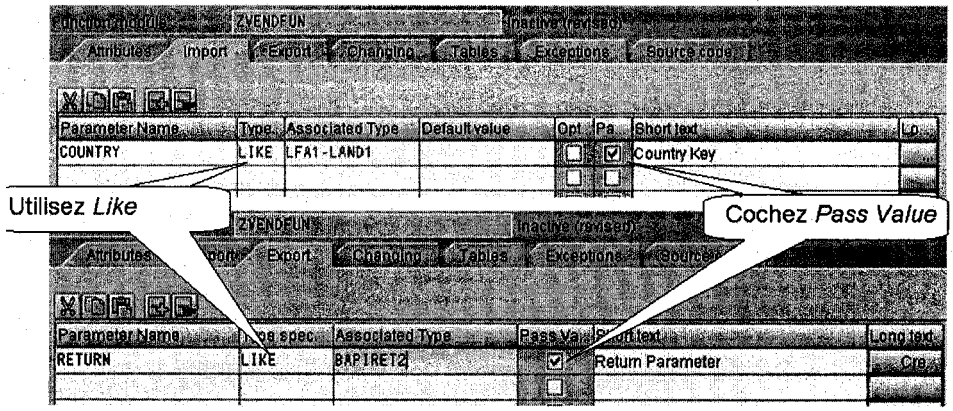
- Tcode : **SE37**



Créer une fonction (2)



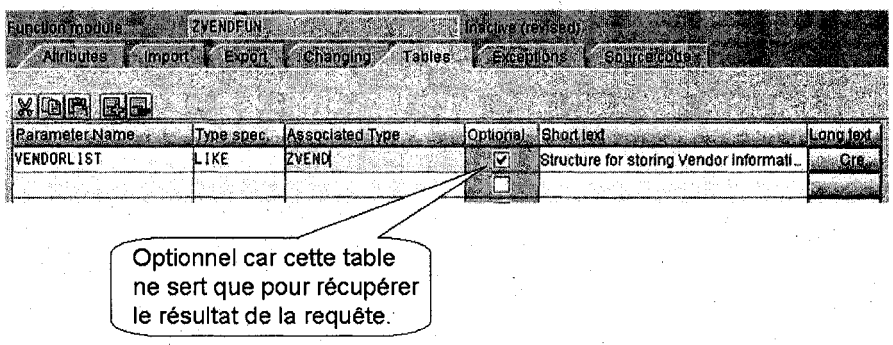
Créer une fonction (3)



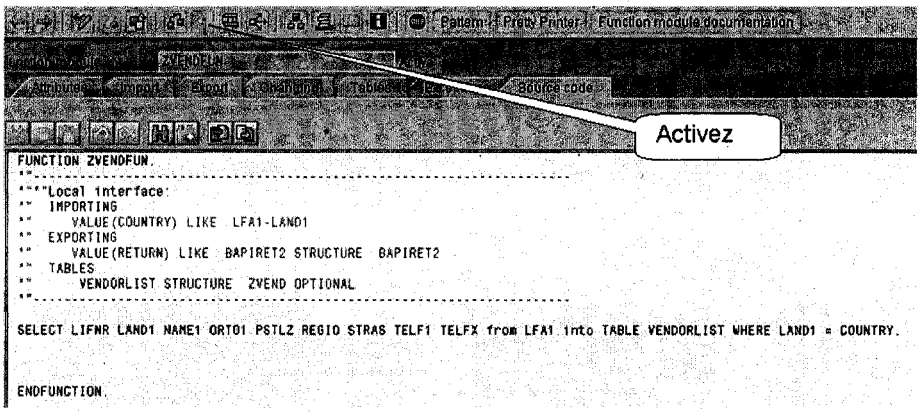
Créer une fonction (4)

- Pour le code de retour, les structures généralement utilisez sont :
  - BAPIRETURN
  - BAPIRETURN1
  - BAPIRET1
  - BAPIRET2

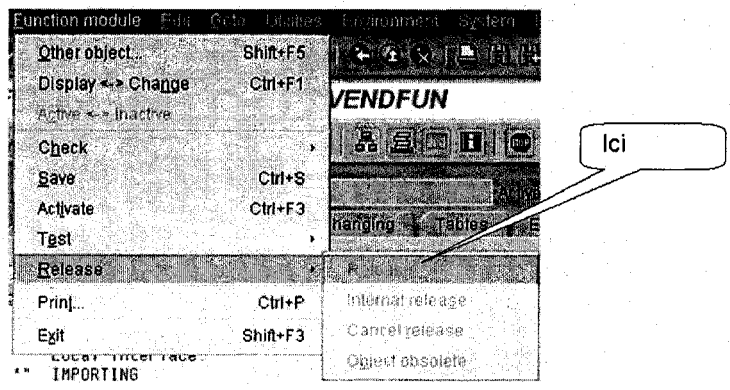
Créer une fonction (5)



Créer une fonction (6)



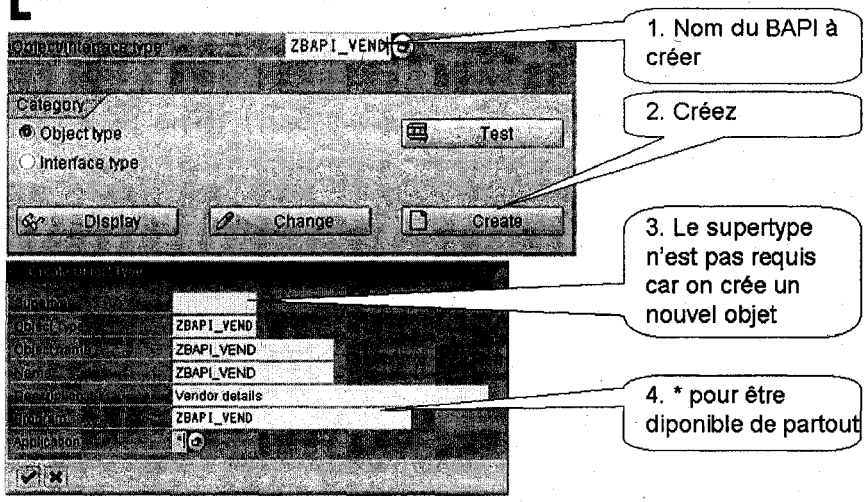
Créer une fonction (7)



Créer un BAPI (1)

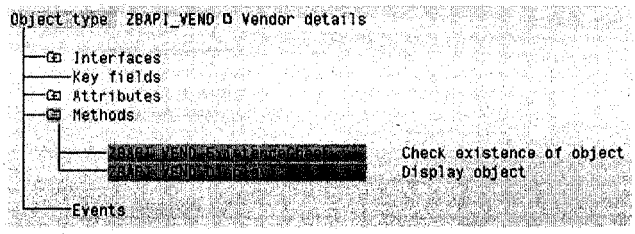
- On utilise ici le BAPI wizard qui va générer du code supplémentaire pour que la fonction soit une méthode valide du BOR
- Chaque fonction va correspondre a une méthode du BOR
- Tcode : **SWO1**

Créer un BAPI (2)



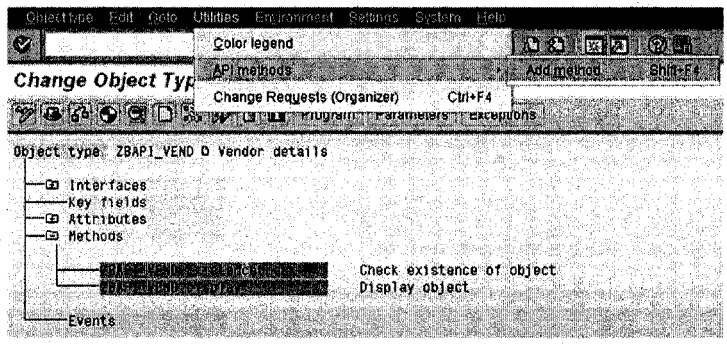
Créer un BAPI (3)

- Interfaces, attributs et méthodes créés par défaut

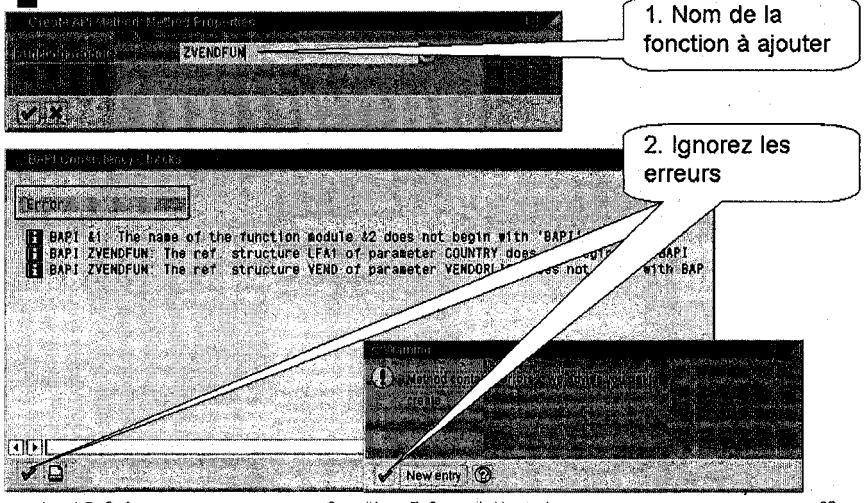


Créer un BAPI (4)

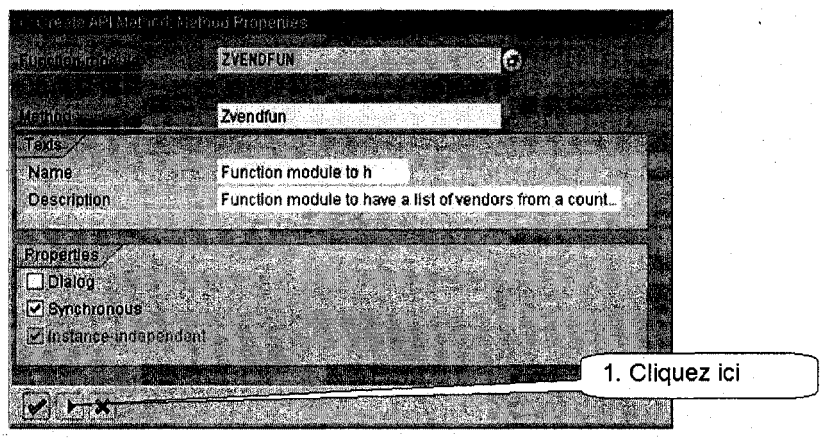
- Ajouter une méthode au BAPI



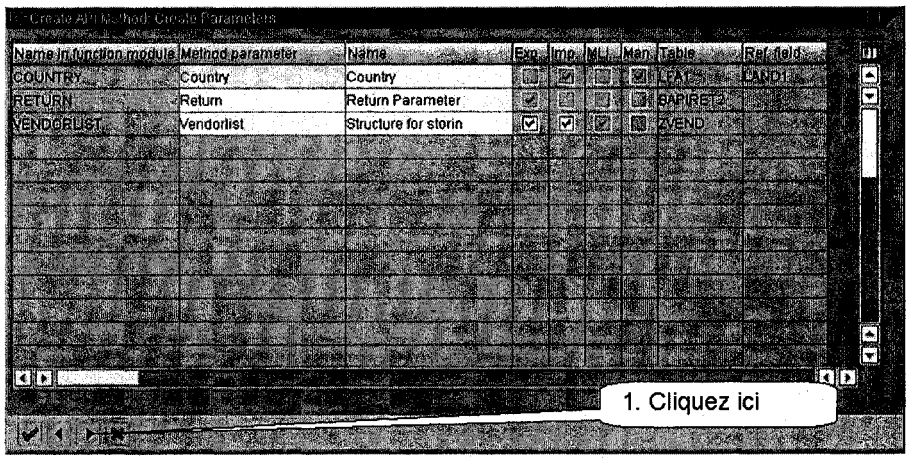
Créer un BAPI (5)



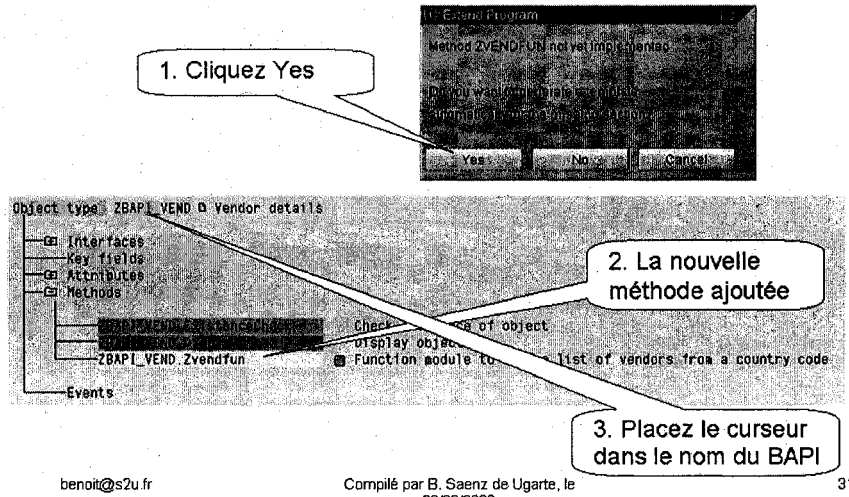
Créer un BAPI (6)



Créer un BAPI (7)

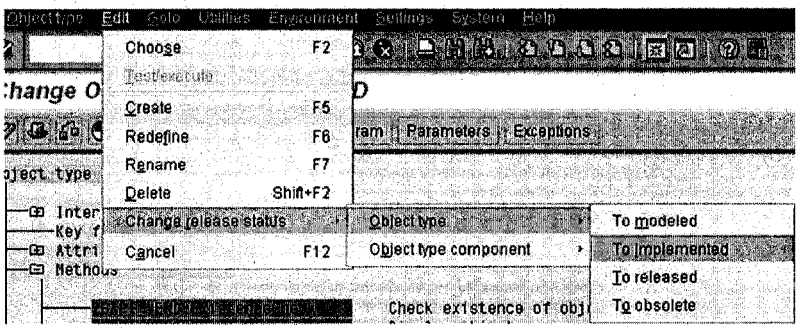


Créer un BAPI (8)



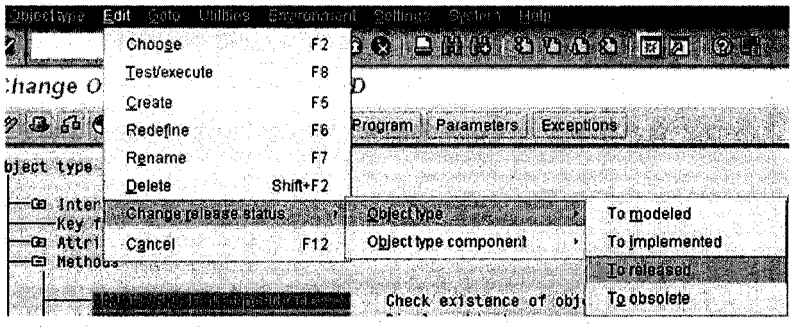
Créer un BAPI (9)

- Mettre le statu du BAPI sur *To implemented*

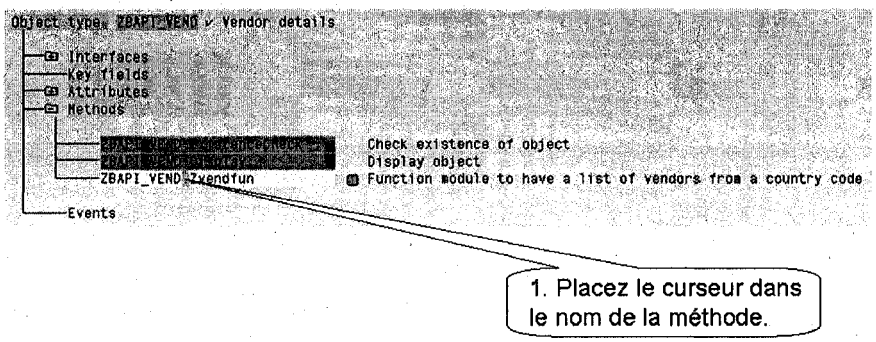


Créer un BAPI (10)

- Mettre le statu du BAPI sur *To released*

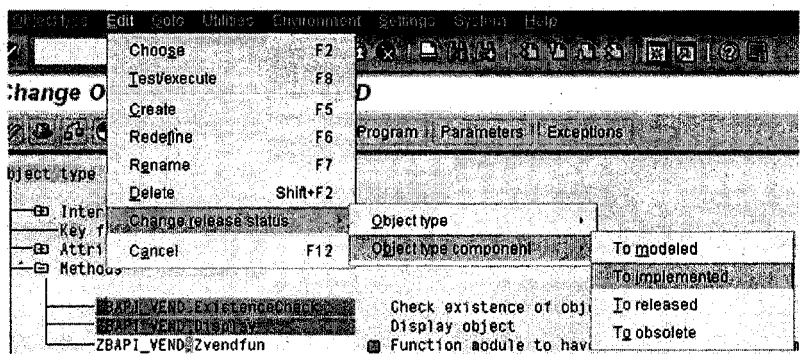


Créer un BAPI (11)



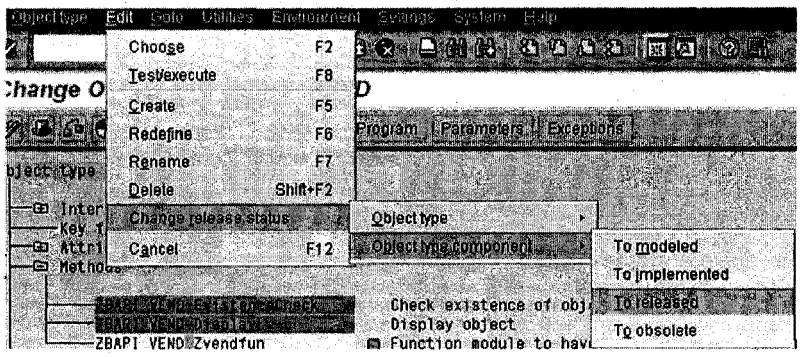
Créer un BAPI (12)

- Mettre le statu de la méthode sur *To implemented*

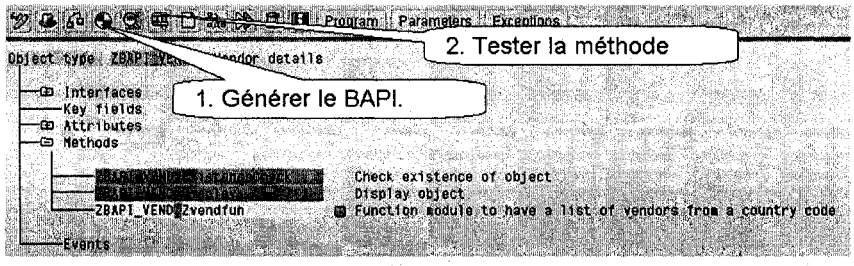


Créer un BAPI (13)

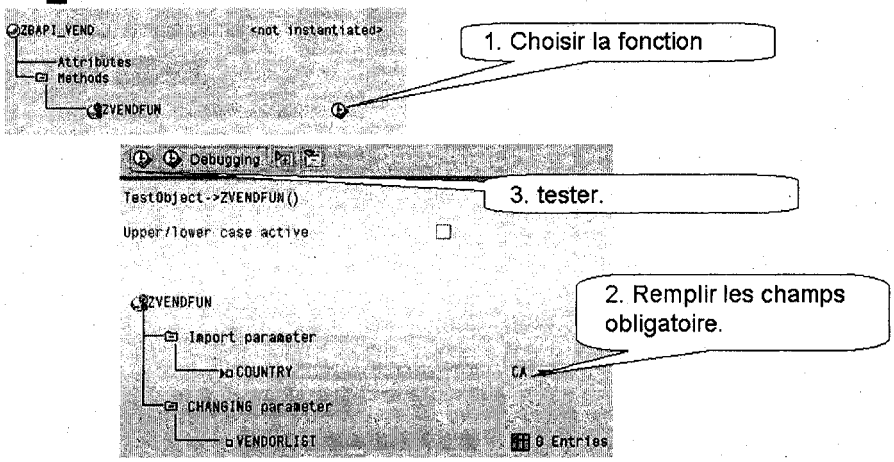
- Mettre le statu de la méthode sur *To released*



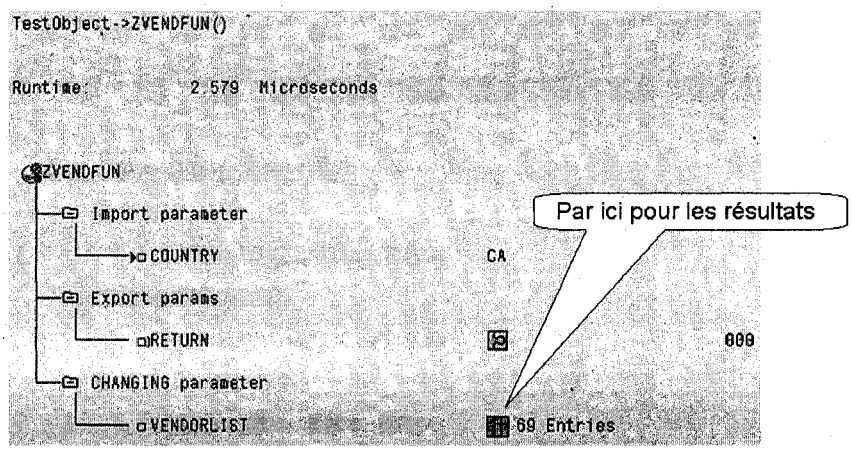
Créer un BAPI (14)



Créer un BAPI (15)



Créer un BAPI (16)



ANNEXE 6

CODE JAVA D'APPEL D'UN BAPI

CODE JAVA D'APPEL D'UN BAPI

```
import com.sap.mw.jco.*;

public class BAPI_ProductionOrder {

	JCO.Client mConnection;
	JCO.Repository mRepository;

	boolean bEtatConnexion;

	public BAPI_ProductionOrder() {
		this.bEtatConnexion = false;
	}

	public void Connexion() {
		try {
			mConnection = JCO.createClient(
					"800",				// Client SAP
					"beugar",			// userid
					"verdiers",			// mot de passe
					"en",				// langue
					"132.207.34.13",		// Hostname
					"00");				// System number
			mConnection.connect();
			mRepository = new JCO.Repository("ARAsoft", mConnection);

			bEtatConnexion = true;
		}
		catch (Exception ex) {
			ex.printStackTrace();
			bEtatConnexion = false;
		}
	}

	public void callBapi(String Material, String FileName) {
		JCO.Function function = null;
		JCO.Structure returnStructure;

		try {
			function =
this.createFunction("ZSHORTPRODORDER_GETLIST");

			if (function == null) {
				System.out.println("ZSHORTPRODORDER_GETLIST
introuvable dans SAP");
				System.exit(1);
```

```
                }

                function.getImportParameterList().setValue("ZPL1", "PLANT");
                function.getImportParameterList().setValue(Material,
"MATERIAL");

                this.mConnection.execute(function);

                returnStructure =
function.getExportParameterList().getStructure("RETURN");

                if (returnStructure.getString("TYPE").equals("S")) {
                        System.out.println("Liste des ordres pour l'alliage " +
Material + " obtenue avec succes.");
                }
                else {
                        System.out.println("echec : " +
returnStructure.getString("TYPE") + " " + returnStructure.getString("MESSAGE"));
                }

                function.writeXML("C:\\Documents and
Settings\\saenzb.P_SAPI3.000\\Mes documents\\My JAVA workspaces\\JCo-
Arasoft\\"+ FileName +".xml");
        }
        catch (Exception ex) {
                ex.printStackTrace();
        }
    }


    public JCO.Function createFunction(String name) throws Exception {
        try {
                IFunctionTemplate ft =
mRepository.getFunctionTemplate(name.toUpperCase());
                if (ft == null) {
                        return null;
                }
                return ft.getFunction();
        }
        catch (Exception ex) {
                throw new Exception("Probleme pour recuperer l'objet
JCO.Function ayant pour nom : " + name);
        }
    }


    public static void main(String[] args) {

        BAPI_ProductionOrder app = new BAPI_ProductionOrder();
```

```
                // login
                app.Connexion();

                // défaut de connexion
                if (app.bEtatConnexion == false) {
                        System.out.println("Pas de connexion etablie.");
                        System.exit(1);
                }
                /*else {
                        System.out.println("Connexion etablie.");
                        System.exit(1);
                }*/

                System.out.println("Connexion etablie.");

                app.callBapi("000000000000000938", "A");
                app.callBapi("000000000000000939", "B");
                app.callBapi("000000000000000940", "C");

                // Deconnexion
                app.mConnection.disconnect();
                System.out.println("Deconnexion.");
        }
}
```

ANNEXE 7

INTRODUCTION À DCOM

INTRODUCTION À DCOM

Ce document est tiré de la FAQ sur DCOM/OLE de Christian Casteyde. Je le reprends en partie pour que le lecteur n'étant pas familiarisé avec cette technologie ne se trouve pas totalement dépourvu.

Qu'est-ce qu'OLE ?

OLE est l'abréviation de Object Linking and Embedding, ce qui signifie « Intégration d'objets et Liens sur des objets ». OLE est une technologie qui a été développée par Microsoft, initialement dans le but de permettre la programmation d'objets capables d'être insérés dans des applications réceptacles soit par intégration complète, soit par référence (ce que l'on appelle une liaison). Ce but a été atteint pour la plupart des applications et apparaît à présent dans le menu « Coller | Collage spécial ». Les objets intégrés ou liés sont capables de s'afficher dans l'application qui les contient, ils sont également capables de fournir un certain nombre de services standards permettant leur manipulation (ces services peuvent être la sauvegarde de leur état, la capacité à être édité, etc.). OLE est donc un remplacement efficace des liaisons DDE.

Qu'est-ce que COM ?

Pour réaliser cet objectif, Microsoft a dû fournir un standard de communication entre les différentes applications qui voulaient être OLE. Ce standard de communication définit la méthode à employer pour accéder aux fonctionnalités des objets OLE, ainsi que les principaux services qui peuvent être nécessaires lors de l'intégration des objets. Les programmeurs désirant réaliser une application OLE devaient se conformer au protocole d'appel du standard et fournir un certain nombre de services définis dans OLE. Ce standard a été conçu de manière ouverte, c'est-à-dire qu'il n'est pas nécessaire de fournir tous les services définis dans OLE pour être fonctionnel. Cependant, plus un objet OLE

offre de service, meilleur son intégration est. De même, plus une application est capable d'utiliser des services, plus elle est fonctionnelle avec les objets qui fournissent ces services. Par ailleurs, il est possible de définir des services différents de ceux définis dans OLE, le système est donc également extensible.

Le standard de communication qui a été défini se nomme COM, ce qui signifie « *Component Objet Model* », ou « Modèle Objet de Composants ». La plupart des services sont définit dans OLE, cependant, COM lui-même utilise un certain nombre de services. La limite entre ce qui est défini par COM et ce qui est défini par OLE est donc floue, mais en principe, les services COM sont tous des services systèmes.

Comme on l'a dit, initialement, OLE devait permettre l'intégration des objets entre applications. En fait, il s'est avéré qu'OLE faisait beaucoup plus que cela : grâce à COM, il permettait l'écriture de composants logiciels réutilisables par différentes applications. Il s'agit donc bien d'une technologie de composants.

Qu'est-ce que DCOM ?

Microsoft a ensuite complété la technologie COM afin de permettre la répartition des composant sur un réseau. L'aspect distribué des composants COM ainsi répartis a donné le nom à cette extension de COM, que l'on appelle simplement DCOM (pour *Distributed COM*). Dans la suite de ce document, on considérera que COM est distribué, on utilisera donc systématiquement le terme DCOM.

Quels sont les services fournis par DCOM ?

Du point de vue du programmeur, DCOM spécifie la manière dont les composants peuvent être utilisés. Du point de vue du système, DCOM spécifie les services que ce dernier doit fournir aux programmeurs. Ce dernier aspect est moins intéressant (ou du

moins il n'intéresse que Microsoft), parce que ces spécifications ne concerne que les éditeurs de logiciels qui désirent implémenter DCOM. Je ne décrierai donc DCOM que du point de vue des programmeurs.

DCOM regroupe les fonctionnalités des composants par interfaces. Une interface est un jeu de fonctions plus ou moins liées sémantiquement, qui permettent d'utiliser un composant. Un composant peut implémenter plusieurs interfaces.

DCOM spécifie la forme des interfaces au niveau binaire. Ces interfaces sont en fait un pointeur sur le tableau de pointeurs de fonctions que les compilateurs C++ pour Windows génèrent pour gérer les méthodes virtuelles des classes C++. Ceci implique qu'il est relativement facile de programmer des composant DCOM et encore plus facile de les utiliser en C++ : les interfaces sont simplement des classes abstraites pures. Cependant, comme cette structure peut être recréée dans d'autres langages, on n'est donc pas obligé d'utiliser le C++. DCOM est donc indépendant du langage. En particulier, on peut programmer et utiliser des composants DCOM en C, en Pascal, en Visual Basic, en Assembleur, en J++.

Les interfaces sont identifiées dans le système par des nombres uniques à 128 bits. Ces nombres sont appelés des GUID (*Globally Unique Identifier*, soit IDentificateur gLobale Unique), ou UUID (*Universally Unique Identifier*, soit Identificateur Universel Unique). Un utilitaire, UUIDGEN.EXE, est fourni pour générer de tels nombres. Ces nombres sont calculés pour être absolument unique dans le monde et en tout temps, ils permettent donc d'éviter toute collision entre les identificateurs des interfaces définies par tous les programmeurs du monde. Ceci implique qu'à chaque interface correspond un GUID et un seul. Une fois qu'une interface a été définie, on ne peut plus la modifier (pas même l'étendre) : on doit refaire une autre interface (cette nouvelle interface pourra cependant hériter des méthodes de l'interface à étendre, mais sera identifiée par son propre UUID). Une interface est donc parfaitement identifié par son UUID, souvent appelé IID (pour

Interface Identifier), et constitue le contrat qu'un composant passe avec ceux qui l'utilisent. Les composants qui disposent de cette interface signifient simplement qu'ils gèrent les fonctionnalités de cette interface, et qu'ils les géreront toujours sous cette forme.

Pour utiliser une interface d'un composant, il faut d'abord la demander. Ce mécanisme permet l'identification dynamique des fonctionnalités d'un composant : soit ce composant serveur gère l'interface demandée et on obtient un pointeur sur cette interface, soit il ne la gère pas et on ne reçoit rien en retour. On ne peut donc utiliser les composants que par l'intermédiaire des interfaces qu'ils gèrent. DCOM spécifie la manière de créer les composants et d'obtenir des interfaces.

Les composants sont également identifiés par un GUID. Comme les composants de COM sont appelés des classes, dont les instance sont les objets OLE eux-mêmes, les GUID qui leurs sont attribués sont appelés CLSID (*CLaSs Identifier*, soit IDentifieur de CLaSse). Tous ces GUID sont stockés dons les entrées spécifiques de la base de registre, que le système peut consulter pour faire fonctionner DCOM. La clé principale qui permet de stocker toutes les informations du système sur DCOM est HKEY_CLASSES_ROOT. [...]

ANNEXE 8

RÈGLES DU MODÈLE DE SIMULATION

RÈGLES DU MODÈLE DE SIMULATION

Pattern 1
Nom : **Entamer_creuset_pat**
Type : rule
Ressources relevantes

creuset (un_creuset)
transformer 1 : Keep
choix : from
expression : creuset.Etat = libre
and Not_Exist **(** une_creuset : une_creuset.Etat = siphonnage **and** une_creuset.MAJ = 1 **)**
and creuset.MAJ = 1
sélection : first
convert rule : creuset.Etat = siphonnage
creuset.Contenu = cuve.produit
creuset.Qté_en_cuves = 1
creuset.Heure_min = cuve.Heure_siph - Tolerance_avant
creuset.Heure_max = cuve.Heure_siph + Tolerance_après

cuve (une_cuve)
transformer 1 : Keep
choix : from
expression : cuve.Etat = à_siphonner
sélection : withmin
expression sélection : cuve.Heure_siph
and cuve.Heure_siph
convert rule : cuve.Etat = siphonnée
cuve. Destination = creuset.Numéro

Pattern 2
Nom : **Ajouter_cuve_pat**
Type : rule
Ressources relevantes

creuset (un_creuset)
transformer 1 : Keep
choix : from
expression : creuset.Etat = siphonnage
and creuset.Qté_en_cuves < Nb_cuves_dans_creuset
and creuset.MAJ = 1
sélection : first
convert rule : creuset.Qté_en_cuves = creuset.Qté_en_cuves + 1
creuset.Heure_min = Max (creuset.Heure_min, cuve.Heure_siph - Tolerance_avant)
creuset.Heure_max = Min (creuset.Heure_max, cuve.Heure_siph + Tolerance_après)

cuve (une_cuve)
transformer 1 : Keep
choix : from
expression : cuve.Etat = à_siphonner
and cuve.Produit = creuset.Contenu
and cuve.Heure_siph >= creuset.Heure_min
and cuve.Heure_siph <= creuset.Heure_max
sélection : withmin
expression sélection : cuve.Heure_siph
and cuve.Heure_siph
convert rule : cuve.Etat = siphonnée
cuve. Destination = creuset.Numéro

Pattern 3
Nom : **Cloturer_creuset_pat**
Type : rule
Ressources relevantes

creuset (un_creuset)
transformer 1 : Keep
choix : from
expression : creuset.Etat = siphonnage
and [
creuset.Qté_en_cuves = Nb_cuves_dans_creuset
or Not_Exist **(** une_cuve : une_cuve.Etat = à_siphonner
and une_cuve.Produit = creuset.Contenu
and une_cuve.Heure_siph >= creuset.Heure_min
and une_cuve.Heure_siph <= creuset.Heure_max **)**
]
and creuset.MAJ = 1
sélection : first
convert rule : creuset.Etat = à_vider

Pattern 4
Nom : **Horloge_pat**
Type : irregular_event
Temps : 0,25
Ressources relevantes

usine (une_usine)
transformer 1 : Keep
convert rule : usine.Num_tranche = usine.Num_tranche

Pattern 5
Nom : **Allocation_four_pat**
Type : rule
Ressources relevantes

four (un_four)
transformer 1 : Keep
choix : from
expression : four.Type = melange
and four.Etat = libre
and four.MAJ = 1
convert rule : four.Contenu = lot.Produit
four.Etat = remplissage
four.Num_lot = lot.IndexOF
four.Qté_en_tonnes = 0
four.Qté_à_faire = lot.Qté_en_tonnes
four.Melange_duree = 4.0

lot (un_ordo_unit)
transformer 1 : NoChange
choix : from
expression : lot.IndexOF = usine.IndexOF
convert rule : lot.Date_debut = Time_now
lot.Numéro_four_mélange = four.Numéro

usine (une_usine)
transformer 1 : Keep
convert rule : usine.IndexOF = usine.IndexOF + 1

Pattern 6
Nom : **Remplissage_four_pat**
Type : rule
Choix de l'objet : with_min
Expression : Pénalité (creuset.Contenu, four.Contenu)
Ressources relevantes

four (un_four)
transformer 1 : NoChange
choix : from
expression : four.Etat = remplissage
and four.Qté_en_tonnes < four.Qté_à_faire
and four.MAJ = 1
convert rule : four.Aux_qté = Min (creuset.Qté_en_cuves * (1 / Nb_creuset_dans_four), four.Qté_à_faire - four.Qté_en_tonnes)
four.Qté_en_tonnes = four.Qté_en_tonnes + four.Aux_qté

four_attente (un_four)
transformer 1 : Keep
choix : from
expression : four_attente.Type = attente
and [
four_attente.Qté_en_tonnes = 0.0
or Pénalité (four_attente.Contenu, four.Contenu) < 0
]
and four_attente.MAJ = 1

creuset (un_creuset)
transformer 1 : Keep
choix : from
expression : creuset.Etat = à_vider
and Pénalité (creuset.Contenu, four.Contenu) >= 0
and creuset.Heure_min <= Time_now
and creuset.Heure_max >= Time_now
and creuset.MAJ = 1
convert rule : creuset.Etat = libre **si** four.Aux_qté = creuset.Qté_en_cuves * (1 / Nb_creuset_dans_four)
creuset.Qté_en_cuves = creuset.Qté_en_cuves - Round (four.Aux_qté / (1 / Nb_creuset_dans_four))
creuset.Destination_type = four.Type
creuset.Destination_num = four.Numéro

usine (une_usine)
transformer 1 : Keep
choix : NoCheck
convert rule : usine.Tonnes_A_dans_B = usine.Tonnes_A_dans_B + four.Aux_qté **si** four.Contenu = B **and** creuset.Contenu = A
usine.Tonnes_A_dans_C = usine.Tonnes_A_dans_C + four.Aux_qté **si** four.Contenu = C **and** creuset.Contenu = A
usine.Tonnes_B_dans_C = usine.Tonnes_B_dans_C + four.Aux_qté **si** four.Contenu = C **and** creuset.Contenu = B

Pattern 7
Nom : **Melange_pat**
Type : operation
Temps : four.Melange_duree
Ressources relevantes

four (un_four)
transformer 1 : Keep
transformer 2 : Keep
choix : from
expression : four.Etat = remplissage
and four.Qté_en_tonnes = four.Qté_à_faire
and four.MAJ = 1
convert begin : four.Etat = mélange
convert end : four.Etat = mélangé **si** 2 = 2

Pattern 8
Nom : **Coulée_pat**
Type : operation
Temps : 0,25
Ressources relevantes

four (un_four)
transformer 1 : Keep
transformer 2 : Keep
choix : from
expression : [
[
four.Etat = mélangé
and Not_Exist **(** un_four : un_four.Etat = coulée **and** un_four.MAJ = 1**)**
]
or four.Etat = coulée
and four.Qté_en_tonnes >= 1.0
]
and four.Step = 0
and four.MAJ = 1
convert begin : four.Etat = coulée
four.Step = 1
convert end : four.Etat = libre **si** four.Qté_en_tonnes = 1
four.Qté_en_tonnes = four.Qté_en_tonnes - 1
four.Step = 0

lot (un_ordo_unit)
transformer 1 : Keep
transformer 2 : Keep
choix : from
expression : lot.IndexOF = four.Num_lot
convert begin : lot.Numéro_four_mélange = four.Numéro
convert end : lot.Date_fin = Time_now **si** four.Qté_en_tonnes = 0

usine (une_usine)
transformer 1 : Keep
transformer 2 : Keep
choix : NoCheck
selection : first
convert begin : usine.Num_tranche = usine.Num_tranche **si** 2 = 2
convert end : usine.A_faire_A = usine.A_faire_A + 1 **si** four.Contenu = A
usine.A_faire_B = usine.A_faire_B + 1 **si** four.Contenu = B
usine.A_faire_C = usine.A_faire_C + 1 **si** four.Contenu = C

Pattern 9
Nom : **Mise_en_attente_pat**
Type : rule
Ressources relevantes

usine (une_usine)
transformer 1 : Keep
choix : NoCheck
selection : first
convert rule : usine.Tonnes_A_dans_B = usine.Tonnes_A_dans_B + four_attente.Qté_en_tonnes **si** four_attente.Contenu = A and creuset.Contenu = B
usine.Tonnes_A_dans_B = usine.Tonnes_A_dans_B + creuset.Qté_en_cuves * (1 / Nb_creuset_dans_four) **si** four_attente.Contenu = B and creuset.Contenu = A
usine.Tonnes_A_dans_C = usine.Tonnes_A_dans_C + four_attente.Qté_en_tonnes **si** four_attente.Contenu = A and creuset.Contenu = C
usine.Tonnes_A_dans_C = usine.Tonnes_A_dans_C + creuset.Qté_en_cuves * (1 / Nb_creuset_dans_four) **si** four_attente.Contenu = C and creuset.Contenu = A
usine.Tonnes_B_dans_C = usine.Tonnes_B_dans_C + four_attente.Qté_en_tonnes **si** four_attente.Contenu = B and creuset.Contenu = C
usine.Tonnes_B_dans_C = usine.Tonnes_B_dans_C + creuset.Qté_en_cuves * (1 / Nb_creuset_dans_four) **si** four_attente.Contenu = C and creuset.Contenu = B

four_attente (un_four)
transformer 1 : Keep
choix : from
expression : four_attente.Type = attente
and four_attente.Qté_en_tonnes < Tonnes_dans_four
and four_attente.MAJ = 1
sélection : first
convert rule : four_attente.Contenu = B **si** four_attente.Contenu = A **and** creuset.Contenu = B
four_attente.Contenu = C **si** four_attente.Contenu = A **and** creuset.Contenu = C
four_attente.Contenu = C **si** four_attente.Contenu = B **and** creuset.Contenu = C
four_attente.Aux_qté = Min (Tonnes_dans_four - four_attente.Qté_en_tonnes, creuset.Qté_en_cuves * (1 / Nb_creuset_dans_four))
four_attente.Qté_en_tonnes = four_attente.Qté_en_tonnes + four_attente.Aux_qté

creuset (une_creuset)
transformer 1 : Keep
choix : from
expression : creuset.Etat = à_vider
and creuset.Heure_max = Time_now
and creuset.MAJ = 1
sélection : with_min
expression de la selection : Pénalité (creuset.Contenu, four_attente.Contenu)

convert rule : creuset.Etat = libre **si** creuset.Qté_en_cuves = Round (
four_attente.Aux_qté / (1 / Nb_creuset_dans_four))
creuset.Qté_en_cuves = creuset.Qté_en_cuves - Round (
four_attente.Aux_qté / (1 / Nb_creuset_dans_four))
creuset.Destination_type = four_attente.Type
creuset.Destination_num = four_attente.Numéro

Pattern 10
Nom : **Vider_four_attente_pat**
Type : rule
Choix de l'objet : with_min
Expression : Pénalité (four_attente.Contenu, four.Contenu)
Ressources relevantes

four (un_four)
transformer 1 : Keep
choix : from
expression : four.Etat = remplissage
and four.Qté_en_tonnes < four.Qté_à_faire
and four.MAJ = 1
convert rule : four.Qté_aux = Min (four.Qté_à_faire - four.Qté_en_tonnes,
four_attente.Qté_en_tonnes)
four.Qté_en_tonnes = four.Qté_en_tonnes + four.Aux_qté

four_attente (un_four)
transformer 1 : Keep
choix : from
expression : four_attente.Type = attente
and four_attente.Qté_en_tonnes > 0.0
and Pénalité (four_attente.Contenu, four.Contenu) >= 0
and four_attente.MAJ = 1
convert rule : four_attente.Contenu = A **si** four_attente.Qté_en_tonnes =
four.Aux_qté
four_attente.Qté_en_tonnes = four_attente.Qté_en_tonnes -
four.Aux_qté

usine (une_usine)
transformer 1 : Keep
choix : NoCheck
convert rule : usine.Tonnes_A_dans_B = usine.Tonnes_A_dans_B + four.Aux_qté **si**
four_attente.Contenu = A and four.Contenu = B
usine.Tonnes_A_dans_C = usine.Tonnes_A_dans_C + four.Aux_qté **si**
four_attente.Contenu = A and four.Contenu = C
usine.Tonnes_B_dans_C = usine.Tonnes_B_dans_C + four.Aux_qté **si**
four_attente.Contenu = B and four.Contenu = C

Pattern 11
Nom : **Initialisation_four_pat**
Type : rule
Choix de l'objet : first
Ressources relevantes

four (un_four)
transformer 1 : Keep
choix : from
expression : four.MAJ = 0
and four.Numéro < 3
convert rule : four.Contenu = four_mirroir.Contenu
four.Etat = four_mirroir.Etat
four.Num_lot = four_mirroir.Num_lot
four.Qté_en_tonnes = four_mirroir.Qté_en_tonnes
four.Qté_à_faire = four_mirroir.Qté_à_faire
four.Melange_duree = four_mirroir.Melange_duree
four.MAJ = 1

four_mirroir (un_four)
transformer 1 : NoChange
choix : from
expression : four_mirroir.Numéro = four.Numéro + 100
and four_mirroir.Type = four.Type

Pattern 12
Nom : **Initialisation_creuset_pat**
Type : rule
Choix de l'objet : first
Ressources relevantes

creuset (une_creuset)
transformer 1 : Keep
choix : from
expression : creuset.MAJ = 0
and creuset.Numéro <= 20
convert rule : creuset.Etat = creuset _mirroir.Etat
creuset.Contenu = creuset_mirroir.Contenu
creuset.Qté_en_cuves = creuset_mirroir.Qté_en_cuves
creuset.Heure_min = creuset_mirroir.Heure_min
creuset.Heure_max = creuset_mirroir.Heure_max
creuset.MAJ = 1

creuset_mirroir (une_creuset)
transformer 1 : NoChange
choix : from
expression : creuset_mirroir.Numéro = creuset.Numéro + 100

Pattern 13
Nom : **Remplir_cuve_pat**
Type : rule
Choix de l'objet : first
Ressources relevantes

cuve (une_cuve)
transformer 1 : Keep
choix : From
expression : cuve.Etat = siphonnée
convert rule : cuve.Produit = Get_property_cuve
cuve.Heure_siph = cuve.Heure_siph + 24 * 1
cuve.Etat = à_siphonner

Pattern 14
Nom : **Scraper_cuve_pat**
Type : rule
Choix de l'objet : first
Ressources relevantes

usine (une_usine)
transformer 1 : Keep
choix : NoCheck
convert rule : usine.Scrap_A = usine.Scrap_A + creuset.Qté_en_cuves **si** creuset.Contenu = A
usine.Scrap_B = usine.Scrap_B + creuset.Qté_en_cuves **si** creuset.Contenu = B
usine.Scrap_C = usine.Scrap_C + creuset.Qté_en_cuves **si** creuset.Contenu = C

creuset (un_creuset)
transformer 1 : Keep
choix : From
expression : Time_now > creuset.Heure_max **and** creuset.Etat = à_vider
convert rule : creuset.Etat = libre
creuset.Qté_en_cuves = 0

Pattern 15
Nom : **Initialiser_usine**
Type : rule
Choix de l'objet : with_max
Expression : OF.IndexOF
Ressources relevantes

usine (une_usine)
transformer 1 : Keep
choix : From
expression : usine.OFs_à_faire = 0
convert rule : usine.OFs_à_faire = OF.IndexOF

OF (un_ordo_unit)
transformer 1 : Keep
choix : From
expression : OF.IndexOF = OF.IndexOF
convert rule : OF.Numéro_four_mélange = OF.Numéro_four_mélange

BIBLIOGRAPHIE

[1] T. Davenport and J. Short, "The new Industrial Engineering: Information Technology and Business Process Redesign," *Sloan Management Review*, pp. 11-27, 1990.

[2] J. Griffin, "Information strategy: Where's the Beef," in *DM Review Magazine*, 1999.

[3] W. Wagner and Y. L. Antonucci, "An analysis of the imagine PA public sector ERP project," Big Island, HI., United States, 2004.

[4] J. A. Orlicky, *Material Requirements Planning*. London: McGraw-Hill, 1975.

[5] W. G. Plossl, *La nouvelle donne de la gestion de la production*. Paris: Afnor gestion, 1993.

[6] W. Hopp and M. Spearman, *Factory Physics*. New-York, NY: McGraw-Hill, 2000.

[7] P. Mandal and A. Gunasekaran, "Application of SAP R/3 in on-line inventory control," *International Journal of Production Economics*, vol. 75, pp. 47-55, 2002.

[8] Y. B. Moon and D. Phatak, "Enhancing ERP system's functionality with discrete event simulation," *Industrial Management and Data Systems*, vol. 105, pp. 1206-1224, 2005.

[9] MESA, "Collaborative Manufacturing Explained," *White Paper*.

[10] MESA, "The Benefits of MES: A Report from the Field," in *White paper 1*. Pittsburg (PA), USA: MESA International, available www.mesa.org, 1997.

[11] Frost and Sullivan, "The European Manufacturing Execution Systems Market." London: Frost & Sullivan Publication Division, 2002, pp. I-65 – 3-26.

[12] AMR, "Produce to Demand is a Manufacturing Strategy, Not an MES Project," *AMR Alert*, 2004.

[13] J. S. Smith and S. B. Joshi, "A shop floor controller class for computer-integrated manufacturing," *International Journal of Computer Integrated Manufacturing*, vol. 8, pp. 327-39, 1995.

[14] G. R. Drake and J. S. Smith, "Simulation system for real-time planning, scheduling, and control," in *1996 Winter Simulation Conference*. Coronado, CA, USA: IEEE, Piscataway, NJ, USA, 1996, pp. 1083-1090.

[15] M. Siemiatkowski and W. Przybylski, "Simulation studies of process flow with in-line part inspection in machining cells," *Journal of Materials Processing Technology*, vol. 171, pp. 27-34, 2006.

[16] K. Kouiss and H. Pierreval, "Implementing an on-line simulation in a flexible manufacturing system," in *Simulation in Industry'99. 11th European Simulation Symposium 1999. ESS'99*. Erlangen, Germany: SCS, 1999, pp. 484-8.

[17] L. Holst, *Integrating Discrete-Event Simulation into the Manufacturing System Development Process*. Lund, Sweden: Division of Robotics, Department of Mechanical Engineering, Lund University, 2001.

[18] MESA, "MES Explained: A High Level Vision," in *White paper 6*. Pittsburg (PA), USA: MESA International, available www.mesa.org, 1997.

[19] MESA, "MES Functionalities and MRP to MES Data Flow Possibilities," in *White paper 2*. Pittsburg (PA), USA: MESA International, available www.mesa.org, 1997.

[20] Camstar, "Products : MESA & InSite [online]," available http://www.camstar.com.

[21] SiView, "IBM SiView Standard," available http://www-06.ibm.com/jp/iisc/english/po/siview/about.html.

[22] Simatic IT, "SIMATIC IT Framework - Production Modeling and Component Coordination," available http://www.sea.siemens.com/mes/product/msitfram.html.

[23] Camstar, "MES automation—Amkor technology—IC assembly," Camstar White Paper, available http://www.camstar.com.

[24] Proficy, "Proficy, Plant Applications," GE Fanuc, available http://www.gefanuc.com/Downloads/en/proficyplantapp_cutsheet_gfa594.pdf.

[25] FactoryTalk, "Integrate Your Manufacturing and Enterprise Information," Rockwell Automation, available http://www.rockwellautomation.com/rockwellsoftware/factorytalk/.

[26] P. Massotte, "Analysis and approaches for the management of complex production systems," in *The Planning and Scheduling of Production Systems*, A. A. a. S. E. Elmaghraby, Ed.: Chapman & Hall, 1997.

[27] S. Jain, "Virtual factory framework: a key enabler for agile manufacturing," Paris, Fr, 1995.

[28] W. Shen and D. H. Norrie, "Agent-based systems for intelligent manufacturing: a state-of-the-art survey," *Knowledge and Information Systems, an International Journal*, vol. 1, pp. 129-156, 1999.

[29] H. Van Dyke Parunak, "What can Agents do in Industry, and Why ? An Overview of Industrially-Oriented R&D at CEC," in *Cooperative Information Agents II*: Lecture Notes of Computer Science 1435:1, pp. 1-18.

[30] P. Valckenaers, H. Van Brussel, F. Bonneville, L. Bongaerts, and J. Wyns, "IMS test case 5. Holonic manufacturing systems," Vienna, Aust, 1994.

[31] D. M. Diltis, N. P. Boyd, and H. H. Whorms, "The evolution of control architectures for automated manufacturing systems," *Journal of Manufacturing Systems*, vol. 10, pp. 63-79, 1991.

[32] P. Leitão and F. Restivo, "A layered approach to distributed manufacturing," in *Advanced Summer Institute International Conference*. Leuven, Belgium, 1999.

[33] PABADIS Group, "Revolutionizing Plant Automation : the PABADIS Approach," Deliverable 6.3, available http://www.pabadis.org/downloads/pabadis_del_6_3.pdf, 2003.

[34] D. Diep, P. Massotte, and A. Meimouni, "A distributed manufacturing execution system implemented with agents: the PABADIS model," Banff, Alta., Canada, 2003.

[35] J. M. Simao, P. C. Stadzisz, and G. Morel, "Manufacturing execution systems for customized production," *Journal of Materials Processing Technology*, vol. 179, pp. 268-275, 2006.

[36] SEMATECH Inc., *Computer Integrated Manufacturing Framework Specification Version 2.0*. Austin, TX, 1998.

[37] G. Harhalakis, L. Chang-Pin, L. Mark, and P. Muro-Medrano, "Implementation of rule-based information systems for integrated manufacturing," *IEEE Transactions on Knowledge and Data Engineering*, vol. 6, pp. 892-908, 1994.

[38] G. Harhalakis, C. P. Lin, L. Mark, and P. R. Muro-Medrano, "Structured representation of rule-based specifications in CIM using updated petri nets," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 25, pp. 130-144, 1995.

[39] C.-P. Lin, L.-D. Jeng, Y.-P. Lin, and M. Jeng, "Management and control of information flow in CIM systems using UML and Petri nets," *International Journal of Computer Integrated Manufacturing*, vol. 18, pp. 107-121, 2005.

[40] L. Chang-Pin and J. MuDer, "An expanded SEMATECH CIM framework for heterogeneous applications integration," *IEEE Transactions on Systems, Man & Cybernetics, Part A (Systems & Humans)*, vol. 36, pp. 76-90, 2006.

[41] J. Barry, M. Aparicio, T. Durniak, P. Herman, J. Karuturi, C. Woods, C. Gilman, H. Lam, and R. Ramnath, "NIIIP-SMART: an investigation of distributed object approaches to support MES development and deployment in a virtual enterprise," La Jolla, CA, USA, 1998.

[42] M. Hori, T. Kawamura, and A. Okano, "OpenMES: scalable manufacturing execution framework based on distributed object computing," Tokyo, Jpn, 1999.

[43] C.-Y. Huang, "Distributed manufacturing execution systems: A workflow perspective," *Journal of Intelligent Manufacturing*, vol. 13, pp. 485-497, 2002.

[44] Z. Bing-hai, W. Shi-jin, and X. Li-feng, "Data model design for manufacturing execution system," *Journal of Manufacturing Technology Management*, vol. 16, pp. 909-35, 2005.

[45] H.-C. Yang, F.-T. Cheng, and D. Huang, "Development of a generic equipment manager for semiconductor manufacturing," Barcelona, Spain, 1999.

[46] F.-T. Cheng, H.-C. Yang, T.-L. Kuo, C. Feng, and M. Jeng, "Modeling and analysis of equipment managers in manufacturing execution systems for semiconductor packaging," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 30, pp. 772-782, 2000.

[47] F.-T. Cheng, M.-T. Lin, and R.-S. Lee, "Developing a web-enabled equipment driver for semiconductor equipment communications," San Francisco, CA, USA, 2000.

[48] S. X. Ye and R. G. Qiu, "An architecture of configurable equipment connectivity in a future manufacturing information system," presented at 2003 IEEE International Symposium on Computational Intelligence in Robotics and Automation, Kobe, Japan, 2003.

[49] R. G. Qiu, "E-manufacturing: The keystone of a plant-wide real time information system," *Journal of the Chinese Institute of Industrial Engineers*, vol. 20, pp. 266-274, 2003.

[50] A. T. Jones, L. H. Reeker, and A. V. Deshmukh, "The Controls Layer: Controls Definition and MES to Controls Data Flow Possibilities," http://www.ifm.eng.cam.ac.uk/mcn/pdf_files/part6_1.pdf, 2002.

[51] Rockwell, "Making sense of e-manufacturing: A roadmap for manufacturers," Rockwell Automation, White Paper. available http://www.automation.rockwwell.com.

[52] D. Slater, "Talk to Your Plants," available http://www.cio.com/archive/031500/plants.html.

[53] R. Qiu and Q. Xu, "Heterogeneous knowledge syntheses for e-manufacturing systems," presented at Internationale Conference Industrial Engineering & Engineering Management, Shanghai, China, 2003.

[54] J. Baliga, "MES and CIM: At the center of productivity," *Semiconductor Int.*, vol. 8, pp. 104-112.

[55] R. Qiu, R. Burda, and R. Chylak, "Distributed WIP control in advanced semiconductor manufacturing," presented at 13th Annual IEEE/SEMI Advanced Semiconductor Manufacturing Conference, Boston, MA, 2002.

[56] R. Qiu, "A data fusion framework for an integrated plant-wide information system," presented at 5th International Conference Information Fusion, Annapolis, MD, 2002.

[57] Q. Xu and R. Qiu, "Heterogeneous knowledge syntheses for manufacturing information systems," *Jisuanji Jicheng Zhizao Xitong/Computer Integrated Manufacturing Systems, CIMS*, vol. 10, pp. 732-736, 2004.

[58] XML: http://www.w3.org/XML/.

[59] XSLT: http://www.w3.org/Style/XSL/.

[60] R. Wrembel and C. Koncilia, *Data Warehouses And Olap: Concepts, Architectures And Solutions*. Hershey, PA: IRM Press, 2006.

[61] K.-Y. Chen and T.-C. Wu, "Data warehouse design for manufacturing execution systems," Taipei, Taiwan, 2005.

[62] MESA, "The Controls Layer: Controls Definition and MES to Controls Data Flow Possibilities," in *White paper 3*. Pittsburg (PA), USA: MESA International, available www.mesa.org, 2000.

[63] APICS, *APICS Dictionary*, 9th ed. Atlanta: APICS Edition, 1998.

[64] T. E. Vollman, W. L. Berry, and D. C. Whybark, *Manufacturing planning and control systems*, 4th ed: New York et al, 1997.

[65] V. A. Mabert, A. Soni, and M. A. Venkataramanan, "Enterprise resource planning survey of U.S. manufacturing firms," *Production and Inventory Management Journal*, vol. 41, pp. 52-58, 2000.

[66] J. Olhager and E. Selldin, "Enterprise resource planning survey of Swedish manufacturing firms," *European Journal of Operational Research*, vol. 146, pp. 365-373, 2003.

[67] M. D. Okrent and R. J. Vokurka, "Process mapping in successful ERP implementations," *Industrial Management and Data Systems*, vol. 104, pp. 637-643, 2004.

[68] K. Reilly, "Enterprise Resource Planning Software Will Grow to $29 Billion in 2006," AMR Research Inc., 2006.

[69] K. J. Musselman and R. Uzsoy, "Advanced planning and scheduling for manufacturing," in *Handbook of Industrial Engineering*, G. Salvendy, Ed., 3rd ed. New York, NY.: Wiley, 2001.

[70] C. A. Toye, Jr., "Let's update capacity requirements planning logic," New Orleans, LA, USA, 1990.

[71] R. S. John, "The cost of inflated planning lead-times in MRP systems," *Journal of Operations Management*, vol. 5, pp. 119-28, 1985.

[72] S. A. Melnyk and C. J. Piper, "Lead-Time Errors in MRP: The Lot-sizing Effect," *International Journal of Production Research*, vol. 23, pp. 253-264, 1985.

[73] S. C. L. Koh and S. M. Saad, "A holistic approach to diagnose uncertainty in ERP-controlled manufacturing shop floor," *Production Planning and Control*, vol. 14, pp. 273-289, 2003.

[74] R. A. Lindau and K. R. Lumsden, "Actions taken to prevent the propagation of disturbances in manufacturing systems," *International Journal of Production Economics*, vol. 41, pp. 241, 1995.

[75] S. C. L. Koh and S. M. Saad, "Managing uncertainty in ERP-controlled manufacturing environments in SMEs," *International Journal of Production Economics*, vol. 101, pp. 109-127, 2006.

[76] V. D. R. Guide Jr and R. Srivastava, "A review of techniques for buffering against uncertainty with MRP systems," *Production Planning and Control*, vol. 11, pp. 223-233, 2000.

[77] S. C. Koh, M. H. Jones, S. M. Saad, S. Arunachalam, and A. Gunasekaran, "Measuring uncertainties in MRP environments," *Logistics Information Management*, vol. 13, pp. 177-183, Jun 2000.

[78] C. J. Ho, W. K. Law, and R. Rampal, "Uncertainty-dampening methods for reducing MRP system nervousness," *International Journal of Production Research*, vol. 33, pp. 483, 1995.

[79] D. N. P. Murthy and L. Ma, "Material planning with uncertain product quality," *Production Planning and Control*, vol. 7, pp. 566-576, 1996.

[80] J. A. G. Krupp, "Safety stock management," *Production and Inventory Management Journal*, vol. 38, pp. 11-18, 1997.

[81] L. Brennan and S. M. Gupta, "A structured analysis of material requirements planning systems under combined demand and supply uncertainty," *International Journal of Production Research*, vol. 31, pp. 1689-707, 1993.

[82] ITGROUP-BGM, "De la planification locale à l'optimisation globale : l'interaction dynamique entre les plans," rapport interne, disponible sur www.supplychaincenter.com, 1999.

[83] T. H. Davenport and J. D. Brooks, "Enterprise systems and the supply chain," *Journal of Enterprise Information Management*, vol. 17, pp. 8-19, 2004.

[84] R. Howells, "ERP needs shop-floor data," *Manufacturing Engineering*, vol. 125, pp. 54-56, 2000.

[85] Y. Mustafa and O. Mejabi, "An approach for developing flexible MRP systems," *Information Systems Management*, vol. 16, pp. 58-63, 1999.

[86] W. H. Ip, C. K. Kwong, and R. Fung, "Design of maintenance system in MRPII," *Journal of Quality in Maintenance Engineering*, vol. 6, pp. 177-191, 2000.

[87] K. Nikolopoulos, K. Metaxiotis, N. Lekatis, and V. Assimakopoulos, "Integrating industrial maintenance strategy into ERP," *Industrial Management and Data Systems*, vol. 103, pp. 184-191, 2003.

[88] A. Rizzi and R. Zamboni, "Efficiency improvement in manual warehouses through ERP systems implementation and redesign of the logistics processes," *Logistics Information Management*, vol. 12, pp. 367-77, 1999.

[89] P. Mertens, J. Falk, and S. Spieck, "Comparisons of agent approaches with centralized alternatives based on logistical scenarios," *Information Systems*, vol. 19, pp. 699-709, 1994.

[90] D. Wong, N. Paciorek, and D. Moore, "Java-based mobile agents," *Communications of the ACM*, vol. 42, pp. 92-102, 1999.

[91] K. Wong, "We gather today to join ERP and PLM: Marrying enterprise data to product data," *Cadalyst*, vol. 23, pp. 42-44, 2006.

[92] A. F. Salam, H. R. Rao, and S. Bhattacharjee, "Internet-based technologies: value creation for the customer and the value chain across industries," Milwaukee, WI, USA, 1999.

[93] R. Lobecke and T. Slawinski, "Integrated manufacturing execution systems - history and current state: from system integration to overall optimisation of business processes," *Automatisierungstechnische Praxis*, vol. 46, pp. 21-5, 2004.

[94] S. B. Gershwin, *Manufacturing Systems Engineering*: Prentice Hall, March 1993.

[95] D. G. Luenberger, *Linear and Nonlinear Programming*: Addison-Wesley, Facsimile edition, 1984.

[96] A. M. Law and M. G. McComas, "Simulation-based optimization," in *2002 Winter Simulation Conference*, vol. 1. San Diego, CA, United States: Institute of Electrical and Electronics Engineers Inc., 2002, pp. 41-44.

[97] S. Olafsson and J. Kim, "Simulation optimization," in *2002 Winter Simulation Conference*, vol. 1. San Diego, CA, United States: Institute of Electrical and Electronics Engineers Inc., 2002, pp. 79-84.

[98] A. W. M. Lung, "Benefits of a systematic approach for problem analysis on manufacturing simulation modelling," in *1998 International Conference on Simulation, IEE Conference Publication*, 457 ed. York, UK: IEE, Stevenage, Engl, 1998, pp. 271-276.

[99] N. Melao and M. Pidd, "Use of business process simulation: A survey of practitioners," *Journal of the Operational Research Society*, vol. 54, pp. 2-10, 2003.

[100] D. Scott, "Comparative advantage through manufacturing execution systems," presented at the 1996 7th Annual IEEE/SEMI Advanced Semiconductor Manufacturing Conference, ASMC 96, Cambridge, MA, USA, 1996.

[101] D. G. Watt, "Integrating simulation based scheduling with MES in a semi-conductor fab," in *1998 Winter Simulation Conference*, vol. 2. Washington, DC, USA: IEEE, Piscataway, NJ, USA, 1998, pp. 1713-1715.

[102] F. Shin, B. Ram, A. Gupta, X. Yu, and R. Menassa, "A decision tool for assembly line breakdown action," in *2004 Winter Simulation Conference*, vol. 2. Washington, DC, United States: Institute of Electrical and Electronics Engineers Inc., New York, NY 10016-5997, United States, 2004, pp. 1122-1127.

[103] C. S. Chong, R. Gay, and A. I. Sivakumar, "Simulation based scheduling using a two-pass approach," in *2003 Winter Simulation Conference*, vol. 2. New Orleans, LA, United States: Institute of Electrical and Electronics Engineers Inc., 2003, pp. 1433-1439.

[104] O. Bagatourova and S. K. Mallya, "Coupled heuristic and simulation scheduling in a highly variable environment," in *2004 Winter Simulation Conference*, vol. 2. Washington, DC, United States: Institute of Electrical and Electronics Engineers Inc., New York, NY 10016-5997, United States, 2004, pp. 1856-1859.

[105] P. Hwa-Gyoo, B. Jong Myung, P. Sang Bong, and L. Chan Ho, "A development of object-oriented simulator for manufacturing execution systems," in *24th International Conference on Computers and Industrial Engineering*, vol. 37, 1-2 ed. Uxbridge, UK: Elsevier, 1999, pp. 239-42.

[106] S. Weygandt, "Getting the MES model - methods for system analysis," *ISA Transactions*, vol. 35, pp. 95-103, 1996.

[107] G. Morel, H. Panetto, M. Zaremba, and F. Mayer, "Manufacturing Enterprise Control and Management System Engineering: Paradigms and Open Issues," *Annual Reviews in Control*, vol. 27 II, pp. 199-209, 2003.

[108] F.-T. Cheng and C.-Y. Teng, "An object-based controller for equipment communications in semiconductor manufacturing," *Robotics and Computer-Integrated Manufacturing*, vol. 18, pp. 387-402, 2002.

[109] Y. T. Lee and L. Yan, "Data exchange for machine shop simulation," in *2005 Winter Simulation Conference (IEEE Cat. No.05CH37732C)*. Orlando, FL, USA: IEEE, 2005, pp. 7 pp.

[110] A. Hanisch, J. Tolujew, and T. Schulze, "Initialization of online simulation models," in *2005 Winter Simulation Conference (IEEE Cat. No.05CH37732C)*. Orlando, FL, USA: IEEE, 2005, pp. 1795-1803.

[111] G. A. Harrison, D. S. Maynard, and E. Pollak, "Automated database and schema-based data interchange for modeling and simulation," in *2004 Winter Simulation Conference*, vol. 1. Washington, DC, United States: Institute of Electrical and Electronics Engineers Inc., New York, NY 10016-5997, United States, 2004, pp. 191-196.

[112] G. Qiao, F. Riddick, and C. McLean, "Data driven design and simulation system based on XML," in *2003 Winter Simulation Conference*, vol. 2. New Orleans, LA, United States: Institute of Electrical and Electronics Engineers Inc., 2003, pp. 1143-1148.

[113] C. Koch, "Why your integration efforts end up looking like this..." *CIO*, vol. 15, pp. 98-108, 2001.

[114] T. Westerlund, "ERP and MES integration: Reducing cycle time," Chicago, IL, USA, 1996.

[115] P. J. Rondeau and L. A. Litteral, "Evolution of manufacturing planning and control systems: From reorder point to enterprise resource planning," *Production and Inventory Management Journal*, vol. 42, pp. 1-7, 2001.

[116] W. Liu, T. J. Chua, J. Lam, F. Y. Wang, T. X. Cai, and X. F. Yin, "APS, ERP and MES systems integration for Semiconductor Backend Assembly," Marine Mandarin, Singapore, 2002.

[117] T. Erl, *Service-Oriented Architecture: Concepts, Technology, and Design*. Upper Saddle River: Prentice Hall PTR, 2005.

[118] T. Homem-de-Mello, A. Shapiro, and M. L. Spearman, "Finding optimal material release times using simulation-based optimization," *Management Science*, vol. 45, pp. 86-101, 1999.

[119] V. Emelyanov and S. Iassinovski, "Production simulator of manufacturing systems and processes," *Vestnik mashinostroeniya*, vol. 5, pp. 41-44, 1992.

[120] A. Artiba, V. Emelyanov, and S. Iassinovski, "Introduction to intelligent simulation : The RAO language," *Kluwer Academic Publishers*, 1998.

[121] A. Artiba, S. Iassinovski, and C. Raczy, "A method of complex discrete systems representation for hybrid decision making applications development," *Proceedings of International conference on industrial engineering and production management (IEPM'99), Glasgow, July 12-15, 1999*, vol. 2, pp. 222-231, 1999.

[122] J. A. Hernandez, F. Martinez, and J. Keogh, *SAP R/3 Handbook, Third Edition*: McGraw-Hill Osborne Media, 2005.

[123] P. Buxmann, W. König, M. Fricke, F. Hollich, L. M. Diaz, and S. Weber, *Inter-organizational Cooperation with SAP Solutions*: Springer, 2004.