EPM/RT - 93/13

UN SIMULATEUR ORIENTÉ-OBJET POUR L'ANALYSE OPÉRATIONNELLE ET ÉCONOMIQUE DES SYSTÈMES MANUFACTURIERS

Michel Fabre, M.Sc.A.
Jean-Claude Pitre, B.ing.
Daniel Leblanc, Ph.D.

Département de génie industriel

École Polytechnique de Montréal

juillet 1993

Dépôt légal, juillet 1993 (juillet 1993)
Bibliothèque nationale du Québec
Bibliothèque nationale du Canada

Pour se procurer une copie de ce document, s'adresser:

Les Éditions de l'École Polytechnique
École Polytechnique de Montréal
Case postale 6079, Succursale A
Montréal (Québec) H3C 3A7
Téléphone: (514) 340-4473
Télécopie: (514) 340-3734

Compter 0,10 $ par page et ajouter 3,00 $ pour la couverture, les frais de poste et la manutention. Régler en dollars canadiens par chèque ou mandat-poste au nom de l'École Polytechnique de Montréal.

Nous n'honorerons que les commandes accompagnées d'un paiement, sauf s'il y a eu entente préalable dans le cas d'établissements d'enseignement, de sociétés ou d'organismes canadiens.

Sommaire

Un Simulateur Orienté-Objet pour l'Analyse Opérationnelle et Économique des Systèmes Manufacturiers [1]

par

Michel Fabre, Jean-Claude Pitre et Daniel Leblanc

Département de Génie Industriel
École Polytechnique de Montréal
C.P. 6079, Succ. "A", Montréal, Québec, H3C 3A7

. Résumé

Le projet ASMÉMA (ASsistance MÉcanique à la MAnutention) a conduit au développement d'un logiciel d'analyse multi-critères pour l'étude des postes de manutention. Nous présentons ici le module de simulation par événements discrets de ce logiciel, destiné à effectuer les évaluations opérationnelles et économiques des postes en tenant compte des effets systémiques. Ce module se compose d'un simulateur, d'une librairie d'objets manufacturiers, d'un gestionnaire de modèles, d'un compilateur et d'une interface multi-fenêtrée auxquels s'ajoute une librairie d'instruments de mesures pour la partie analyse proprement dite. Ce module met en évidence l'intérêt de la représentation par objets dans le cadre de l'analyse des systèmes manufacturiers. La quasi bijection existant entre objets "réels" et informatiques facilite le développement de modèles par l'usager. La librairie d'instruments permet par ailleurs d'effectuer des mesures précises quasiment n'importe où et n'importe quand au cours de la simulation. Les développements actuels prévoient l'extension de cette librairie, l'utilisation du graphisme pour la modélisation statique et dynamique (avec l'usage de graphes de déroulement matière modifiés) et pour l'animation du modèle.

1. Le projet ASMÉMA

Le projet ASMÉMA (METIS 1992), dont le module de simulation est présenté ici, a vu le développement d'un système intégré de conception rationnelle de postes de manutention adaptés aux caractéristiques technologiques, ergonomiques et économiques contextuelles d'un système de production. Le but premier de ce projet est de favoriser la réduction significative, en qualité et en gravité, des accidents qui accompagnent fréquemment les tâches de manutention manuelle dans l'industrie.

Aux États-Unis, des études sur les accidents du travail, concluent que 25 à 30 % des blessures

[1] Ce texte a été présenté au 61e congrès de l'ACFAS, Rimouski, Québec, 17-21 Mai 1993

infligées à des travailleurs le sont à l'occasion d'une tâche de manutention (NIOSH 1981). La région du corps humain la plus fréquemment touchée dans ces cas est alors la colonne vertébrale. Les nombreuses recherches entreprises se heurtent à l'un ou l'autre des deux aspects suivants de la problématique des maux de dos:

- l'identification des causes d'accidents;
- la recherche de solutions.

Ainsi, les démarches actuelles, pour la résolution du problème des lésions au dos, suggèrent plutôt l'élimination à la source des causes d'accidents (causes réelles ou potentielles). L'éventail des solutions envisageables s'étend de la prévention par le contrôle jusqu'à l'automatisation, en passant par l'assistance mécanique. Quel que soit le moyen choisi pour corriger les inadaptations du travail, l'intervention s'exerce essentiellement à deux niveaux:

- l'action sur l'opérateur;
- l'action sur la tâche et son organisation.

Dans le deuxième cas, qui nous intéresse plus particulièrement, on tente, par l'intégration des connaissances technologiques et de l'organisme humain, soit de modifier la tâche de l'opérateur, soit de transformer son rôle en exploitant ses capacités mentales (perception, raisonnement, mémoire,...) plutôt que physiques, soit encore de supprimer son rôle en s'engageant alors dans un processus d'automatisation.

En dépit de l'intérêt grandissant témoigné par les industries québécoises à l'égard de l'automatisation et de la robotique, on compte aujourd'hui encore un nombre considérable de postes de travail où l'homme constitue la source d'énergie exclusive ou partielle. Les problèmes de reconnaissance des formes, la difficulté d'adaptation aux divers environnements industriels, la monopolisation de grandes surfaces d'usine, l'importance du coût de telles installations et du volume de production nécessaires à la rentabilisation de l'investissement, l'hésitation des travailleurs face au développement des machines automatisées dans les usines (pertes d'emplois, déshumanisation du travail,...) expliquent, en partie, cette situation. Si l'automatisation est une solution qui sera de plus en plus utilisée, il n'en reste pas moins que cette évolution restera lente et qu'à son terme elle ne sera pas générale.

La solution intermédiaire de l'assistance mécanique est donc intéressante et peut se révéler appropriée dans le contexte décrit. En effet, en plus de réduire le nombre d'accidents, elle répond mieux, selon notre expérience, aux besoins et attentes des parties intéressées, maintient les emplois, réduit les coûts sociaux et économiques et s'applique facilement à différents milieux industriels. L'assistance mécanique à la manutention (équipements et accessoires de levage, matériel de mise à niveau, chariots automoteurs, palettiseurs...) vise à faciliter le transport de marchandises tant au niveau de l'effort en tant que tel et de sa répétition qu'à celui du positionnement, du guidage, du support des charges.

L'étude des différentes solution envisageables se fait suivant une analyse multi-critères (ergonomiques, économiques et opérationnels). Toute solution retenue, notamment celle d'assistance mécanique à la manutention, se justifie donc par l'atteinte d'objectifs ergonomiques, économiques et opérationnels tel ceux énumérés ci-dessous:

- assister dynamiquement l'opérateur dans les tâches au niveau des efforts à fournir, de leur répétition et du positionnement des objets manipulés;
- réduire les incertitudes qui induisent un risque pour l'opérateur dans l'exécution de sa tâche;
- donner l'accès au poste à une classe plus large d'opérateurs;
- entraîner des diminutions de coûts directs et indirects par diminution des blessures subies par les opérateurs;
- de préférence, ne pas provoquer d'augmentation de coûts de production non compensée;
- en cas de bilan économique tangible négatif, compenser par bénéfices intangibles suffisants;
- faciliter les opérations sans pour autant les gêner afin qu'il n'y ait pas de détérioration des performances en regard des quantités produites, de la qualité du produit et des délais de production.

2. Le logiciel ASMÉMA

Le logiciel que nous avons développé permet de modéliser un poste de travail et les relations avec son environnement en mettant donc en exergue le rôle des opérateurs. Il permet aussi d'évaluer les différentes configurations de poste (entièrement manuel jusqu'à entièrement automatisé) suivant les trois catégories de critères énoncées ci-dessus. Cet outil sert d'une part, à faciliter la justification de la conception ou de la modification d'un poste de travail, dans l'optique du problème automatisation/non-automatisation (Fabre & al. 1992) et d'autre part, à proposer des améliorations sur le poste de travail en profitant de ces différents critères d'analyse pour établir des comparaisons entre les alternatives étudiées en intégrant, par rapport aux systèmes traditionnels, la dimension ergonomique. Il se compose essentiellement de deux modules:

- le module de conception graphique et d'analyse ergonomique (fig. 1);
- le module de simulation et d'analyse opérationnelle et économique (présenté plus en détail dans les prochains paragraphes).

Le premier bloc est utilisé à des fins de modélisation graphique 3D et constitue de ce fait un véritable module de DAO. Il permet de représenter un espace de travail qui est reconstruit interactivement à l'écran à l'aide de formes géométriques simples. Il offre également la possibilité de mettre en situation des mannequins qui matérialisent les opérateurs du poste. Ces mannequins sont articulés et permettent d'étudier différentes postures prises par les opérateurs afin d'évaluer leur dangerosité grâce à des instruments de mesures ergonomiques. La conception

de l'espace de travail et des mannequins ainsi que leur manipulation ne nécessitent aucun codage de la part de l'usager et peuvent être réalisé en manipulant des boutons et des glissières à l'écran avec la souris. L'accès à l'information est également facilité par une interface multi-fenêtrée.

Les informations issues de ce premier bloc (conception), qui représentent les caractéristiques physiques des objets présents dans le modèle, sont reprises pour la simulation, tandis que certaines informations découlant de cette dernière (estimation de fréquence de phénomènes) sont utilisées en retour par les instruments ergonomiques.

Ce logiciel a été développé sur des stations de travail Silicon Graphics (retenues en raison de leurs performances graphiques 3D), en utilisant le langage C++ (Wiener & Pinson 1988), le système de fenêtrage X (Barkakati 1991) et la librairie Motif (OSF 1991).

Les composants du logiciel ASMÉMA ont été développés suivant les principes de la programmation orientée-objet (POO). La POO prend de plus en plus d'importance dans le développement d'applications manufacturières. Elle conduit à assembler des blocs de code indépendants et autonomes pour développer de nouvelles applications. Ce type de représentation assure un découpage rationnel du code et simplifie sa maintenance.

Dans le domaine de la simulation, les produits actuellement disponibles sur le marché offrent des caractéristiques relativement similaires et sont pour la plupart basés sur des principes de programmation séquentielle qui rendent malaisée leur évolutivité. La POO permet au contraire de construire des logiciels modulaires et de les doter de moyens d'interaction spécifiques plus conviviaux, élargissant ainsi leur champ d'application, tant pour le support à l'usager lors d'une étude de simulation, que pour la variété de ces études. L'intérêt de ce type de modélisation réside principalement dans la possibilité:

- d'associer un objet informatique à un objet physique (notion de classe et d'"encapsulation");
- de modéliser à différents niveaux d'abstraction et donc de détail un système industriel (hiérarchisation);
- de concevoir des librairies spécifiques à partir des objets existant (héritage, réutilisation, modularité);

et d'autres avantages en ce qui concerne la simulation, comme on peut le voir par ailleurs (Adiga & Glassey 1991, Roberts & Heim 1988). La comparaison de ce type de représentation avec la programmation classique peut être trouvée par ailleurs (Mize & al. 1992). Les principaux langages utilisés en simulation OO sont Smalltalk 80 (Mize & al. 1992), C++ (Eldredge & al. 1990, Choi & Minoura 1991) et Objective C (Adiga & Glassey 1991). Dans un tel contexte, un logiciel de simulation basé sur ce principe tend à devenir un environnement de modélisation articulé autour de la représentation du système industriel.

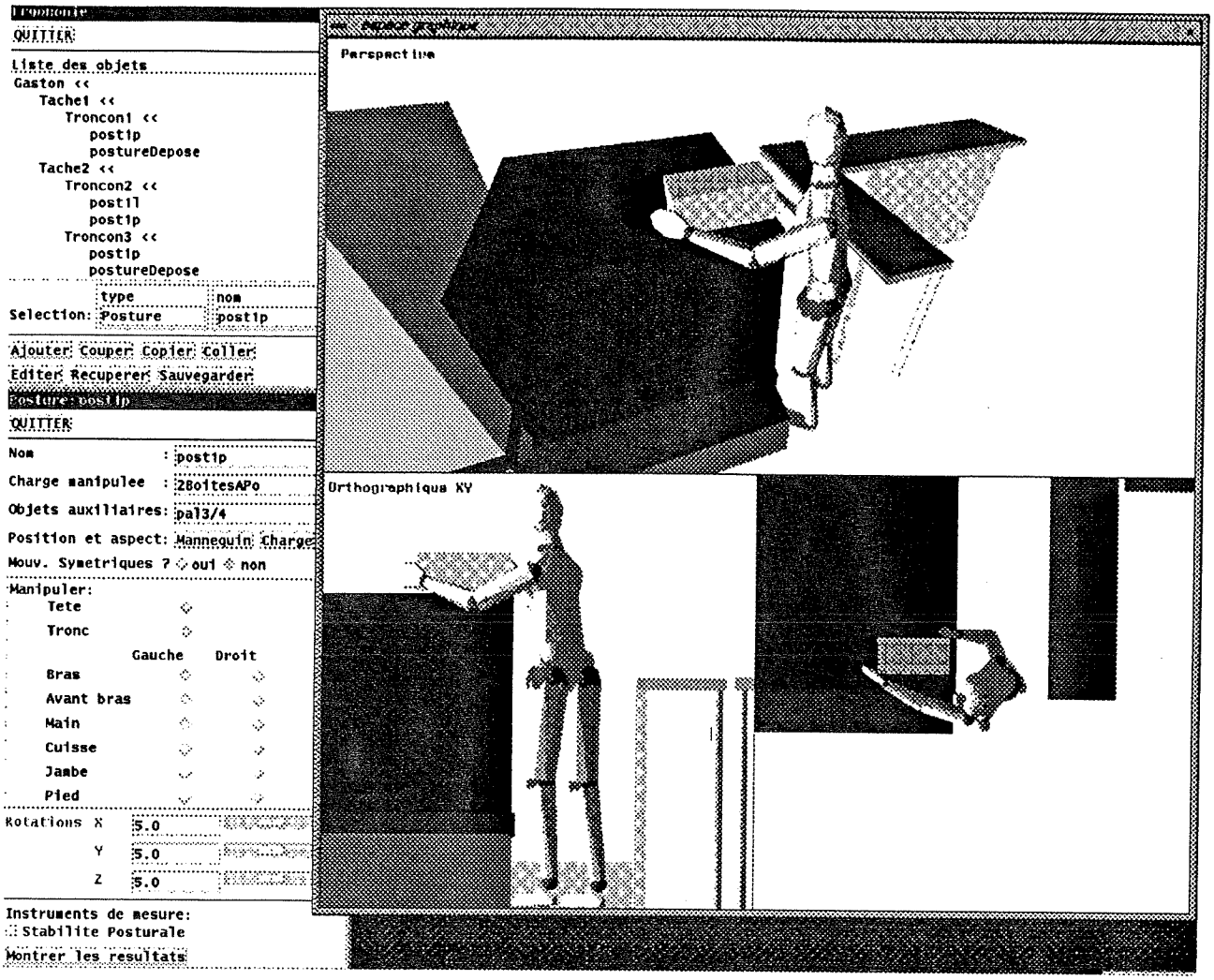

Figure 1: Le module de conception graphique et d'ergonomie

Le problème principal de la simulation OO est la contrepartie de sa modularité: les modèles réalisés réclament beaucoup d'espace mémoire et les échanges de messages entre les différents objets ainsi que les allocations et libérations dynamiques qui accompagnent les apparitions et disparitions d'objets, tendent à ralentir le processus dans son ensemble. Des travaux font état de procédures pour optimiser la gestion de l'espace mémoire (Beaumariage & Roberts 1991) et du traitement parallèle des informations (Fujimoto 1990). Si la première solution ne semble pas offrir de résultats intéressants, la deuxième avenue par contre est beaucoup plus prometteuse. Dans ce cas toutefois, se pose la question de la synchronisation des processus parallèles et de leur dépendance qui reste encore à résoudre. Une autre utilisation, plus immédiate, du parallélisme est la réalisation simultanée de plusieurs réplication de la même expérience. Finallement, les progrès du matériel informatique permettent de penser que la limitation actuelle de la simulation OO n'est pas une raison suffisante pour abandonner cette direction.

3. Le module de simulation

3.1 Les outils de simulation pour l'analyse des systèmes manufacturiers

Les outils de simulations disponibles sur le marché peuvent être classés dans deux grandes catégories: les langages de programmation (FORTRAN, Pascal, C) et de simulation (Siman, SLAM, GPSS...) d'une part, et les simulateurs d'autre part (Witness, SimFactory, FACTOR, ProModel...) (voir, par exemple, WSC 1990 pour une présentation de ces outils). Dans leur ensemble, ces outils ne couvrent que très partiellement les étapes d'une étude de simulation, qui se décompose globalement une fois le problème défini, en des phases de: recherche et mise en forme des informations, modélisation, validation du modèle, définition du scénario expérimental, simulations; analyse des résultats, conclusion et recommandations, avec des bouclages éventuels entre ces étapes. L'évaluation des outils disponibles peut donc se faire suivant leur aptitude à faciliter l'accomplissement de ces tâches. Le tableau 1 présente une synthèse des étapes d'un projet de simulation et la façon dont les différents types de logiciels de simulation disponibles actuellement les accomplissent, tandis que le tableau 2 présente la même information pour les outils d'appoints (externes, comme UNIFIT, ou d'enrobage, comme TESS) et les nouveaux types de simulateurs (AutoMod, Simple++...).

	Langages de 4ème gén.	Langages de simulation	Simulateurs spécialisés
Recueil des données	–	–	–
Organisation des données	–	–	+
Modélisation	–	=	+
Conception d'expérience	–	=	=
Exécution de simulation	–	+	+
analyse de résultats	–	–	–
Conclusion (Rapport)	–	=	=
Facilité d'utilisation	–	=	+
Réutilisation de modèles	–	=	=
Flexibilité	++	+	=
Programmation nécessaire	oui	oui	non

+ : bonne = : moyenne - : absente

Tableau 1: Couverture des étapes de simulation (1)

	Outils externes	Outils internes	Interfaces intelligentes	Générateurs de simulation	KBS
Recueil des données	–	–	–	–	–
Organisation des données	–	+	+	+	+
Modélisation	+	+	+	+	+
Conception d'expérience	+	–	–	–	+
Exécution de simulation	–	–	–	–	+
Analyse de résultats	+	–	+	–	+
Conclusion (Rapport)	+	+	–	+	+
Facilité d'utilisation	=	=	+	+	=
Réutilisation de modèles	–	=	=	=	=
Flexibilité	–	–	=	=	=
Programmation nécessaire	non	oui	oui/non	non	non

Tableau 2: Couverture des étapes de simulation (2)

Au terme de cette synthèse, présentée dans Fabre et Leblanc (1993), on a pu constater:

- le besoin d'outils de modélisation, de définition de scénarios et d'analyse des résultats de la simulation pour assister l'usager et "expertiser" le logiciel;
- l'émergence d'une vision d'ensemble, qui privilégie le mode de représentation OO en intègrant les modèles de simulation au processus de production, ce qui oblige à se départir des schémas de développement procéduraux qui ont prévalu jusqu'à présent.

Ces constatations nous ont conduits à développer notre propre environnement modulaire et évolutif de simulation OO.

3.2 Les principaux composants du module de simulation

Le module de simulation comprend pour l'instant sept éléments distincts (fig. 2). La librairie manufacturière et le module de simulation sont les éléments clés de cet environnement.

L'interface

L'interface est couplée au gestionnaire de modèles et de composants dont elle présente le mode d'organisation. Son aspect n'est pas définitif, il s'agit d'un prototype qui nous a permis d'accéder rapidement aux autres éléments de l'environnement pour les tester. De nouveaux éléments destinés à offrir une pleine modularité au niveau de la description graphique du modèle sont actuellement en cours de réalisation et comprennent principalement des structures de

graphes ainsi que des icones. Néanmoins, l'usage de fiches (fig. 3) pour saisir les caractéristiques des objets est acquis. Les champs, suivant leur type, "connaissent" la nature des informations (numériques ou non) qu'ils sont censés recevoir et signalent les erreurs de frappe. Les commentaires donnent accès à un système d'aide "on-line" qui détaille le type d'information demandé pour ce champ particulier. Les commentaires, messages d'aide ainsi que les types et valeurs par défaut des champs sont définis dans des fichiers ASCII et sont accessibles et modifiables par l'usager qui peut ainsi reconfigurer une fiche sans avoir à compiler le modèle de nouveau (en respectant l'ordre des informations à saisir et leur nature). Il en est de même pour l'ensemble des boutons et des messages de l'interface.

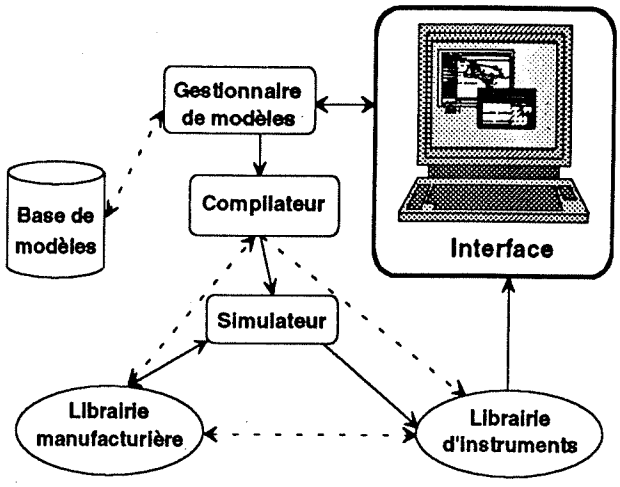

Figure 2: Organisation du module de simulation

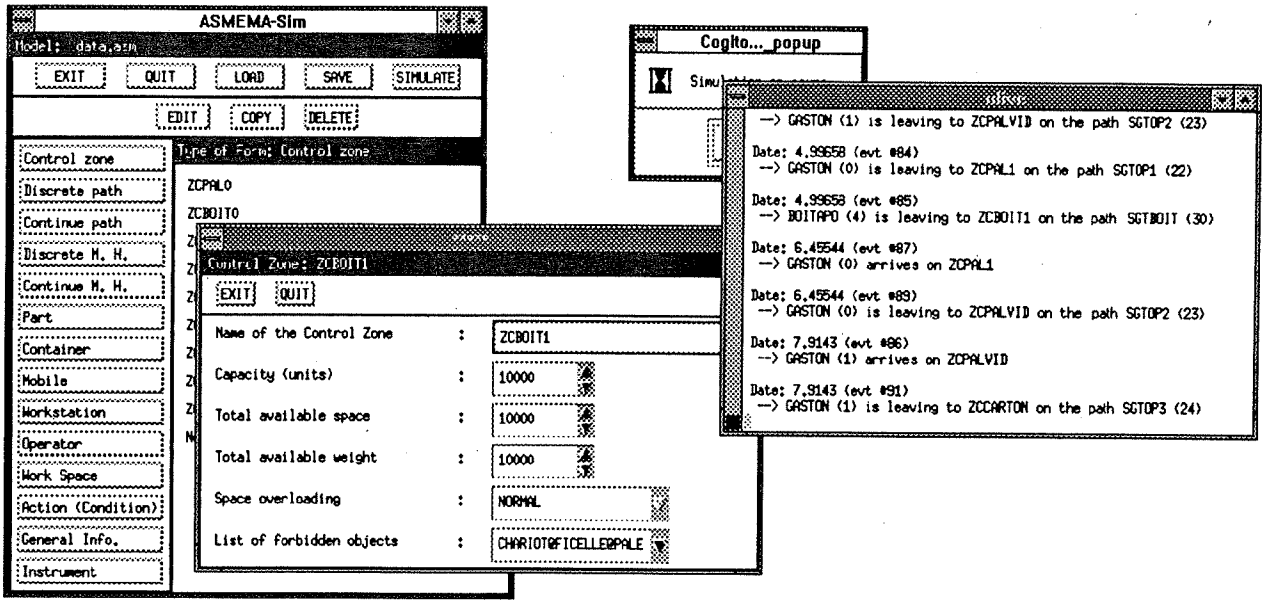

Figure 3: Aperçu de l'interface du module de simulation

Le gestionnaire de modèles

Comme l'interface, ce module est un prototype. Les fiches sont validées dans leur ensemble (validation intra et inter champs) au moment de leur sauvegarde et par rapport au format des informations et à leur teneur, quand ce type de contrôle est possible. Il en est de même lors de la

sauvegarde du modèle pour lequel on contrôle la présence de toutes les fiches indispensables et de celles auxquelles l'usager a fait référence lors de la modélisation. La modularité inhérente à la modélisation OO laisse envisager la réutilisation partielle des modèles développés en procédant à deux types de regroupements. Le premier divise un modèle en objets transformés (les produits), objets "transformateurs" (le système de production), scénario et mesures effectuées (les instruments, au sens large). On peut parler dans ce dernier cas de découpage fonctionnel. Le second, dit de découpage hiérarchique, sauve des sous-modèles représentant par exemple un atelier dans une usine pour les utiliser comme composants d'autres modèles. L'interface et le gestionnaire de modèles, peuvent, moyennant des transformations minimes, être adaptés à des logiciels pour lesquels le mode de saisie des informations par fiches est retenu. Il convient à ce moment-là de développer un nouveau compilateur pour faire le lien avec l'application (Fabre 1990). En effet, l'interface, la base de modèles et son gestionnaire sont indépendants du simulateur et seul le compilateur connaît leur spécificité respective et assure un lien entre eux. Il est la clé de voûte de cette partie de l'environnement.

Le compilateur

Le compilateur est invoqué avant de commencer toute simulation, il traduit les informations entrées dans les différentes fiches de manière à les rendre compréhensibles par le simulateur. Son travail se décompose en deux étapes de création et d'affectation des objets appartenant aux différents types (charges, moyens de manutention,...) dans un premier temps puis de compilation des suites d'actions conditionnelles qui définissent le processus manufacturier dans un second temps. Cette deuxième étape nécessite la connaissance de tous les éléments statiques qui composent le modèle. Le compilateur se charge en plus dans la deuxième étape de rajouter des conditions et des actions implicites (voir paragraphe 3.5). Ces deux étapes comprennent chacune une phase de détection d'erreurs. La compilation et la simulation du modèle s'enchaînent directement quand aucune erreur n'a été détectée lors de la première étape. Pour l'instant, en raison du niveau d'"expertise" contenu dans le simulateur, notamment pour la génération des conditions et des actions implicites, il n'est pas possible de modifier des paramètres en cours de simulation sans re-compilation. Ce type de modification pourra toutefois être autorisé dans l'avenir uniquement pour les caractéristiques statiques des objets ou pour certaines règles, comme celles d'ordonnancement par exemple.

Le module de contrôle de la simulation

Les classes regroupées dans ce module sont indépendantes du type d'étude que l'on fait et se décomposent de la manière suivante (fig. 4):

- les suites action/conditions (voir paragraphe 3.4);

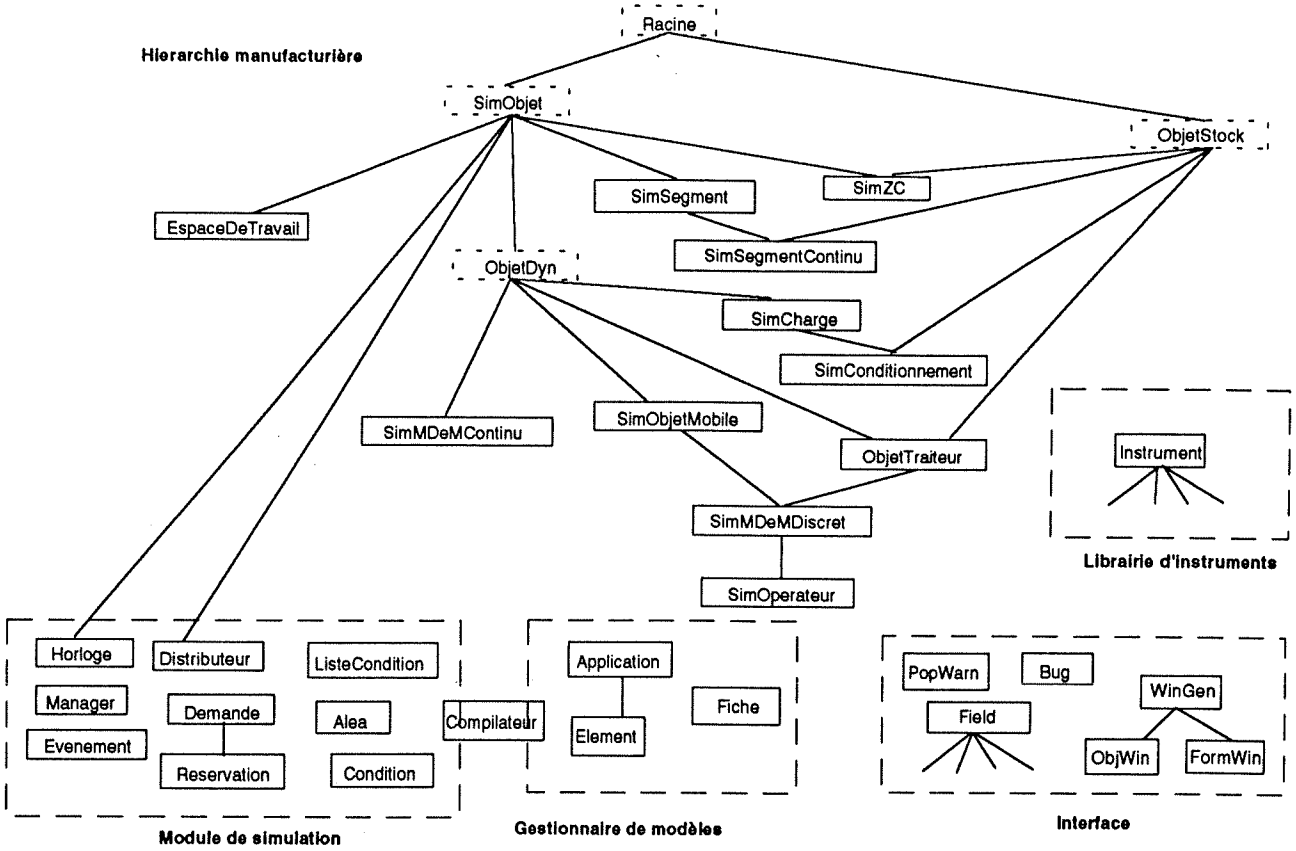

Figure 4: Hiérarchie des classes pour le module de simulation

- le gestionnaire du calendrier d'actions (Manager);
- les mécanismes auxiliaires (Horloge, Distributeur);
- le générateur d'aléa (Alea).

L'objet *manager,* seule instance de la classe Manager, gère le calendrier d'événements. Les événements qu'il reçoit sont classés en fonction tout d'abord de leur horaire d'occurrence et, en cas de conflit, suivant leur type (un ordre de priorité ayant été établi entre les différentes sortes d'événements suivant qu'ils s'appliquent ou non au même objet). L'objet *horloge,* seule instance de la classe Horloge, maintient l'heure courante et l'horaire de fin de simulation; il est aussi chargé de vérifier les conditions portant sur l'horaire (voir paragraphe 3.4). L'objet *distributeur,* également unique instance de la classe du même nom, génère soit des constantes, soit des nombres aléatoires, distribués suivant l'une des huit lois de probabilité qu'il supporte (i. e. normale, uniforme, triangulaire, exponentielle, erlang, gamma, bêta, weibull). Il peut aussi accéder à des fichiers ASCII où est stockées une série de valeurs numériques modélisant un phénomène. Cet objet est en plus, chargé de vérifier les conditions probabilistes (voir paragraphe 3.4).

Les objets de la classe Alea sont au nombre de quatre: *panne, maintenance, maintenanceHS* et *pauseOp.* Ils génèrent, suivant les informations disponibles, des événements qui bloquent le

fonctionnement d'un objet d'un type donné ou de l'ensemble de l'espace de travail. Certains de ces blocages arrêtent les actions en cours (pannes et pauses immédiates) tandis que d'autres sont retardés jusqu'à leur fin normale (maintenances et pauses flexibles).

Les classes de ce module ne connaissent et n'utilisent que les classes SimObjet et ObjetDyn qui sont à la base de toute hiérarchie. Un événement, par exemple, porte toujours sur un objet SimObjet, le mécanisme d'héritage et de définition virtuelle des prototypes de méthodes (ou procédures), propre à la POO permet dans tous les cas d'assurer la connexion avec le bon élément et la bonne méthode.

Les instruments de mesure

Les instruments de mesure, concept introduit avec KBS (Fox & al. 1988), offrent plus de souplesse et de puissance au niveau de la collecte et de l'analyse des résultats que les outils proposés par les simulateurs traditionnels. Ils s'apparentent à leurs homonymes physiques et peuvent être associés facilement à n'importe quel objet, accordant même la possibilité, par exemple, de prendre des mesures précises sur le 37e objet d'un type donné si nécessaire. Les instruments développés jusqu'à présent et qui ont permis de tester le comportement du module de simulation sont de trois types simples:

- suivi des événements ou TRACE (fig. 3);
- instruments de mesures opérationnelles;
- instruments de mesures économiques.

Les instruments peuvent être construits pour répondre à des besoins spécifiques et contenir des informations pour l'analyse experte des résultats de la simulation (proposition de décisions ou de modifications des points litigieux du système modélisé...). S'ils doivent obligatoirement être reliés aux objets dont ils mesurent les paramètres, leur usage n'est pas limité à la seule simulation du modèle, ils restent utilisables dans le cas où les événements ne sont plus générés aléatoirement par le module de simulation mais proviennent du système manufacturier lui-même. Ils permettent alors d'en suivre l'évolution réelle. Leur origine OO (avec notamment les concepts de hiérarchie et d'héritage) favorise la création rapide de nouveaux et l'adaptation à des études différentes sans remettre en cause le noyau de simulation ni son fonctionnement. Au même titre que pour les objets modélisant le système manufacturier et dont il est question au paragraphe suivant, les instruments sont donc organisés en librairies.

3.3 Hiérarchie des classes manufacturières

La figure 4 présente également le noyau d'objets génériques qui servent à décrire un système manufacturier à un premier niveau d'abstraction, tandis que le tableau 3 explique les relations

entre ces objets en des termes généraux, faisant apparaître en caractères gras et italiques les noms des classes.

L'intégration de nouvelles classes destinées à modéliser plus fidèlement encore un système manufacturier particulier est à partir de là possible. On peut ainsi créer des classes plus spécifiques pour différents types de machines (grues, chariots, MOCN...) à partir des classes génériques existantes. Le concept d'héritage propre à la POO favorise ainsi la conception de librairies adaptées à un champ d'application donné (entreposage, distribution, "job shop", services...), modulaires dans leur conception et compatibles entre elles et avec les autres éléments de l'environnement. Du point de vue informatique, il faut noter l'usage qui est fait de l'héritage multiple qui permet à une classe d'hériter son comportement de plusieurs parents. Par exemple, la classe ObjetDyn présente la capacité de traitement d'événements et la classe ObjetStock reprend les caractéristiques de stockage d'objets et toutes deux servent à définir la classe ObjetTraiteur (fig. 4) qui conceptualise les éléments pouvant recevoir des charges et les modifier. Le principe d'héritage multiple simplifie la représentation de nouvelles classes en favorisant la réutilisation des classes génériques.

« Nous cherchons à représenter un ***espace de travail*** dans lequel des ***charges*** sont déplacées à l'aide de ***moyens de manutention*** (ou ***m de m*** en abrégé) "***continus***" (convoyeurs...) et "***discrets***" (chariots...) - par opposition à continus-, et/ou transformées sur des ***postes de travail***. Ces différents matériels peuvent requérir la présence d'***opérateurs*** qui assurent également des tâches de manutention ou de transformation. L'espace de travail peut aussi être traversé par des ***objets mobiles*** (externes) et qui ne s'y arrêtent pas. Les charges peuvent être regroupées sur (ou dans) un même ***conditionnement*** (palette, boite..). Le chemin qu'emprunte un moyen de manutention, un opérateur ou un objet mobile dans l'espace de travail est décomposé en ***segments*** (***continus*** et ***discrets***) modélisant des portions de trajectoires et qui s'articulent aux ***zones de contrôles*** (***ZC*** en abrégé). Des ***instruments*** de mesure nous permettent aussi d'étudier l'espace de travail et ses composants au cours de leur fonctionnement ».

Tableau 3: Mise en situation des classes de la hiérarchie manufacturière

3.4 Types d'actions et modélisation du comportement

Du point de vue dynamique, l'usager peut définir 10 types d'actions (classe Evenement) et 8 types de conditions (classe Condition) qui sont présentées dans les tableaux 4 et 5. Le module de simulation gère 5 autres types d'actions (dont deux pour l'animation). Les actions que doivent effectuer les objets et les listes de conditions (classe listeCondition) qui leur sont associées sont cristallisées sur les zones de contrôles (ou ZC, c'est à dire les objets de la classe SimZC). Les ZC jouent en effet le rôle de noeud logique et de point de décision dans le système et servent en même temps à matérialiser, quand cela est nécessaire, les espaces d'entreposage. Les actions sont pour l'instant stockées au moment de la compilation, mais pourront l'être dynamiquement par la suite pour tenir compte d'événements imprévus et de routes alternatives par exemple.

Les actions peuvent être soit immédiates, soit conditionnelles. Les objets sont avertis des

actions qu'ils doivent accomplir à leur arrivée sur la ZC (fig. 5).

Cette dernière se charge de contacter les objets qui doivent vérifier une condition. Une action est exécutable chaque fois que toutes ses conditions sont vraies. L'objet insère alors un événement dans le calendrier à la date où finit l'action puis modifie ses paramètres en conséquence. Il annule ensuite les demandes que la ZC avait effectuées en son nom et lui indique de demander la vérification des conditions pour la prochaine action à effectuer et ainsi de suite. Les actions (conditions) peuvent être séquentielles, séparées par le mot de liaison ET, ou alternatives, reliées par un OU. Elles sont évaluées dans l'ordre où elles ont été programmées et dans le cas d'actions alternatives simultanément réalisables (c'est à dire dont les conditions de réalisation sont simultanément vraies) priorité est donnée à la première codée. Les demandes effectuées auprès d'un objet peuvent être de deux types: informatives (Demande) ou consommatrices (Réservation).

Actions *Modificateurs*
APPARAIT *date_première_apparition fréquence*
DISPARAIT *fréquence (rien par défaut)*
CHARGE *poste_de_travail*
PORTE *conditionnement_ou_MdeM_ou_operateur*
DEPOSE
ASSEMBLE *objet_créé {composant(quantité)}*
DEMOLIT *{objet_produit(quantité)}*
MODIFIE *durée_modif {paramètre absolu valeur}*
ATTEND *durée_attente*
PART *tronçon_emprunté*
STOPPE
BLOQUE *objet_bloqué*
DEPLACE *x y z*
DESSINE *date*
ARRIVE *ZC*

Tableau 4: Types d'actions

Conditions *Modificateurs*
HORAIRE *quantificateur date_visée*
PROBABILITE *quantif. valeur_visée*
BLOCAGE *nom_de_la_zone_de_contrôle*
ESPACE *nom_de_la_ZC nom_du_tronçon*
(1) PRESENCE *lieu objet_visé quantificateur quantité*
(2) PARAMETRE (1) + *para. quantif._para. valeur_para.*
RSRV+PRESENCE (1) + *portée_de_la_réservation*
RSRV+PARAMETRE (2) + *portée_de_la_réservation*

Tableau 5: Types de conditions

Les premières ne modifient pas l'objet auquel elles sont adressées, elles ne s'intéressent qu'à son état. Les secondes, quand elles sont vérifiées, notifient à l'objet qu'il est réservé et ne peut accepter d'autre réservation avant d'avoir satisfait celui qui l'a réservé. Ce type de demande est utilisé par exemple pour qu'un moyen de manutention discret ou un opérateur vienne se saisir d'une charge pour la transporter à sa destination. Ainsi, les transporteurs sont en quelque sorte "guidés" vers leur destination par les objets qu'ils transportent ou par ceux qu'ils viennent chercher.

Pour illustrer par un exemple le processus de communication entre les éléments d'un modèle, considérons par exemple le déplacement de la charge sombre *Objet dynamique* de la figure 5. Dès qu'elle reçoit notification de son départ de la première zone de contrôle (1), l'événement ARRIVE est inséré dans le gestionnaire d'événements à sa date d'occurrence. Quand vient le tour de cet événement, la nouvelle zone de contrôle (ZC d'arrivée) est avertie de l'arrivée de ce nouvel

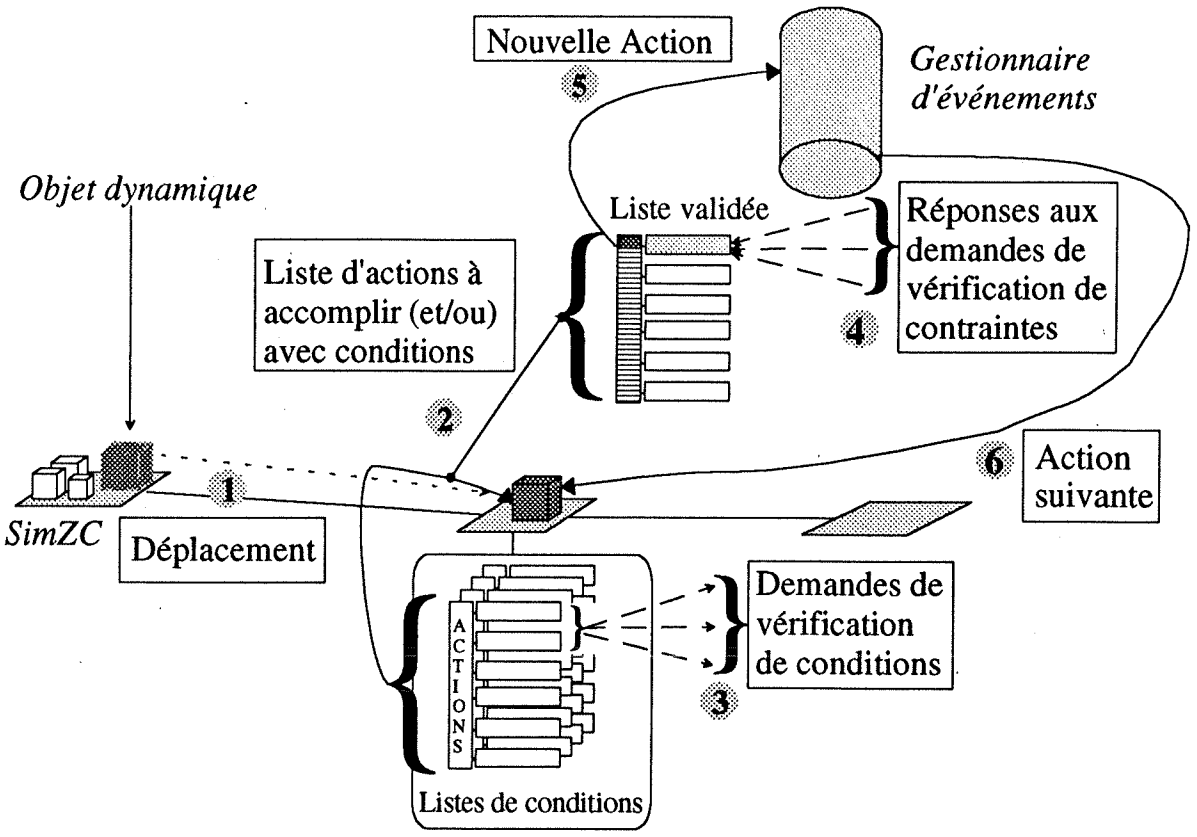

Figure 5: Communication entre objets

objet. Aussitôt après avoir ajusté ses compteurs et ses listes, elle communique au nouvel arrivant la liste des actions conditionnelles qu'il devra accomplir durant son séjour (2). Elle envoie ensuite en son nom les demandes de vérification de conditions concernant la (ou les) première(s) action(s) exécutable(s) (3). Ceci fait, elle vérifie à son tour l'influence de l'arrivée de cet objet sur les conditions qu'elle-même doit vérifier. Dès que l'objet a reçu notification de la validité des conditions pour une action exécutable (4), les demandes de vérifications concernant cette action et les actions alternatives sont annulées. L'action validée est alors insérée à sa date d'occurence dans le gestionnaire d'événements (5) avant d'être renvoyé à l'objet en temps utile (6). Et le processus reprend ensuite à l'étape (3) jusqu'à ce que l'objet n'ait plus d'action à accomplir.

3.5 Principales caractéristiques

En termes de lignes de code, ce logiciel représente, en l'état actuel, un ensemble d'une quarantaine de classes et d'environ 30000 lignes écrites en C++. La rapidité du simulateur est fonction du nombre de messages échangés entre les objets du modèle et de la lourdeur des structures manipulées (les objets). Pour l'instant, sur les modèles étudiés, le module de simulation traite en moyenne entre 20 et 50 événements à la seconde (mesure sommaire), suivant si l'on

demande d'avoir la "trace écrite" (dont un exemple est présenté à la figure 3) des événements à mesure qu'ils se produisent, ce qui est pénalisant pour la vitesse du simulateur. Dans l'état actuel, il tient compte de 3 règles de fin de quart de travail, de 40 paramètres pour les actions et les conditions qui peuvent faire référence à l'objet demandeur lui-même ou à son porteur et être limitées à une ZC ou étendu à l'espace de travail.

L'apparition de lots de nouveaux objets peut être unique, conditionnelle ou relié à une fréquence. Les transporteurs (i.e. moyens de manutention discrets ou opérateurs) disposent de 7 règles de choix (FCFS, LCLS, date de livraison, priorité par type, priorité par ZC, plus loin, plus près) pour déterminer le prochain objet à transporter ou à déposer. Leurs déplacements à vide se font en suivant le plus court chemin disponible entre la ZC courante et la destination. Leur fonctionnement, tout comme celui des postes de travail, est conditionné par la présence des pilotes qui sont libérés une fois la tâche accomplie ou lors de l'occurrence d'un aléa. Dans ce dernier cas, des réparateurs sont appelés puis dissociés (implicitement) une fois la réparation terminée. Il est finalement possible de définir des temps de configuration (setup) entre des charges de type différent ou identique.

4. Directions d'évolution

Outre l'amélioration de l'interface usager, l'évolution du module de simulation dans le cadre d'ASMÉMA se poursuit vers l'intégration avec le module de conception graphique d'une part (modélisation et animation) et vers la création de nouveaux instruments de mesures économiques facilitant la comptabilité basée sur les activités (ou ABC) et tenant compte également de facteurs intangibles comme la qualité ou la flexibilité, comme c'est le cas dans les nouvelles méthodes de comptabilité (Arbel & Seidman 1984, Azzone & Bertele 1989, Park & Son 1988). D'autre part, un environnement de simulation intégrant les composants de ce module et destiné à l'analyse et au contrôle en temps réel d'un système de production discret s'inscrivant dans la ligne des simulateurs de nouvelle génération tels que décrits par Norman (1992) est actuellement en cours de réalisation.

Bibliographie

Adiga, S., Glassey, R., 1991, Object Oriented Simulation to support research in manufacturing systems, *International Journal of Production Research*, 29(12), pp 2529-2542

Arbel, A., Seidman, A., 1984, Performance evaluation of FMS, *IEEE Transactions on systems, man and cybernetics,* 14, (July - August 84), pp 118-129

Azzone, G., Bertele, U., 1989, Measuring the economic effectiveness of flexible automation: a new approach, *International Journal of Production Research,* 27(5), pp 735-746

Barkakati, N., *X Window System™ Programming,* SAMS, Macmillan Computer Publishing, Carmel, In, 1991

Beaumariage, T. E., Roberts, C. A., 1991, Object Oriented modeling: attempts at improving model execution speed, paru dans *Simulation series,* 23(3), Raimund Ege editor, pp 93-99

Choi, S., Minoura, T., 1991, Active Object System for discrete system simulation, paru dans *Simulation series,* 23(3), Raimund Ege éditeur, pp 209-215

Elderedge, D. L., Mc Gregor, J. D., Summers M. K., 1990, Applying the object oriented paradigm to discrete event simulation using the C++ language, *Simulation,* February 90, pp 83-91

Fabre, M., 1990, Une interface orientée objet pour la construction modulaire de modèles en simulation manufacturière, mémoire de M. Sc. A., Ecole Polytechnique de Montréal

Fabre, M., Pitre, J.C., Levasseur, M., Gilbert, R., Leblanc, D., Normandin, M., 1992, The decision of non-automation in manned workstation, *Rapport technique EPM-RT 92/36*, présenté à POMS 3rd annual meeting, Oct. 18-21, Orlando, Fl

Fabre, M., Leblanc, D., 1993, Towards an object-oriented simulation environment, *proceedings of the 26th annual simulation symposium,* IEEE, SCS, Washington, pp 247-256

Fox, M. S., Husain, N., Mc Roberts, M., Reddy, Y. V. R., 1988, Knowledge based simulation: an artificial intelligence approach to system modeling and automating the simulation life cycle, paru dans *AI, Simulation and modeling*, L. E. Widman, K. A. Loparo, N. R. Nielsen éditeurs, John Wiley, pp 447-486

Fujimoto, R., 1990, Parallel discrete event simulation, *Communication of the ACM,* 33(10), pp 31-53

METIS, 1992, Rapport de présentation du projet et du logiciel ASMEMA, *rapport technique*, Ecole Polytechnique de Montréal

Mize, J. H., Bhuskute, H. C., Pratt, D. B., Kamath, M., 1992, Modeling of Integrated manufacturing systems using an object oriented approach, *IIE Transactions*, 24(3), pp 14-25

NIOSH, 1981, *Work practices guide for manual lifting*, U.S. Department of Commerce, (National Technical Information Service)

Norman, V. B., 1992, Future directions in manufacturing simulation, *Industrial Engineering,* 24(7), pp 36-37

OSF, OSF/Motif™ programmer's reference (rev. 1.1), Prentice Hall, Englewood Cliffs, NJ, 1991

Park, C. S., Son, Y. K., 1988, An economic evaluation model for advanced manufacturing systems, *Engineering Economist,* 34(1), Fall 88, pp 1-26

Roberts, S. D., Heim J., 1988, A perspective on object oriented simulation, *proceedings of the 1988 winter simulation conference*, SCS, pp 277-281

Wiener, R. S., Pinson, L. J., 1988, *Introduction to object-oriented programming and C++,* Addison-Wesley Publishing Co., Reading, Ma

WSC, 1990, *proceedings of the 1990 Winter Simulation Conference*, IEEE, O. Balci, R. P. Sadowski & R. E. Nance éditeurs