Composants pour la modélisation des processus métier en productique, basés sur CIMOSA

Anis Abdmouleh

THESE

Présentée par

ABDMOULEH Anis

Pour obtenir le titre de

DOCTEUR DE L'UNIVERSITE DE METZ
(arrêté ministériel du 30 Mars 1992)

Spécialité : AUTOMATIQUE / PRODUCTIQUE

Ecole doctorale Informatique, Automatique, Électronique - Électrotechnique, Mathématiques (IAEM)

Composants pour la modélisation des processus métier en Productique, basés sur CIMOSA

Date de soutenance : 15 Septembre 2004

Composition du jury :

Président : Gérard MOREL, *Professeur à l'Université Henri Poincaré de Nancy I*

Rapporteurs : Jean-Pierre BOUREY, *Professeur à l'Ecole Centrale de Lille*
Laurent FOULLOY, *Professeur à l'Ecole Supérieure d'Ingénieurs d'Annecy*

Examinateurs : Pierre PADILLA, *Professeur à l'Ecole Nationale d'Ingénieurs de Metz*
Michel SPADONI, *Maître de Conférences à l'Ecole Nationale d'Ingénieurs de Metz* (Co-encadrant)
François VERNADAT, *Professeur à l'Université de METZ* (Directeur de thèse)

Thèse préparée au sein du Laboratoire de Génie Industriel et de Production Mécanique
- Ecole Nationale d'Ingénieurs de Metz & Université de METZ -

THESE

Présentée par

ABDMOULEH Anis

Pour obtenir le titre de

DOCTEUR DE L'UNIVERSITE DE METZ
(arrêté ministériel du 30 Mars 1992)

Spécialité : AUTOMATIQUE / PRODUCTIQUE

Ecole doctorale Informatique, Automatique, Électronique - Électrotechnique, Mathématiques (IAEM)

Composants pour la modélisation des processus métier en Productique, basés sur CIMOSA

Date de soutenance : 15 Septembre 2004

Composition du jury :

Président : Gérard MOREL, *Professeur à l'Université Henri Poincaré de Nancy I*

Rapporteurs : Jean-Pierre BOUREY, *Professeur à l'Ecole Centrale de Lille*
Laurent FOULLOY, *Professeur à l'Ecole Supérieure d'Ingénieurs d'Annecy*

Examinateurs : Pierre PADILLA, *Professeur à l'Ecole Nationale d'Ingénieurs de Metz*
Michel SPADONI, *Maître de Conférences à l'Ecole Nationale d'Ingénieurs de Metz* (Co-encadrant)
François VERNADAT, *Professeur à l'Université de METZ* (Directeur de thèse)

Thèse préparée au sein du Laboratoire de Génie Industriel et de Production Mécanique
- Ecole Nationale d'Ingénieurs de Metz & Université de METZ -

À mes parents qui m'ont encouragé à suivre cette thèse,

À mon épouse qui m'a soutenu et encouragé tout au long des travaux,

À mon fils qui me redonne courage grâce à ses beaux sourires,

Et au futur nouveau né.

REMERCIEMENTS

Ce travail a été réalisé au sein du laboratoire Génie Industriel et Production Mécanique de Metz (LGIPM), sous la direction du Professeur François VERNADAT et de Monsieur Michel SPADONI Maître de conférences à l'Ecole Nationale d'Ingénieurs de Metz (ENIM). Que je remercie vivement pour l'acceuil, l'encadrement, le soutien et les précieux conseils dont j'ai bénéficié tout au long de ce travail.

Toute ma gratitude va ensuite aux membres du jury de ma thèse. Je remercie ainsi :

- Monsieur Pierre PADILLA, Professeur et directeur de l'Ecole Nationale d'Ingénieurs de Metz,
- Monsieur Gérard MOREL, Professeur à l'Université Henri Poincaré de Nancy,
- Monsieur Jean-Pierre BOUREY, Professeur à l'Ecole Centrale de Lille,
- Monsieur Laurent FOULLOY, Professeur et Directeur de l'Ecole Supérieure d'Ingénieurs de Annecy.

Je suis très reconnaissant à Monsieur BOUREY et à Monsieur FOULLOY pour l'honneur qu'ils me font en acceptant d'examiner ce travail et d'être les rapporteurs de cette thèse. Leurs remarques et conseils m'ont permis d'améliorer sensiblement la qualité du mémoire.

Je tiens également à adresser mes remerciements à tous les membres de l'équipe AGIP, le personnel de l'ENIM et de l'Université de METZ pour la bonne ambiance et la convivialité qu'ils ont su maintenir, ainsi que leur encouragement et leur amitié.

Je remercie également :

- Monsieur Gabriel ABBA, Professeur et Directeur du LGIPM,
- Monsieur Xiaolan Xie , Professeur à l'Université de METZ et Responsable de l'équipe AGIP,
- Nidhal REZG, Professeur à l'Université de METZ, équipe AGIP,

- Catherine JUNG, Secrétaire au LGIPM, pour son aide précieuse tout au long de la thèse,
- Brigitte FINEL, Ingénieur-Chercheur au LGIPM, pour son aide et sa générosité,
- Etc.

Mes remerciements vont aussi à tous ceux qui m'on aidé durant la thèse et dont je n'ai pas cité le nom.

TABLE DE MATIERES

TABLE DE FIGURES

TABLE DE TABLEAUX

INTRODUCTION

INTRODUCTION

CADRE D'ETUDE

La notion d'entreprise, telle que nous l'appréhendons dans cette thèse, se réfère à un ensemble d'activités mises en œuvre par un ensemble de moyens dans le cadre d'une finalité définie par un ou plusieurs objectifs. C'est le cas des entreprises industrielles de production de biens ou de services, les hôpitaux ou les établissements de santé, les établissements de formation ou de recherche.

Aujourd'hui, beaucoup d'entreprises font le constat qu'être performantes ne suffit plus pour assurer leur pérennité. En effet, les exigences des clients sont si multiples et les intervenants dans la chaîne (chaîne d'entreprises) si nombreux, que l'entreprise se trouve en situation d'interdépendance très forte avec ses partenaires. C'est la raison pour laquelle l'enjeu futur ne concerne plus la compétitivité au sein de l'entreprise mais dans les chaînes d'entreprises. C'est ainsi que la modélisation en entreprise peut dépasser le cadre d'une seule entreprise pour s'étendre à un réseau complet d'entreprises.

Le premier pas pour améliorer l'efficacité opérationnelle repose sur l'ingénierie intégrée de l'ensemble d'une chaîne d'entreprises en une seule application capable d'appréhender les divers aspects d'une telle chaîne. Il ne s'agit pas de modéliser l'entreprise dans sa globalité ni à un niveau de détail excessif ; il s'agit de comprendre la structure et le fonctionnement de la partie de l'entreprise fixée comme objet d'étude dans un souci d'analyse, de conception et d'évaluation de performance. De plus, les résultats issus de la modélisation en entreprise aident à la capitalisation des connaissances sur l'entreprise réseau par l'élaboration de modèles. En effet, ces modèles permettent de représenter les aspects essentiels d'une entreprise dans un formalisme unifié et compréhensible par les différents acteurs de cette chaîne d'entreprises, comme c'est notamment le cas avec UML (Unified Modeling Language) qui est appliqué principalement pour la conception des systèmes d'information.

L'élaboration d'un modèle est une occasion de bâtir un consensus ou une vision commune des objectifs, de la structure et du fonctionnement de toute ou partie de l'entreprise. Cette vision peut être ensuite partagée et exploitée par les différents acteurs de la chaîne (clients, consommateurs, sous-traitants, fournisseurs, etc.).

NOTRE CONTRIBUTION

Notre objectif est de contribuer à l'élaboration et à l'agencement de modèles orientés processus métier en productique, à partir des objectifs de l'entreprise. Ceci afin d'aider l'entreprise à optimiser son fonctionnement et à capitaliser une partie de son savoir-faire sous forme de modèles réutilisables et partageables par les différents acteurs de la chaîne d'entreprises. Cette approche, bien que développée dans le cadre de la productique, nous parait applicable à tout type d'entreprise.

Après comparaison des approches existantes de modélisation en entreprise (CEN-ENV 40003, CIMOSA, GRAI-GIM, GERAM, PERA, ARIS, IDEF, BPR et autres), nous avons retenu les architectures qui permettent de modéliser les fonctions d'entreprise. Ces dernières sont modélisées par le biais de processus, qui permettent une organisation transversale de l'entreprise. En effet, nous rappelons que notre préoccupation première est de prendre en compte les aspects fonctionnels et comportementaux de toute ou partie d'une entreprise ou d'un réseau d'entreprises.

Parmi l'ensemble de ces architectures, CIMOSA a introduit la notion de modélisation et de pilotage des systèmes de production par processus métier (business processes) [AMICE, 93] et non par activité, comme ce fut le cas des méthodes des années 80 (GRAI [Roboam, 93], IDEF, IPAD, ...). CIMOSA offre un concept permettant l'organisation des entreprises autour de leurs processus métier (modèles orientés processus) [Vernadat, 96]. Il s'agit d'un changement radical d'organisation des systèmes de production visant à abolir les barrières organisationnelles ayant conduit à la création d'îlots d'automatisation et formant un frein au développement de systèmes intégrés [Davenport, 93] [Hammer, 93] [Harrington, 91].

Dans le cadre de nos travaux de thèse, nous proposons une approche par composants logiciels pour l'aide à la modélisation et la conception des systèmes en entreprise dans le but de permettre la « distribution » des modèles de systèmes. Cette aide se concrétise sous forme d'une méthode qui a été élaborée en se basant sur l'approche CIMOSA. Nos travaux portent sur deux parties principales :

- Une première partie concerne l'identification et la définition des concepts[1] nécessaires à la modélisation d'un système incluant un référentiel de concepts.
- Une deuxième partie propose une méthodologie d'utilisation des concepts pour permettre le développement de modèles de processus d'un système.

L'implémentation du référentiel est réalisée sur la base d'une *architecture distribuée de type client/serveur*, mettant en œuvre des *composants réutilisables* et *partageables* par les modélisateurs et les concepteurs.

Un système peut être défini comme un ensemble de méthodes et de procédés[2] destinés à assurer une fonction définie ou à produire un résultat [Larsen, 99]. Pour cela, il est important d'en définir son objectif et ses concepts.

Les concepts de base d'un système d'entreprise sont les entités de l'entreprise, les processus et les rôles des acteurs [Eriksson, 00].

En fait, le flux de contrôle du système est décrit par les processus qui transforment des entrées en sorties par le biais de ressources.

Ces processus sont appelés des processus opérationnels ou encore processus métier (*business processes*) [Vernadat, 96]. Ils permettent de décrire le comportement du système.

Dans la thèse, nous proposons un méta-modèle (modèle de référence) regroupant plusieurs concepts de CIMOSA ainsi que quelques patrons métier (*business patterns*). Ce méta-modèle intègre ainsi le savoir-faire de méta-modélisation (CIMOSA) et de conception métier (patrons métier).

Le méta-modèle est réalisé suivant une approche orientée objet et représenté au moyen du langage objet unifié: UML (*Unified Modeling Language*) [Roques, 00] [Booch, 00] [Rumbaugh, 94]. Ceci permet d'améliorer la communication entre les modélisateurs et les concepteurs pour faciliter des travaux ultérieurs de conception et de développement informatique.

---

[1] Concept : représentation générale et abstraite d'un objet ou d'un ensemble d'objets. Définition des caractères spécifiques d'un projet, d'un produit par rapport à l'objectif ciblé [Larsen, 99].
[2] Procédé : moyen, méthode pratique pour faire quelque chose, pour obtenir un résultat [Larsen , 99].

Dans ce méta-modèle, nous avons retenu l'approche par vues de CIMOSA pour répartir les différents concepts. Ces derniers sont présentés sous forme d'une ou plusieurs classes d'objets regroupées dans des paquetages en UML.

Ce méta-modèle décrit donc la structure du Référentiel des concepts cités ci-dessus.

Du fait que ce méta-modèle se base sur le concept de 'processus métier', nous avons développé une méthodologie permettant la préparation d'une cartographie des processus d'un système d'entreprise.

Les paquetages sont implémentés au moyen de cinq composants (Gestion & Conception de Processus, Comportement, Gestion d'informations, Gestion de Ressources et Structure d'Organisation). Ils intègrent la méthodologie à suivre pour développer un modèle d'un système par le biais de leurs interfaces correspondantes.

Ces composants jouent le rôle de serveurs métier au sein d'une architecture distribuée client/serveur, laquelle supporte le référentiel que nous nommons Référentiel de Composants de Modélisation en entreprise ou E.M.C (*Enterprise Modeling Components*).

PLAN DE LECTURE DE LA THÈSE

Après cette introduction générale, le chapitre I donne un état de l'art du sujet et explique l'importance de l'approche par processus dans l'entreprise ainsi que le rôle qu'elle peut jouer lors d'une organisation ou une réorganisation. Ensuite, nous parlons de l'intérêt de la modélisation des processus. Enfin, nous dénombrons les avantages de la distribution des modèles de processus.

Dans le chapitre II, nous donnons des définitions de processus et nous présentons des typologies. Ensuite, nous présentons les architectures, les méthodes et les techniques de modélisation des processus d'entreprise, en se focalisant sur la modélisation des processus métier dans l'approche CIMOSA.

Après les deux premiers chapitres qui donnent le contexte global du sujet de notre thèse, le chapitre III positionne plus précisément notre travail dans le projet CAS qui vient de voir le jour lors du lancement de cette thèse.

Dans le chapitre IV, nous introduisons le référentiel de modélisation de processus que nous proposons en se basant sur l'approche CIMOSA. Ensuite, nous proposons un méta-modèle qui se base sur cette approche. Ce méta-modèle est utilisé par les acteurs métier de

l'entreprise pour développer des modèles particuliers à l'entreprise. Cette utilisation se fait par le biais de composants métier. Ces composants sont :

- Gestion et Conception de Processus,
- Comportement,
- Gestion d'information,
- Gestion de ressources,
- Structure d'Organisation.

Pour que les acteurs puissent coopérer lors de développement de modèles, les composants doivent fournir et manipuler des documents en un format neutre permettant l'intégration en entreprise et l'interopérabilité avec d'autres applications.

D'autre part, nous avons proposé une solution pour ces deux problèmes au chapitre V. En effet, nous avons proposé d'utiliser le format XML pour fournir des documents XML. Ces documents stockent des informations à propos des systèmes à modéliser. Ces documents sont fournis par un framework basé sur les composants métier proposés au chapitre IV. Enfin, nous avons validé le fonctionnement du framework par un exemple académique.

Au point du vue développement informatique, nous nous sommes limités dans cette thèse à développer la couche métier des composants proposés et une partie de l'interface graphique du framework correspondant au composant « Gestion et Conception de Processus ».

Enfin, nous terminons par une conclusion générale et nous présentons nos perspectives.

Nous avons joint à ce rapport trois annexes permettant la clarification et l'approfondissement de quelques points traités au cours de la thèse. Ces trois annexes traitent respectivement les règles de comportement de CIMOSA, les patrons métier et le langage unifié UML.

CHAPITRE I

DISTRIBUTION DE MODELES DE PROCESSUS EN ENTREPRISE

CHAPITRE I : DISTRIBUTION DE MODELES DE PROCESSUS EN ENTREPRISE

I.1. INTRODUCTION

Le contexte économique actuel, dominé par une concurrence exacerbée où la satisfaction du client domine, amène les entreprises à gérer directement et essentiellement les processus et les activités pour réduire leur coût et augmenter leur performance. En effet, cela leur permet une gestion plus rapide et surtout plus flexible en fonction des performances à atteindre, donc plus apte à s'adapter à la satisfaction du client et par voie de conséquence à la concurrence.

Dès lors qu'une entreprise évolue dans un cadre concurrentiel, son objectif est le même, c'est à dire améliorer de façon significative la productivité et par là sa compétitivité. Aujourd'hui, de par les technologies existantes, un des moyens d'atteindre cet objectif est de migrer en partie ou en totalité le système d'information de l'entreprise vers un système intégré (ex : ERP).

A cause de la taille des entreprises, les raisons qui expliquent ce besoin de migrer sont paradoxalement opposées. En effet, les grandes entreprises souffrent d'un parc applicatif existant important et ne peuvent le faire évoluer en temps voulu et en un coût acceptable que difficilement et avec un investissement important. A contrario, les petites entreprises souffrent d'un déficit en système d'information, elles ne disposent pas des ressources et des infrastructures requises pour développer elles-mêmes leurs propres solutions.

Dans ce contexte, il devient nécessaire de poser le problème de migration des applications organisées à partir de bases de données vers des architectures de type client/serveur ou client web ou même par web services [Tomas, 99]. Cette approche a permis de passer de l'ère de la gestion de bases de données à l'ère de la gestion de l'information, elle-même précurseur de l'ère de la connaissance. Dès lors la notion de **distribution** [Ezran, 99] d'information est capitale. Cette distribution pose les problèmes de communication, d'hétérogénéité, d'intégration et d'interopérabilité [Dogac, 98].

En fonction des besoins de l'entreprise, l'existence et la performance des solutions applicatives posent le problème de décider lesquelles doivent provenir de l'extérieur et lesquelles de l'intérieur.

Pour les applications comme pour les services, le critère de choix des activités à externaliser touche ce qui donne un avantage concurrentiel. Par contre, pour ce qui concerne les aspects

qui impliquent des activités telles que le pilotage, l'architecture ou le décisionnel, il est important d'en conserver la maîtrise.

Dans le cadre de cette problématique et de par la progression et la place des processus dans l'entreprise, nous préconisons d'externaliser des modèles de processus sans se préoccuper des architectures.

Des enquêtes ont été effectuées qui montrent le besoin pour les entreprises d'améliorer leurs processus. Nous citons en particulier une enquête qui a été réalisée par le cabinet de Benchmark Group [JDNet, 03a]. Celle-ci montre que 34% des entreprises essaient de modéliser leurs processus métier avec l'hypothèse qu'une entreprise est décomposable en processus, eux-mêmes décomposables, par niveaux successifs, en sous-processus. A cela, nous pouvons ajouter, que l'entreprise elle-même fonctionne comme un processus en interaction avec ses clients, ses fournisseurs, ses concurrents, etc. Ce processus intervenant à son tour dans des processus socio-économiques plus globaux.

Par une dynamique permanente d'amélioration, l'entreprise peut espérer à terme améliorer tous les processus, il faut dans un premier temps s'intéresser à quelques processus et de préférence à ceux dont l'amélioration apportera le plus de gain pour l'entreprise. Ce choix de processus « prioritaires » constitue une phase importante lors de la modélisation et ne doit pas être fait au hasard.

L'expérience a montré que bien souvent l'observation de dysfonctionnements est à l'origine du déclenchement d'une modification de processus. Lorsque ce type de modification est réalisé au coup par coup, il y a fort à parier que l'optimisation du processus sera obtenue après une série de tâtonnements coûteux et dans bien des cas démobilisateurs. Afin d'éviter cette situation, il apparaît nécessaire de définir et regrouper les processus dans une cartographie de façon à identifier les processus principaux et secondaires en se basant sur une typologie qui convient à l'entreprise.

I.2. BESOIN D'UNE APPROCHE PAR PROCESSUS

Une organisation hiérarchique (Figure I.2-1) au sein d'une entreprise est indispensable, car elle permet de savoir « qui fait quoi » (déclinaison d'objectifs dans les niveaux hiérarchiques) et remontée d'indicateurs de performance. Mais celle-ci, seule, ne permet pas à l'entreprise d'être dirigée et pilotée efficacement. Elle devra être complétée par une organisation horizontale (Figure I.2-2) intégrant la vision client/fournisseur. Cette organisation s'appuiera essentiellement sur la mise en œuvre d'une démarche processus, démarche transverse

focalisée plus particulièrement, d'une part, sur les bénéficiaires « pour qui » le travail est réalisé) et d'autre part, sur les processus de réalisation (correspondant à la valeur ajoutée de l'entreprise).

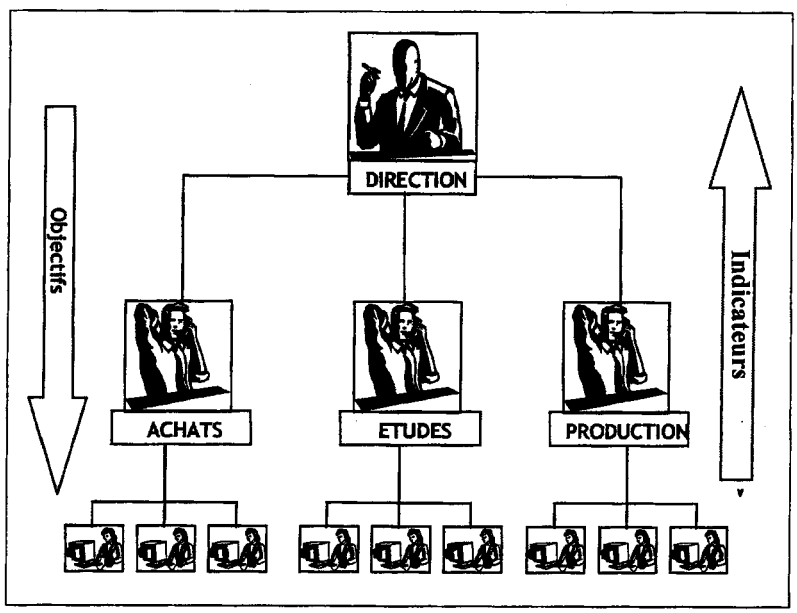

Figure I.2-1- Vision hiérarchique d'une organisation dans l'entreprise (d'après [Frecher, 03])

Cette double vision, hiérarchique/transverse (que l'on peut dénommer verticale/horizontale), est vitale pour l'entreprise. L'organisation hiérarchique permet de « diriger » l'entreprise, la démarche processus liée à l'approche transversale permet de la « manager ». Dans le cadre de la vision horizontale, l'entreprise est alors considérée comme une entité unique prenant en compte, en entrée, les besoins de ses clients et fournissant, en sortie, des produits ou services satisfaisant ces besoins. Cette satisfaction des besoins des clients (bénéficiaires externes) correspond à la valeur ajoutée apportée par l'entreprise.

Toute satisfaction de clients est réalisée au travers de processus internes, transverses à l'organisme, tels que: « concevoir et développer un produit », «vendre un produit », «assurer le service après-vente », etc. Il se peut que certains de ces processus sortent des frontières de l'entreprise (travail en sous-traitance ou en partenariat) ou d'inclure des activités externes à l'entreprise. On parle alors de processus inter-entreprises (entreprise réseau).

Ces processus sont transverses, car les activités qui y sont rattachées sont effectuées par tout ou partie des structures verticales composant l'entreprise. Ainsi, pour le processus «Concevoir

et réaliser un produit », le service achats sera concerné par la fourniture de matières premières, le service études par la conception du produit et le service production par sa réalisation.

L'avantage de cette vision transverse réside dans le fait que l'entreprise (respectivement, le réseau d'entreprise), dans sa totalité, se focalise sur ses clients (respectivement, sur le réseau de clients), en prenant en compte leurs exigences et en recherchant leur satisfaction maximale en fonction des objectifs de l'entreprise (respectivement, du réseau d'entreprises). La création de valeur ajoutée par l'entreprise est apportée par l'ensemble des processus métier qui sont tous axés vers la satisfaction des clients externes (Figure I.2-2).

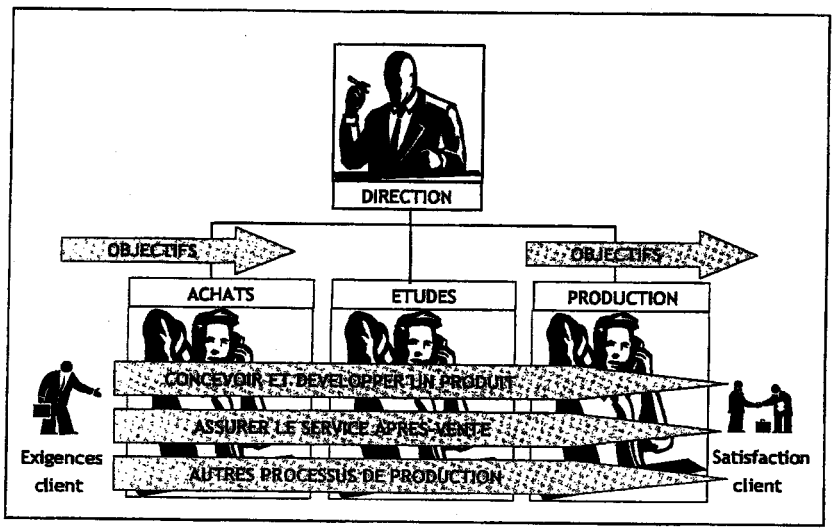

Figure I.2-2- Vision transversale ou horizontale dans une entreprise (d'après [Frecher, 03])

Ces deux visions d'entreprise permettent au manager de définir une stratégie, puis d'en déduire de grands objectifs qu'il assigne à ses subordonnés. Ces derniers les déclineront en stratégies d'actions. Pour que ces stratégies d'actions soient cohérentes pour le client, il s'agira d'identifier les processus qui permettront de réaliser les objectifs fixés (Figure I.2-3).

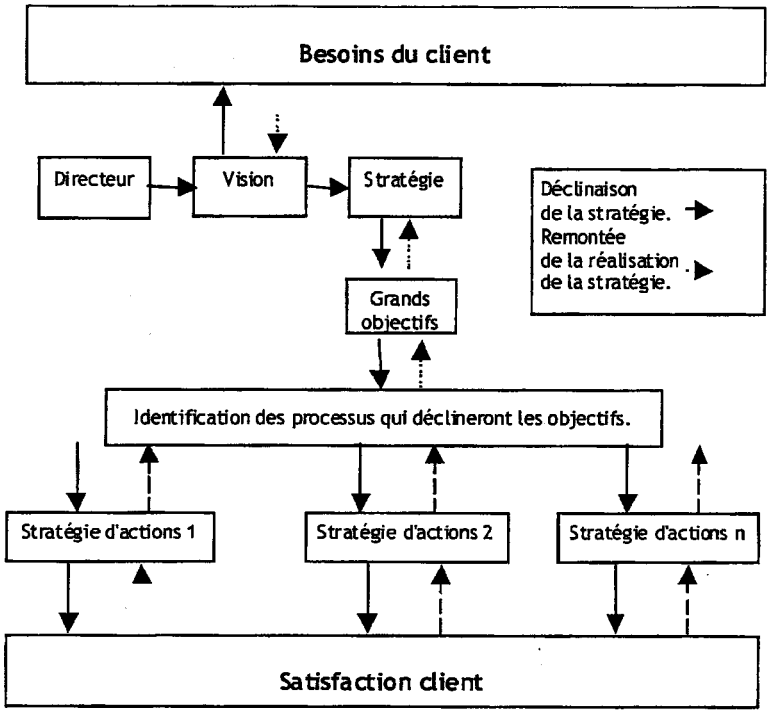

Figure I.2-3- Les objectifs et les processus (d'après [Frecher,03])

I.3. INTERET D'UN MODELE DE PROCESSUS

Dans ce contexte et depuis des dizaines d'années, l'entreprise a tout intérêt à étudier ses processus et à les manager. Mais, plusieurs dysfonctionnements existent encore, et surtout pour définir les interfaces de communication entre les processus de l'entreprise [Martin, 03]. C'est la raison pour laquelle la nouvelle version de la norme ISO (famille 9000) insiste sur la mise en évidence des interfaces entre les processus. C'est pour cela également que plusieurs entreprises essaient de fusionner leurs processus et surtout les processus métier en les intégrant entièrement dans l'entreprise [Gartner, 03]. Les prémices de cette fusion sont perceptibles depuis de très nombreuses années. Tout d'abord, via l'automatisation des processus métier internes, qui a débouché sur les progiciels de gestion intégrée (ERP : '*Enterprise Resource Planning*'). Ensuite, par le biais de la gestion de la relation client (CRM : '*Customer Relationship Management*') puis de la chaîne logistique (SCM : '*Supply Chain Management*'). Le problème est que ces blocs sont par essence fixes et statiques [Deblock, 03]. Or, l'environnement concurrentiel impose toujours plus de réactivité et pousse les processus métier vers des temps de réponse qui frisent le temps réel.

Des enquêtes ont également été effectuées qui montrent le besoin des entreprises à améliorer leurs processus ; citons l'enquête du cabinet IDC[3], en novembre 2002, qui indique que 60% des entreprises cherchent à améliorer leurs processus métier par le biais d'outils convenables [JDNet, 02]. Egalement l'enquête réalisée par le Benchmark Group[4] en avril 2003, révèle que presque 34% des entreprises estiment que la gestion la plus efficace des processus métier passe par la modélisation de l'existant, avant tout [JDNet, 03a].

Cet intérêt pour les processus a déjà encouragé quelques éditeurs de progiciels informatiques à travailler sur la fusion des processus. Or, nous pensons qu'une vision métier des processus est obligatoire avant de faire un développement informatique. Ensuite, il s'agit d'orienter le savoir-faire de la modélisation des processus métier vers un développement d'applications de modélisation pour l'entreprise, ce qui est l'un des objectifs de cette thèse.

Les processus jouent un rôle important dans l'entreprise, surtout pour une organisation transversale, car ils reflètent directement les niveaux de performance de celle-ci. C'est pour cette raison que les entreprises portent leurs efforts sur la modélisation de leurs processus, que ce soient les processus métier, mais aussi les processus de gestion et de support.

Cette modélisation nous paraît importante avant de passer à celle des activités. En effet, elle permet de traduire la stratégie de l'entreprise d'un projet, en identifiant les objectifs correspondant aux besoins et leur attribuer des processus dans le but de les achever (voir Figure I.2-3).

Le regroupement des activités dans les processus permet de simplifier la modélisation des activités et par voie de conséquence diminue la complexité du réseau d'activités de l'entreprise. C'est pourquoi, nous focalisons nos travaux sur les processus métier, car ce sont les processus principaux ayant un effet direct sur la satisfaction du client et la réalisation des objectifs de l'entreprise.

Cependant, une démarche par processus ne suffit pas en soi pour gérer l'entreprise (vision globale). Les responsables d'entreprise (managers) doivent bien évidemment diriger tous les départements et services de l'entreprise (vision locale). Pour cette raison, une organisation verticale (hiérarchique) est toujours importante à mettre en œuvre conjointement à l'organisation transversale [Mintzberg, 82] [Harzallah, 00].

---

[3] IDC : est le premier groupe mondial de conseil et d'étude sur les marchés des technologies de l'information. http://www.idc.com/france/

[4] JDNet ; http://solutions.journaldunet.com

I.4. AVANTAGES A DISTRIBUER LES MODELES DE PROCESSUS

Aujourd'hui, les grandes organisations telles que les entreprises industrielles de l'automobile, de l'aérospatiale, des télécommunications, de la défense et de la finance ont besoin de larges systèmes à grande échelle pour supporter la complexité et la dispersion géographique de leurs activités. La majorité de ces applications sont elles aussi distribuées sur de multiples machines fortement réparties géographiquement et très hétérogènes.

Au sein de ces applications complexes, les utilisateurs coopèrent à la réalisation des différentes activités de l'organisation (entreprises, administrations, etc.). Ces activités évoluent fortement dans le temps pour suivre les changements dans les législations ou réglementations locales, nationales et même internationales (par exemple, les directives européennes). De même, ces applications doivent évoluer pour répondre à de nouvelles opportunités du marché et à la création de nouveaux produits ou services. Les concepteurs de ces applications doivent donc pouvoir répondre rapidement à ces changements en créant de nouvelles fonctionnalités ou en recombinant celles existantes.

Aujourd'hui, un des défis les plus importants de l'informatique est de maîtriser la conception et le déploiement des applications réparties pour offrir un large éventail de services évolutifs accessibles aussi bien par le grand public que par des utilisateurs « experts ».

La modélisation, la réalisation et le déploiement d'applications réparties sont des tâches complexes. De nombreuses caractéristiques orthogonales doivent être prises en compte, par exemple : la communication entre les applications, l'hétérogénéité de celles-ci, l'intégration de l'existant et l'interopérabilité [Dogac, 98]. Ces caractéristiques doivent être traitées de front par les concepteurs d'applications réparties dans l'entreprise.

Du point de vue de la modélisation de processus, les modèles correspondants sont une base essentielle pour les concepteurs d'applications réparties. Le développement de ces modèles peut être partagé par des membres répartis géographiquement d'un groupe de travail. Le partage permet de rester toujours dans l'ingénierie simultanée, ce qui permet un gain de temps important de développement de modèles tout en collaborant.

Ce développement de modèles doit être un processus ou une méthodologie qui se base sur des concepts de modélisation établis et validés par des experts.

Par analogie avec les applications réparties, les modèles distribués permettent aux modélisateurs, dans un groupe de travail, de coopérer pour développer des modèles cohérents, qui couvrent l'aspect statique et dynamique d'une partie ou de l'entreprise entière.

Ces modèles évoluent fortement dans le temps pour suivre les changements dans les législations et l'économie locales, nationales ou internationales. Les modélisateurs doivent donc pouvoir répondre rapidement à ces changements en mettant à jour et/ou en faisant évoluer les modèles, et ceci tout en modélisant de nouveaux processus ou d'étapes de processus, ou en recombinant des morceaux ou des modèles entiers existants.

Par conséquent, la distribution de modèles de processus est une tâche importante dans le cadre de l'ingénierie simultanée pour gagner le pari de développer des modèles cohérents. Dans ce cas, plusieurs points importants doivent être pris en compte pour arriver à cette finalité :

- La communication entre les membres de groupes de travail en tenant compte de leur localisation géographique. Ce point est presque résolu grâce à la technologie web et des différentes interfaces homme-machine (IHM) (Internet Explorer, Netscape, Opera, etc.) sur chaque station connectée au réseau Internet ou Intranet.
- L'hétérogénéité qui est un point important pour la compatibilité de l'interface HM avec l'information reçue sur le réseau, sans tenir compte du système d'exploitation (Solaris, Unix, Microsoft Windows, IBM OS/2 ou Apple MacOS) de la station. Ceci est possible en utilisant les réseaux physiques (Ethernet, FDDI, ATM, X.25, etc.) et leurs protocoles associés (TCP/IP, Ethernet, Novell, etc.).
- L'intégration de l'existant avec les nouvelles technologies doit préserver les investissements passés et offrir de nouveaux services aux entreprises. Au lieu de re-développer complètement les modèles patrimoines, l'idée est ici de conserver la partie critique et vitale de ceux-ci et de faire progressivement évoluer et/ou migrer les modèles vers les nouvelles technologies (client/serveur ou web). L'intérêt de ce mode d'intégration est alors de pouvoir profiter des nouvelles possibilités et opportunités offertes par les nouvelles technologies tout en conservant l'existant.
- L'interopérabilité entre les outils informatiques qui permettent la mise en œuvre opérationnelle des modèles de processus. Dans cette optique, des normes acceptées et suivies par tous les acteurs impliqués dans cette coopération ont été définis.

Ces quatre points permettent la distribution de modèles et de leur partage, ce qui permet une meilleure coopération entre les acteurs modélisateurs.

Après avoir abordé les avantages de la distribution de modèles processus, il est important de savoir comment modéliser les processus. Ceci permet de bien structurer les modèles de processus et de les distribuer tout en ayant une structure unique. De ce fait, une étude des

architectures, méthodes et techniques de modélisation des processus d'entreprise est essentiel avant de commencer à concevoir le référentiel de composants de modélisation des processus d'entreprise proposé au chapitre IV.

Par conséquent, dans le chapitre suivant nous comparons les architectures et les méthodes de modélisation pour retenir celle qui répond à nos besoins tout en partant des différentes définitions de processus.

CHAPITRE II

LA MODELISATION DES PROCESSUS D'ENTREPRISE

CHAPITRE II : LA MODELISATION DES PROCESSUS D'ENTREPRISE

II.1. INTRODUCTION

La modélisation des processus répond à plusieurs objectifs. Lyons et Duffey [Lyons, 95] soulignent que, malgré certaines difficultés rencontrées, il existe un regain d'intérêt dans la recherche pour développer de nouveaux modèles de processus. Ce renouvellement s'explique par les bénéfices potentiels de la modélisation comme :

- la documentation des meilleures pratiques,
- l'identification de goulots d'étranglement et de redondance de tâches,
- la conduite d'analyses «what if» (que ce passe-t-il si ?) de possibilités de conception,
- l'évaluation de risque de dépassement de dates ou de coûts,
- l'archivage des processus,
- la formation, etc.

Huckvale et Ould [Huckvale, 95] recensent eux aussi les différents apports d'un modèle de processus comme :

- un support pour fixer l'attention lors de discussions;
- un outil pour décrire et expliquer une chaîne d'activités;
- un point de départ pour une analyse;
- un moyen pour concevoir un nouveau processus;
- une base pour l'amélioration continue des processus;
- et suivant le cas, un programme pour commander le processus réel.

Plusieurs méthodes et outils de modélisation ont été développés dans différents domaines. Kusiak et Larson [Kusiak, 95] passent en revue différentes techniques utilisées pour modéliser les processus en conception mécanique: les représentations sous forme de graphes, les représentations matricielles, les méthodologies IDEF. Dans le rapport final du projet Broom [Broom, 96] (Banking Process Reengineering through Object-Oriented Modelling,

ESPRIT Project No: 20395, http://hyperion.planet.gr/projects/broom), des techniques pour la modélisation des processus sont rappelées, notamment des modèles d'analyse fonctionnelle, des modèles de systèmes dynamiques, des approches orientées objet. Johansson et al. [Johansson, 93] présentent rapidement diverses techniques de modélisation, telles que des techniques d'étude du travail, des outils informatiques de représentation de flux de données, des outils de simulation des processus.

Wang [Wang, 94] et Huckvale et Ould [Huckvale, 95] rappellent les quatre points de vue les plus fréquents pour représenter un processus. Ces points de vue, initialement proposés par Curtis et al. [Curtis, 92], CIMOSA [AMICE, 93] et Zachman [Zachman, 87] et repris dans ARIS [Scheer, 99] peuvent être utilisés pour classer les méthodes et les outils de modélisation :

- le point de vue fonctionnel, une représentation des activités du processus en train d'être exécutées et les flux de produits ou d'information associés;
- le point de vue comportemental, une représentation de «quand» les activités du processus sont exécutées: enchaînements séquentiels, boucles de rétroaction, itérations, prises de décision, conditions de déclenchement, etc. ;
- le point de vue organisationnel, une représentation de «où» et «par qui » dans l'organisation, les activités du processus sont exécutées, ainsi que les mécanismes physiques de communication et les supports de stockage;
- le point de vue informationnel, une représentation des produits ou des informations générés ou manipulés par le processus, y compris leur structure et les interrelations.

Ces méthodes de modélisation des processus peuvent aussi être regroupées selon une autre classification:

- les méthodes qui proposent uniquement un langage graphique de représentation,
- les méthodes plus évoluées, qui, à partir d'une représentation, généralement graphique, fournissent une aide à l'analyse et/ou une aide à l'étude du comportement dynamique par le biais de la simulation.

Avant de parler des différentes méthodes et architectures de modélisation de processus, il est important de définir concrètement ce que l'on entend par processus.

II.2. LES PROCESSUS

II.2.1. DEFINITIONS

Plusieurs définitions du terme processus sont présentées dans la littérature. Il est vrai que cette notion est suffisamment générale pour être utilisée dans différents domaines scientifiques ou applicatifs. L'étude de ces définitions permet d'avoir une idée claire de ce qui est désigné par un «processus ».

- D'un point de vue général, Hachette [Hachette, 01] définit un processus comme :
  - Un développement plus ou moins réglé ou régulier de phénomènes.
  - Une suite continue de faits ou d'opérations aboutissant à un résultat déterminé.
  - Une succession ordonnée d'états par lesquels passe un système physique au cours de son évolution.
  - On caractérise un processus par les états initiaux et finaux correspondants.
- Du point de vue de l'automatique et de la productique, les définitions proposées sont les suivantes :
  - Selon CIMOSA [AMICE, 93], « un processus est déclenché par des événements provenant du système lui-même ou du monde extérieur (par des machines, commandes clients, ordres de gestion, etc.). Il est formé de sous processus et d'activités. C'est en fait une chaîne d'activités spécifiées au moyen de règles procédurales qui décrivent le comportement de l'entreprise en fonction des événements reçus et de l'état du système. Une activité décrit une tâche réalisée dans l'entreprise au moyen de ressources au cours du temps et consistant à transformer des intrants en extrants sous certaines contraintes, et produisant des informations de sortie.»
  - D'après Vernadat [Vernadat, 96] [Vernadat, 99] « Un processus métier (en anglais, Business Process) est une succession de tâches (ordre partiel) qui contribue à la réalisation des objectifs de l'entreprise. De manière générale, un processus peut être défini comme un enchaînement d'activités à exécuter pour atteindre un but donné. Cet enchaînement forme ce qu'il est convenu d'appeler le flux de contrôle du processus, c'est-à-dire sa logique d'exécution. [Curtis, 92] »
  - D'après ACNOS [ACNOS, 96], un processus opérationnel est « un enchaînement partiellement ordonné d'activités, mobilisant des savoirs-faire multiples et étant finalisés par un objectif. Le processus est déclenché par un ou plusieurs

événements et se termine par la production d'un résultat observable ou quantifiable ». Une activité est « un enchaînement d'opérations dont l'objectif est exprimé par le biais de la tâche. Chaque activité est caractérisée par une fonction qui transforme un état d'entrée, sous l'influence d'objets de contrôle et sous réserve de disponibilité de ressources nécessaires et de temps, en un état de sortie ».

- Des auteurs du *domaine du génie informatique* donnent les définitions suivantes :
  - Curtis et al [Curtis, 92] définissent un processus comme un ensemble partiellement ordonné d'étapes.
  - D'après 'CMU Software Engineering Institute' : « un processus est un ensemble d'étapes partiellement ordonnées, conçues pour atteindre un but. Un processus est décomposable en étapes et composants. Le concepteur représente un niveau atomique permettent d'avoir de grandes parties de processus, partant d'étapes de processus individuels [CMU/SEI, 93] ».
- *Du point de vue organisationnel,*
  - Harrington [Harrington, 91] définit un processus comme « n'importe quelle activité ou groupe d'activités qui reçoit une entrée, lui ajoute de la valeur et fournit une sortie à un client interne ou externe. Les processus utilisent les ressources de l'organisation pour fournir des résultats définitifs ».
  - Lorino [Lorino, 95] définit le processus comme « l'ensemble des activités reliées entre elles par des flux d'informations significatives et qui se combinent pour fournir un produit matériel ou immatériel important et bien défini ».
- Enfin, *du point de vue de la qualité,*
  - La norme ISO 8042 [AFNOR, 92] définit un processus comme « un ensemble de moyens et d'activités liés, qui transforment des éléments entrants en éléments sortants : ces moyens pouvant inclure le personnel, les installations, les équipements, les techniques et les méthodes. »
  - Selon ISO 9000 [Mathieu, 00], « chaque organisation existe pour améliorer un travail. Le travail est accompli à travers un réseau de processus. Chaque processus a des entrées, et des sorties qui sont les résultats du processus. La structure du réseau n'est pas toujours une structure séquentielle simple, mais complexe».
  - La version 2000 de ISO 9000 [Mathieu, 00] définit un processus comme un ensemble d'activités corrélées ou interactives qui transforme des éléments d'entrée en éléments de sortie.

De telles définitions sont utilisées pour modéliser, concevoir et implémenter des processus métier permettant ainsi d'organiser des réseaux complexes d'activités interdépendantes, en partant d'étapes de processus simples et linéaires.

Par déduction, nous définissons un processus d'une entreprise en tant qu'un enchaînement d'activités corrélées ou interactives. Un processus reçoit des objets en entrée et leur ajoute de la valeur, par le moyen de ressources, tout en fournissant des objets de sortie (produits/services) remplissant les besoins et les exigences d'un client (atteindre les objectifs) internes ou externes à l'entreprise. Il ne peut être déclenché que par des événements internes et/ou externes à l'entreprise, c'est-à-dire des changements d'état de composants du système. Chaque processus est en communication avec d'autres et peut être décomposé en sous-processus. Une activité transforme des entrées en sorties par l'influence d'objets de contrôle et en utilisant les ressources requises et disponibles pendant une durée bien définie.

Nous résumons notre définition par les Figures II.2-1 et II.2-2, ensuite nous faisons la synthèse des deux par la Figure II.2-3.

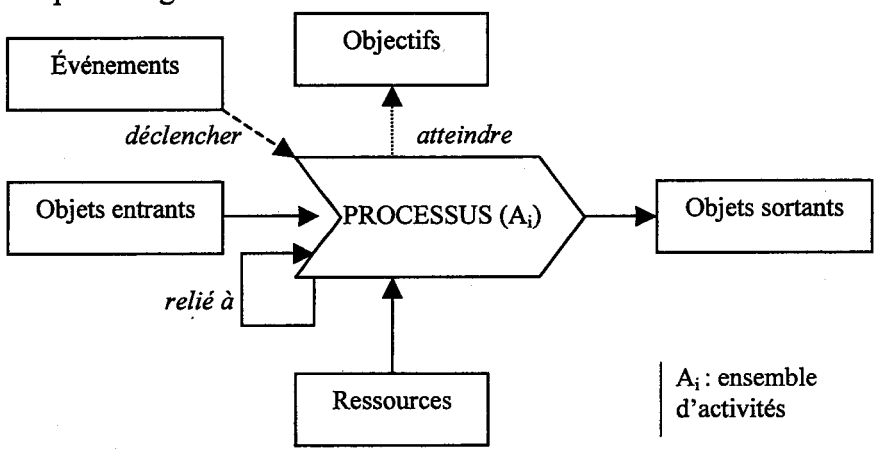

Figure II.2-1- Définition d'un processus

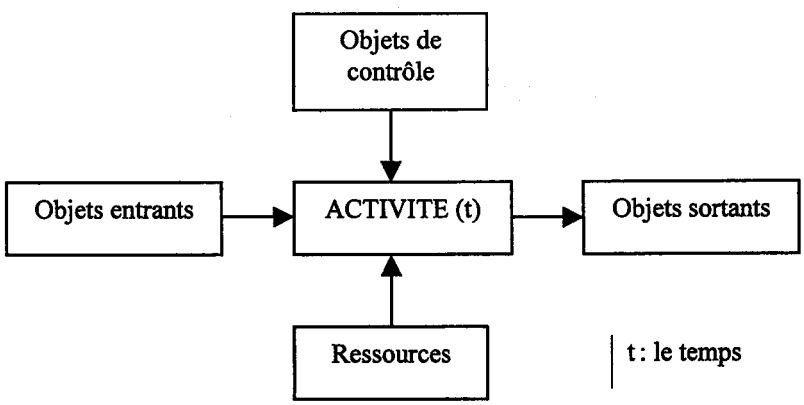

Figure II.2-2- Définition d'une activité

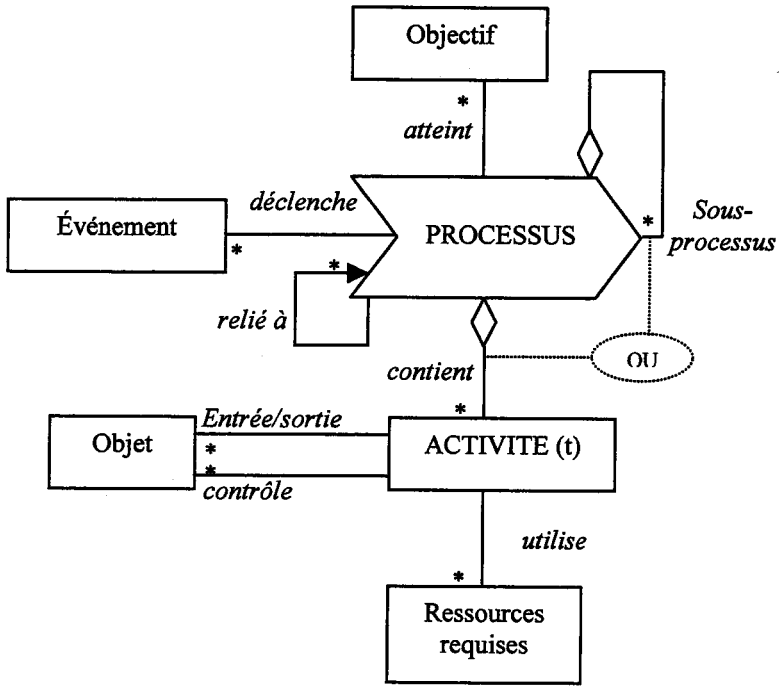

Figure II.2-3- Décomposition d'un processus

On remarque que la Figure II.2-3 rassemble logiquement les Figures II.2-1 et II.2-2. En effet, les objets d'entrées/sorties de toutes les activités d'un processus sont logiquement ceux de ce dernier, de même pour les ressources. La Figure II.2-3 fournit un méta-modèle fonctionnel de la structure d'un processus formalisé en UML[5].

En partant de la définition de l'ISO 9000, une entreprise ou un système englobe un ensemble de travaux à réaliser dans le but de réaliser les objectifs fixés par celle ou pour celui-ci. Et puisqu'un travail est accompli à travers un réseau de processus, nous pensons qu'il est important de bien comprendre ces derniers dans le but d'améliorer leurs performances, d'où celles de l'entreprise ou du système.

Par conséquent, une modélisation des processus est nécessaire pour décrire les chaînes d'activités, les analyser, comprendre leurs évolutions et améliorer leurs performances [Vernadat, 99] [ACNOS, 97].

Cette modélisation permet donc, d'améliorer la qualité des résultats (produit/service) obtenus selon les exigences et les besoins des clients internes et/ou externes à l'entreprise.

[5] Voir Annexe C

Dans cette optique, l'entreprise sera décomposable en processus, eux-mêmes décomposables, par niveaux successifs, en sous-processus ou en activités au plus bas niveau.

Partant de ce principe, il apparaît, en théorie du moins, que l'échelle des niveaux est infinie. L'entreprise elle-même fonctionne comme un processus en interaction avec ses clients, ses fournisseurs, ses concurrents, etc. Ce processus intervenant à son tour dans des processus socio-économiques plus globaux.

Par une dynamique permanente d'amélioration, l'entreprise peut espérer à terme améliorer tous les processus. Dans un premier temps, il est utile de s'intéresser à quelques processus et de préférence à ceux dont l'amélioration apportera le plus de gain pour l'entreprise. Ce choix de processus « prioritaires » constitue une phase importante lors de la modélisation et ne doit pas être faite au hasard.

Souvent l'observation de dysfonctionnements est à l'origine du déclenchement d'une modification du processus concerné [Berrah, 01]. Lorsque ce type de modification est réalisé au coup par coup, il y a fort à parier que l'optimisation du processus sera obtenue après une série de tâtonnements coûteux et dans bien des cas démobilisateurs.

Ces processus peuvent être définis et regroupés dans une cartographie de telle façon à identifier les processus principaux et secondaires en se basant sur une typologie qui convient à l'entreprise [Jacob, 94].

Dans ce qui suit, des catégories de processus sont données avec une typologie proposée par ISO 9000 version 2000 [Mathieu, 00].

II.2.2. CATEGORIE DE PROCESSUS

Généralement, dans la plupart des entreprises il existe un nombre restreint de processus essentiels, de trois à vingt selon la nature de l'entreprise (en anglais, Core processes). Par exemple, IBM France gère en permanence quinze processus, chez Dassault Aviation onze processus majeurs ont été identifiés, Hutchinson a identifié cinq processus critiques. Ces processus complexes font intervenir de nombreuses fonctions qui « traversent» l'entreprise [Cattan, 02].

On peut alors considérer qu'il y a deux grandes catégories de processus:

- Les processus qui constituent les fondements même de l'entreprise et que l'on pourrait qualifier de processus principaux.

  - Pour une société d'ingénierie: les processus de conception, de réalisation, d'exploitation, de maintenance, etc..
  - Pour une entreprise de fabrication: les processus de gestion de commandes clients, de planification de la production, d'usinage, de contrôle, etc.
  - Pour une entreprise de service: les processus de validation du service, d'après-vente, etc. L'analyse et l'amélioration de ce type de processus relèvent de décisions stratégiques au niveau de l'entreprise donc directement de la direction générale.
- Les processus qui correspondent de façon plus concrète au vécu et aux préoccupations de chacun. On peut les qualifier de « simples » eu égard au faible nombre de fonctions et de tâches impliquées.
  - Pour une société d'ingénierie, le processus de réalisation peut se décomposer en plusieurs processus: lancement d'affaire, études détaillées, approvisionnement, livraison, montage, etc.
  - Pour une entreprise de fabrication, le processus d'usinage peut se décomposer en processus de préparation de la machine à usiner (programmation, montage des outils...), application de la procédure d'usinage avec les différents points d'arrêt et de contrôle fixés par le bureau des méthodes, les contrôles dimensionnels, la remise en état de la machine après livraison de la pièce usinée.
  - Pour une entreprise de service, le processus de service après-vente pourrait se décomposer en : processus de planification de l'intervention, de mise à disposition du personnel compétent, d'intervention chez le client, le cas échéant de gestion des pièces de rechange, de retour d'expérience, de facturation...

L'analyse et l'amélioration de cette seconde catégorie de processus sont du ressort du management des équipes et des équipes elles-mêmes. Quels que soient les moyens dont dispose l'entreprise tous les processus ne pourront pas être analysés, voire améliorés, en même temps. Il faudra donc se fixer des priorités et agir par étape.

On observant les exemples donnés ci-dessus dans chacune des catégories, nous pouvons conclure que les processus de réalisation de produits et/ou de services constituent une partie importante des fondements essentiels de l'entreprise. D'autres types de processus existent comme les processus support, les processus de mesure, les processus de management, etc.

Dans ce contexte, la norme ISO 9000 version 2000 indique dans une note du § 4.1 : « il convient que les processus nécessaires au système de management de la qualité comprennent

les processus relatifs aux activités de management, à la mise à disposition de ressources, à la réalisation des produits et aux mesures». Il faut noter que l'expression utilisée « il convient » laisse la porte ouverte à l'utilisation de toute autre typologie.

La norme traite quatre familles de processus dans lesquelles, il faut bien le reconnaître, nous retrouverons l'essentiel des processus d'un organisme.

Ces quatre familles sont :

- Des processus de réalisation, que l'on appelle aussi processus opérationnels, processus à valeur ajoutée client, processus cœur de métier. Ces processus contribuent directement à la réalisation du produit depuis la détection du besoin du client jusqu'à sa satisfaction. Ils regroupent les activités liées au cycle de vie du produit. On peut citer comme exemples, dans le cas d'une entreprise industrielle : Développer un produit, Vendre un produit, Approvisionner un client, Fournir les services associés aux produits.
- Des processus support (Administrer le personnel, Dispenser des formations, Gérer le système d'information, Financer les investissements, Acheter), appelés aussi processus de soutien, processus ressources. Ces processus contribuent au bon fonctionnement des autres processus en leur apportant les ressources nécessaires.
- Des processus de management (Définir et déployer la stratégie, Piloter les activités, Manager l'amélioration continue), que l'on appelle aussi processus de pilotage ou de direction. Ces processus permettent principalement de conduire et guider l'organisme pour améliorer sa capacité à évoluer positivement, de vérifier si les décisions prises sont cohérentes avec les objectifs poursuivis et d'anticiper sur l'environnement.
- Les processus de mesure (Mesurer la satisfaction client, Mesurer l'efficacité des processus, Mesurer la satisfaction et la motivation du personnel, Mesurer et surveiller la qualité des produits, Mesurer la satisfaction de la collectivité). Ces processus contribuent au bon fonctionnement (maîtrise) et à l'amélioration des autres processus et du système de management de la qualité en fournissant des mesures par rapport à des objectifs préalablement définis, ceci afin de piloter les processus et le système de management de la qualité.

Bien que la norme ISO 9000 version 2000 donne un début de typologie des processus, elle n'en exige pas l'application et ne demande pas expressément qu'une typologie soit établie.

Dans la pratique, l'organisme ne doit utiliser une telle typologie que si cela lui rend effectivement service. Il appartient à l'organisme de déterminer, si pour des besoins de gestion, de clarté ou tout simplement de présentation, il y a lieu de définir une typologie de ses processus. L'organisme peut définir sa propre typologie ou, comme cela est souvent le cas, se passer d'une typologie compte tenu du faible nombre de processus représentant son activité. Il convient donc de veiller à ne pas tomber dans le travers qui consisterait à appliquer systématiquement les normes en oubliant que certains éléments de la norme ne sont donnés qu'à titre indicatif pour mieux faire comprendre l'exigence [Mathieu, 00].

Ces processus peuvent être regroupés dans une cartographie qui permet l'organisation des processus de l'entreprise. Notons que le terme cartographie est absent dans l'ISO 9001 version 2000. Par contre, il apparaît dans le fascicule de documentation de l'Afnor FD X50 176 sur le management des processus émis avant la parution de la version définitive de la norme.

La cartographie est une représentation des liens existants entre les différents processus de l'entreprise [Jacob, 94].

Il n'y a pas de cartographie type. En effet, il serait dangereux de voir derrière le mot « cartographie » une exigence en terme de modélisation d'un réseau de processus de l'entreprise. Cette modélisation serait nécessairement complexe et n'apporterait pas grande chose à l'entreprise, sinon une perte de temps et une crédibilité mise à mal de l'approche.

Dans le plus simple des cas et probablement le plus fréquent, établir une cartographie des processus de l'entreprise revient à en dresser une liste ordonnée suivant un niveau de criticité ou d'importance.

II.3. METHODES DE MODELISATION GRAPHIQUE DES PROCESSUS

Initialement, les méthodes de modélisation graphique des processus ont été développées dans le but de représenter les flux d'information et de matières. Ensuite, elles sont été utilisées pour aider à la conception de processus efficaces grâce à des supports graphiques.

Dans ce contexte, nous citons les diagrammes de flux (flowcharts) [Harrington, 91] qui sont des outils d'aide à la compréhension des processus de travail et des relations entre les procédures de travail. Selon Harrington, «*A flowchart is worth a thousand procedures*[6] ». Les diagrammes de flux se basent sur l'utilisation de symboles simples, de lignes, de flèches, de messages textuels nommant les activités et expliquant les séquences du processus.

[6] «Un diagramme de flux vaut une centaine de procédures », traduction libre.

Les diagrammes-blocs sont les diagrammes les plus simples, ils permettent d'avoir une vue globale et rapide d'un processus. D'autres types de diagrammes existent [Harrington, 91] [Johansonn, 93].

SADT (Structured Analysis and Design Technique) (Figure II.4-1), proposée par Ross [Ross, 77] à la fin des années 1970 pour permettre une analyse structurée des systèmes, a ouvert la voie à la modélisation par représentation graphique des activités et des chaînes d'activités.

La méthode SADT introduit le principe de décomposition fonctionnelle et formalise le concept d'activité. Elle se présente comme un langage graphique et un ensemble limité de primitives, des «boîtes» et des «flèches», pour la représentation des composants des systèmes et des interfaces.

Des méthodes de modélisation graphique plus structurées sont depuis disponibles et notamment les DFD (diagrammes de flux de données) introduits par De Marco en 1978. Le formalisme DFD est à la fois un ensemble de symboles et une méthodologie pour la création de modèles de flux d'information. Plusieurs niveaux de détails sont possibles:

- représentation du système dans sa globalité,
- représentation de flux d'information additionnels et plus détaillés en différents niveaux.

Les DFD permettent une représentation claire, mais simplifiée, des flux informationnels et matériels.

Les méthodes précédemment citées et d'autres aussi (§ I.3.) offrent des possibilités de modélisation graphique des processus. D'autres méthodes, plus élaborées mais toujours issues du génie logiciel proposent des supports d'analyse statique ou dynamique en se basant sur des approches fonctionnelles, relationnelles ou objet (MERISE et ses modèles de traitement [Tardieu, 83], OMT [Rumbaugh, 91], OOD [Booch, 91], OOSE [Jacobson, 92], UML [Booch, 00], etc.).

II.4. METHODES D'ANALYSE STATIQUE ET DYNAMIQUE

La modélisation matricielle peut être utilisée pour décrire des relations entre des éléments. Par exemple : la succession temporelle d'activités,

- les flux de produits ou d'information entre les activités,

- des contraintes entre des variables de conception.

L'avantage de cette modélisation est de permettre ensuite un traitement mathématique des données. Après manipulation mathématique de la matrice :

- des cycles entre activités peuvent être détectés,
- des possibilités de chevauchement de tâches peuvent être mises en évidence,
- des groupes d'activités pouvant être exécutés en parallèle peuvent être identifiés.

La famille de méthodes IDEF (Integrated DEFinition) a été développée à partir des années 1970 sous les auspices du programme ICAM (Integrated Computer Aided Manufacturing program) de l'U.S. Air Force [Bravoco, 81]. Ces méthodes fournissent une réponse aux besoins ressentis en analyse et en communication entre les personnes impliquées dans l'amélioration de la productivité des entreprises [Johansson, 93].

Famille des méthodes IDEF :

- IDEF0 a été développée à partir de SADT (Figure II.4-1), elle est utilisée pour décrire les aspects fonctionnels d'un système.

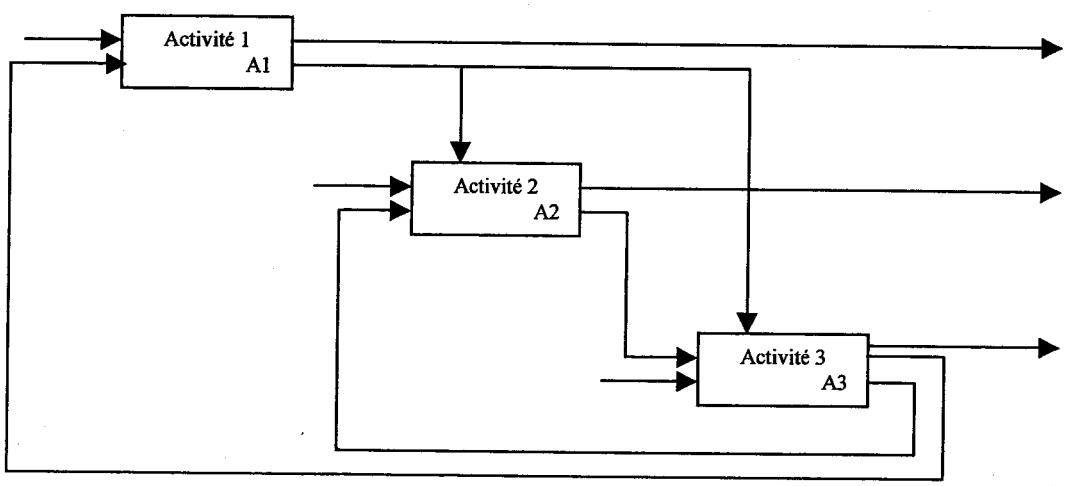

Figure II.4-1- Diagramme IDEF0/SADT (Actigramme)

- IDEF1 est dédiée à l'analyse de l'information (modèle entité-relation).
- IDEF1x est une méthode augmentant IDEF1 pour la conception de bases de données relationnelles.
- IDEF2 sert à l'analyse dynamique (à base de files d'attente).

- Dans cette suite logique, IDEF3 est spécialement conçue pour la modélisation des séquences d'activités ou processsus (Figure II.4-2).
- IDEF4 est spécialisée pour la conception de bases de données orientées objet.
- IDEF5 est utilisée pour construire des ontologies.

D'autres méthodes IDEF font encore l'objet d'études et la liste présentée est non exhaustive.

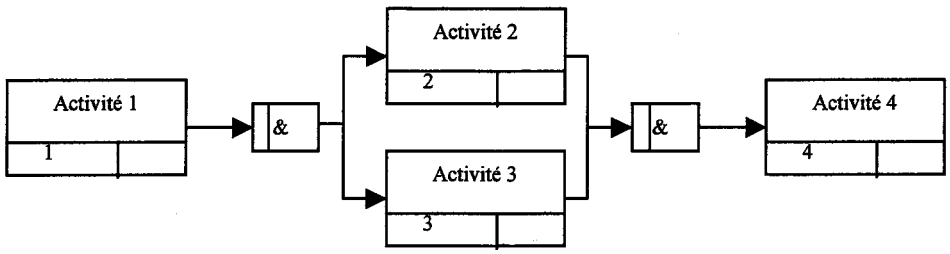

Figure II.4-2- Diagramme IDEF3 (Process Flow)

A ces méthodes de modélisation et d'analyse statique s'ajoutent des outils pour l'analyse dynamique des processus. Parmi ces outils, la simulation permet de modéliser, de comprendre et d'améliorer les processus. Traditionnellement utilisée pour analyser le système de production, la simulation peut aussi être employée pour modéliser et analyser les activités de développement.

De plus, toutes ces méthodes et outils sont souvent combinés pour apporter des informations supplémentaires sur le processus étudié. A titre d'exemple, dans le projet ACNOS [ACNOS, 97], IDEF3 a été couplée à un outil de simulation, en l'occurrence les réseaux de Petri [Proth, 95], pour une analyse du fonctionnement dynamique d'un système.

D'autre part, un langage de modélisation unifié a vu le jour au milieu des années 90. Il s'agit de UML (*Unified Modeling Language*) [Booch, 00] [Rumbaugh, 91]. Celui-ci offre un langage graphique clair et unifié pour faciliter la collaboration entre les acteurs d'un même groupe de travail. Il permet la modélisation graphique des aspects statiques et dynamiques d'un système informatique (exemple : système d'information). En effet, il offre des diagrammes basés sur l'approche objet, permettant de décrire ces aspects. Ce langage n'offre pas une méthode de modélisation, ce qui donne la liberté aux utilisateurs d'adopter les méthodes qu'ils jugent efficaces pour leur environnement. Ce langage a été adopté par des industriels pour la modélisation des systèmes techniques (exemple : les systèmes de

production) de l'entreprise [Marchall, 99]; utilisé dans ce contexte, on parle de modélisation métier.

En complément des méthodes et des outils cités précédemment, des approches orientées objet pour la modélisation des processus opérationnels se développent dans le domaine de l'intelligence artificielle [Wang, 94]. De même, il existe plusieurs études sur la modélisation en entreprise dans son ensemble. Diverses méthodologies d'intégration d'entreprise et architectures de références pour la modélisation en entreprise ont été conçues au cours des quinze dernières années. Parmi les plus connues, on peut citer : l'ENV 40003, CIMOSA, GRAI-GIM, OLYMPIOS, PERA, ARIS, et GERAM. Ces méthodologies peuvent fournir un cadre rigoureux de travail pour l'analyse et la re-conception des processus de développement de produit en cohérence avec les principes de l'ingénierie simultanée.

II.5. ARCHITECTURES DE REFERENCE

II.5.1. CEN ENV 40003

CEN ENV 40003 [Shorter, 02] est une prénorme du Comité Européen de Normalisation (CEN) pour la modélisation en entreprise. Son but est de préciser la terminologie et d'énoncer les principes fondamentaux sous-jacents au domaine de la modélisation en entreprise. L'architecture de référence retenue est basée sur le cadre de modélisation de CIMOSA (Figure II.5-1).

Cette ENV est accompagnée d'une ENV plus récente précisant les constructs recommandés pour la couche générique de l'architecture. Il s'agit de l'ENV 12204 [CEN, 90] [ENV, 96] qui définit les constructs suivants comme base du langage générique de modélisation en entreprise: événement, processus, activité, vue d'objet, produit, ordre, ressource, unité d'organisation et cellule d'organisation [CEN, 90]. Ceux-ci sont décrits dans les chapitres suivants.

Les constructs de cette architecture sont en grande partie équivalents à ceux de CIMOSA et définis par les mêmes pro-formats (templates).

II.5.2. CIMOSA

CIMOSA (CIM Open System Architecture) [AMICE, 93] est une architecture pour construire des systèmes intégrés de production. Elle a été développée par le Consortium AMICE dans le cadre de projets ESPRIT. Cette architecture comprend:

- un cadre de modélisation (MFW « Modeling FrameWork »);
- une plate-forme d'intégration (IIS « Integrating InfraStructure »); et
- le cycle de vie d'un système CIM « Computer-Integrated Manufacturing » (SLC « System Life Cycle »).

Le cadre de modélisation formalise trois principes fondamentaux et orthogonaux pour la modélisation en entreprise suivant une structure à trois axes, communément appelée cube CIMOSA, illustrée par la Figure II.5-1. Les trois axes retenus sont:

- L'axe d'instanciation qui se compose de trois niveaux: un **niveau générique**, où sont définies les primitives de base du langage de modélisation (appelées «constructs» en anglais), un **niveau partiel** contenant des modèles partiels, c'est-à-dire des structures prédéfinies et réutilisables pour un domaine d'application donné, et un **niveau particulier** correspondant aux modèles spécifiques de l'entreprise. Les niveaux générique et partiel constituent l'architecture de référence de CIMOSA (sujette à normalisation) alors que le niveau particulier correspond à l'architecture particulière d'une entreprise donnée (développée pour les besoins particuliers des utilisateurs).
- L'axe de dérivation qui préconise de modéliser l'objet d'étude suivant trois niveaux de modélisation: un **niveau de définition des besoins** permettant l'écriture du cahier des charges dans le langage de l'utilisateur final, un **niveau des spécifications de conception** permettant de spécifier et d'analyser dans le détail des solutions répondant aux besoins exprimés et un **niveau de description de l'implantation** (ou implémentation) permettant de décrire précisément l'implantation de la solution retenue.
- L'axe de génération qui définit quatre **vues** essentielles de modélisation permettant d'accéder au modèle en se focalisant sur certains aspects et en négligeant les autres (ceci permet de gérer la complexité du modèle). D'autres vues peuvent être définies si nécessaires.

Les vues retenues dans CIMOSA sont:

- La **vue fonction**, utilisée pour décrire la fonctionnalité et le comportement de l'entreprise en termes de processus, d'activités et d'opérations, soit ce qu'il y a à faire à différents niveaux de détail.

- La **vue information**, utilisée pour décrire les objets de l'entreprise, leurs relations et leurs différents états possibles, c'est-à-dire quels sont les objets traités et comment ils sont gérés.
- La **vue des ressources**, utilisée pour décrire les moyens nécessaires à mettre en oeuvre pour réaliser les fonctions de l'entreprise, leur rôle et leur mode de gestion, c'est-à-dire qui fait quoi, quand et comment.
- La **vue organisation**, utilisée pour décrire la distribution des responsabilités et des autorités dans les prises de décision, c'est-à-dire qui est responsable de quoi.

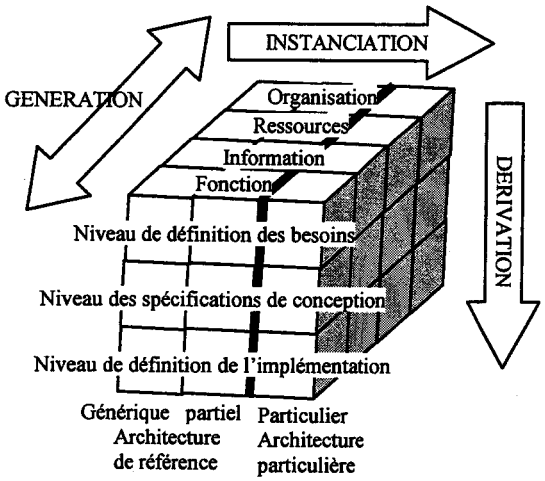

Figure II.5-1- Cube CIMOSA

Remarque **:** *Il est important de bien comprendre que les vues de CIMOSA ne sont pas des sous-modèles indépendants mais des mécanismes d'accès à certains aspects d'un même modèle intégré par filtrage (ou sélection) d'éléments du modèle.*

Le but de la modélisation de CIMOSA est, d'une part, de fournir un support à l'ingénierie systématique d'un système intégré de production et, d'autre part, de participer à l'intégration du système en utilisant le modèle d'entreprise pour piloter les opérations de l'entreprise. Le modèle est alors élaboré suivant la méthodologie d'intervention de CIMOSA et utilisé sous une forme exécutable par les services de la plate-forme d'intégration [AMICE, 93].

CIMOSA offre des langages de modélisation intégrés pour les aspects fonctionnels, informationnels, ressources et organisationnels [Vernadat, 96]. Les deux derniers demandent à être plus finalisés, ce qui est en partie l'objet de nos travaux. D'autre part, CIMOSA ne propose pas seulement une représentation graphique des activités et des processus (Figure

II.5-3), mais fournit un langage formel de description basé sur des «Constructs» ayant une syntaxe et une sémantique [AMICE, 93] [Vernadat, 96].

- Les constructs (au sens de constructions ou briques de base) sont des structures informatiques élémentaires (primitives) constituant les éléments de base du cadre de modélisation de CIMOSA. Ils sont définis dans la couche générique du cube CIMOSA (Figure II.5-1). Ils sont exprimés sous forme de pro-formats (templates) dans le document technique de référence de CIMOSA. Ils auraient pu l'être sous forme de classes d'objet.
- Les modèles CIMOSA sont des structures informatiques complexes issues de l'agrégation et de la spécialisation (particularisation) des constructs génériques.

Les langages de CIMOSA sont utiles pour toutes les étapes de conception et d'implantation de nouveaux modèles orientés-processus.

a) Vision de la modélisation en entreprise par CIMOSA

CIMOSA perçoit l'entreprise comme un système dynamique organisé en des domaines, interagissant entre eux et contenant des processus maîtres (domain processes : DP), générant et utilisant des événements et des objets d'entreprise (utilisés sous forme de vues d'objets ou «Object Views»). Les processus maîtres sont activés par des événements. Ainsi, CIMOSA est piloté par les événements et centré processus, par définition. La Figure II.5-2 montre la vision de la modélisation en entreprise de CIMOSA. Le domaine (DMx) capture les objectifs et les contraintes particuliers. Les fonctionnalités globales d'un domaine sont identifiées comme des processus maîtres (DPx.y). Les événements déclenchent les processus maîtres et changent les vues d'objets. Le domaine et le processus maître sont les constructs essentiels de CIMOSA qui définissent un environnement CIM bien structuré.

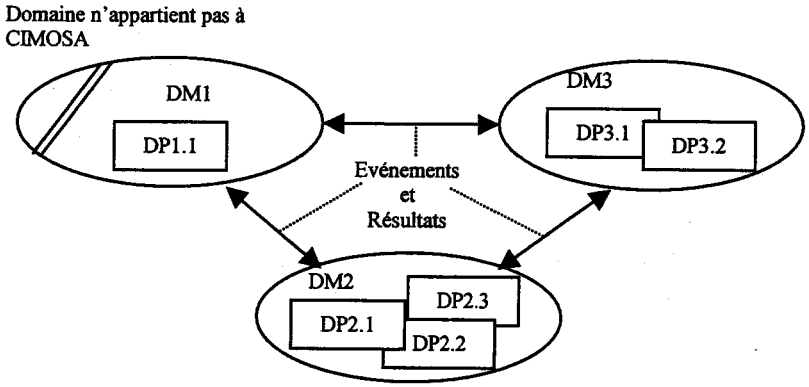

Figure II.5-2 - Ensemble de domaines et de processus maîtres dans CIMOSA

Les processus maîtres définissent les fonctionnalités majeures et le comportement de l'entreprise. La Figure II.5-3 montre des processus déclenchés par des événements et exécutés par des activités d'entreprise (Enterprise Activity) (EAn), qui sont exécutées par un ensemble de règles procédurales formant un réseau d'activités.

Les événements et les règles procédurales représentent le comportement de l'entreprise, et les activités représentent le fonctionnement des processus maîtres.

Les objectifs et les contraintes (Règles déclaratives) représentent le savoir-faire industriel (règles, procédures industrielles). Les activités d'entreprise utilisent et créent des objets d'entreprise (objets physiques ou entités d'information) définis dans le modèle comme des vues d'objets (Object Views). Pour faciliter la structuration du modèle, CIMOSA fournit le concept du processus métier (business process) (BPx.y.z), qui décompose un processus maître en sous processus réutilisables. Ainsi, un processus maître entièrement défini équivaut à un réseau d'activités d'entreprise reliées par un ensemble de règles procédurales (connecteurs ET, OU, XOR). Cette importante caractéristique de CIMOSA, fournit un modèle instantié de :

- Flux de contrôle (sous forme de règles comportementales)
- Flux informationnel sous forme de diagrammes de flux de données, en considérant les vues d'objets informationnelles (objets de nature information).
- Flux matière. Dans ce cas, les vues d'objets physiques sont considérées (objets de nature matérielle).

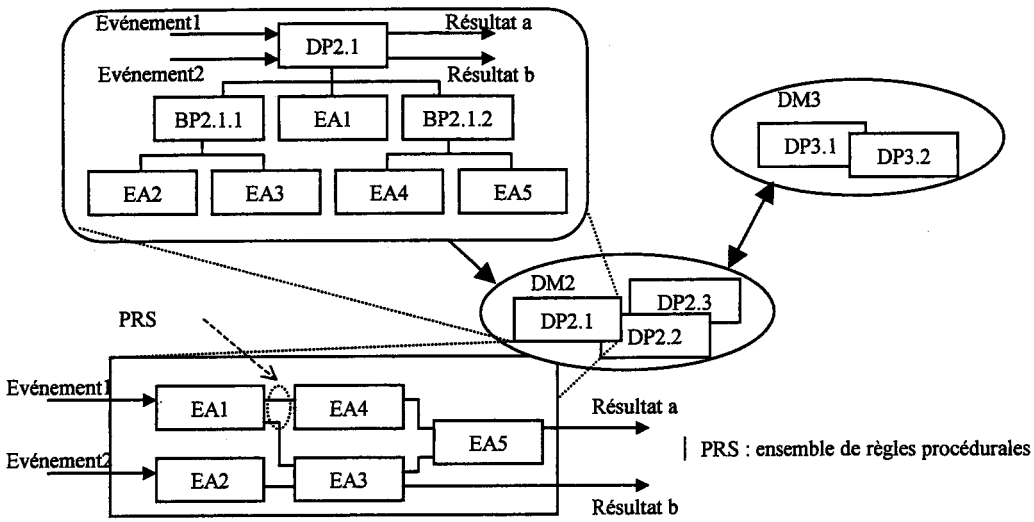

Figure II.5-3 - Décomposition des processus maîtres en un réseau d'activités d'entreprise dans CIMOSA

Cette architecture permet de traiter les aspects essentiels de l'entreprise en partant de l'expression des besoins jusqu'à la définition de l'implémentation. Grâce à son langage de modélisation par constructs, elle permet d'élaborer de modèles de processus métier et d'établir les entités fonctionnelles nécessaires à leur réalisation.

Ceci est bénéfique à nos travaux de thèse, du fait que CIMOSA, par une approche de méta-modélisation, définit le concept processus métier et son interdépendance avec les entités (objets entrants/sortants, ressources, ressources humaines, ...) de l'entreprise.

II.5.3. GRAI-GIM

La méthode GRAI (Graphe de Résultats et Activités Interreliés) est une méthodologie de modélisation et d'analyse des systèmes de décision des entreprises de production de biens ou de services. Elle a été développée à l'origine par les Professeurs Pun et Doumeingts de l'Université de Bordeaux. Elle s'appuie sur deux outils: la grille GRAI et les réseaux GRAI [Doumeingt, 84] [Roboam, 93].

La méthode GRAI a été largement utilisée depuis 1981 pour l'analyse et la conception de systèmes de gestion d'entreprises manufacturières, principalement en gestion de production.

Depuis, elle a fait l'objet d'extensions, en particulier dans le cadre de projets ESPRIT de la CEE pour les systèmes intégrés de production (CIM). Ceci a donné naissance à la méthodologie GIM (GRAI Integrated Methodology) au début des années 90. En termes de modélisation en entreprise, GIM peut être comparée à CIMOSA en ce sens qu'elle propose un cadre de modélisation, des outils de modélisation et une méthodologie.

Le cadre de modélisation de GIM est illustré par la Figure II.5-4. Celui-ci reprend les notions de vues de modélisation et de niveaux d'abstraction vus précédemment. GIM considère quatre vues: information (données/connaissances), décision (chaînes d'activités et centres de décision), physique (ressources) et fonction (décomposition fonctionnelle). La démarche consiste à modéliser un système d'entreprise au niveau conceptuel (définition des besoins tels que perçus par les utilisateurs), puis au niveau structurel (définition d'une solution technologique validée par les utilisateurs) et enfin au niveau dit réalisationnel. Le niveau réalisationnel décrit l'implantation du système conçu ou réorganisé en prenant en compte les aspects organisationnels, les aspects relatifs aux technologies de l'information et les aspects relatifs aux technologies industrielles, en particulier manufacturières.

Contrairement à CIMOSA, GIM ne cherche pas à fournir un langage de modélisation unique basé sur des constructs qui lui soient propres mais préfère utiliser des formalismes existants. Ainsi, GIM s'appuie sur les formalismes Merise (modèles de données, modèles de traitements), réseaux GRAI, IDEF0 et modèle relationnel pour les niveaux conceptuel et structurel.

La force de la méthode GIM réside dans sa méthodologie d'intervention qui a été très développée pour les deux parties comme indiquée dans la Figure II.5-4 : partie centrée sur l'utilisateur et partie centrée sur la technologie. Longtemps restée une méthode manuelle, la méthode GIM bénéfice maintenant d'un support informatisé au niveau conceptuel grâce à l'outil GRAITools. Le niveau réalisationnel est exprimé dans le langage des outils utilisés pour l'implantation du système.

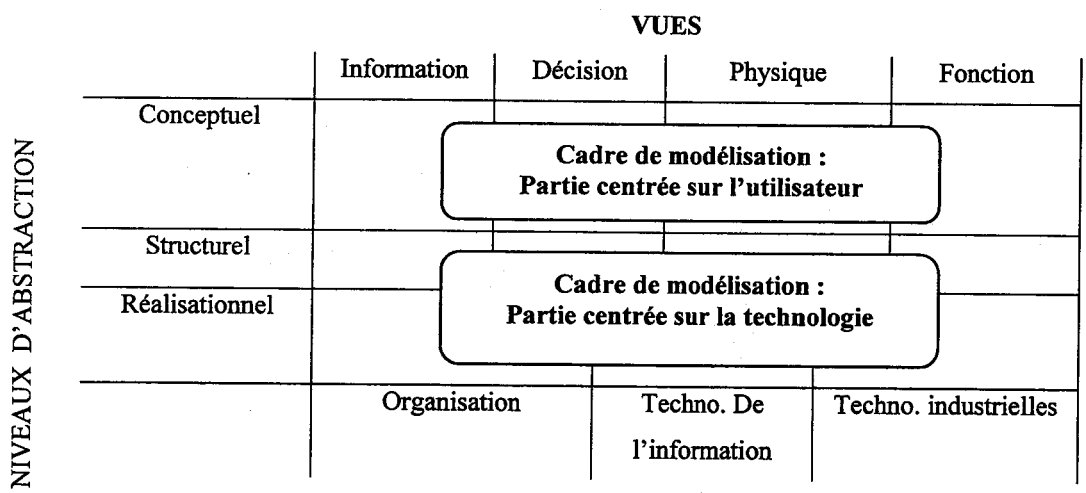

Figure II.5-4- Cadre de modélisation de GIM

Le formalisme de GIM est supporté par deux représentations formelles graphiques qui sont la grille GRAI et le réseau GRAI.

La grille GRAI est l'expression d'une vision globale et macroscopique de la structure du système étudié. Elle est basée sur une décomposition verticale de l'entreprise afin d'identifier ses centres de décision (CD) par fonction et suivant les niveaux de décision (Figure II.5-4). La grille situe ensuite les différents centres de décision les uns par rapport aux autres et met en évidence les principaux liens décisionnels de l'organisation étudiée [Doumeingts, 84][Roboam, 93].

Un centre de décision est un ensemble d'activités de décision appartenant à un même niveau horizon-période et remplissant la même fonction. A chaque centre de décision, sont rattachées des activités d'informations nécessaires à la préparation des données pour les activités de décision. Par ailleurs, des réseaux GRAI sont associés à chaque centre et servent à modéliser la fonctionnalité du centre (sous forme d'un réseau d'activités). Des règles de dysfonctionnement sont développées pour étudier les dysfonctionnements dans la grille GRAI (Figure II.5-5) établie.

La grille GRAI donne une vision globale des centres de décision dans l'entreprise. Cependant, son découpage par fonction et par niveau n'est pas toujours valable, surtout lorsque des décisions peuvent se prendre entre plusieurs fonctions ou/et plusieurs niveaux. Enfin, son découpage par fonction tend à préserver une organisation par fonction et non par projet ou par processus, et par conséquent elle privilégie les missions de chaque fonction à l'égard des missions principales de l'entreprise.

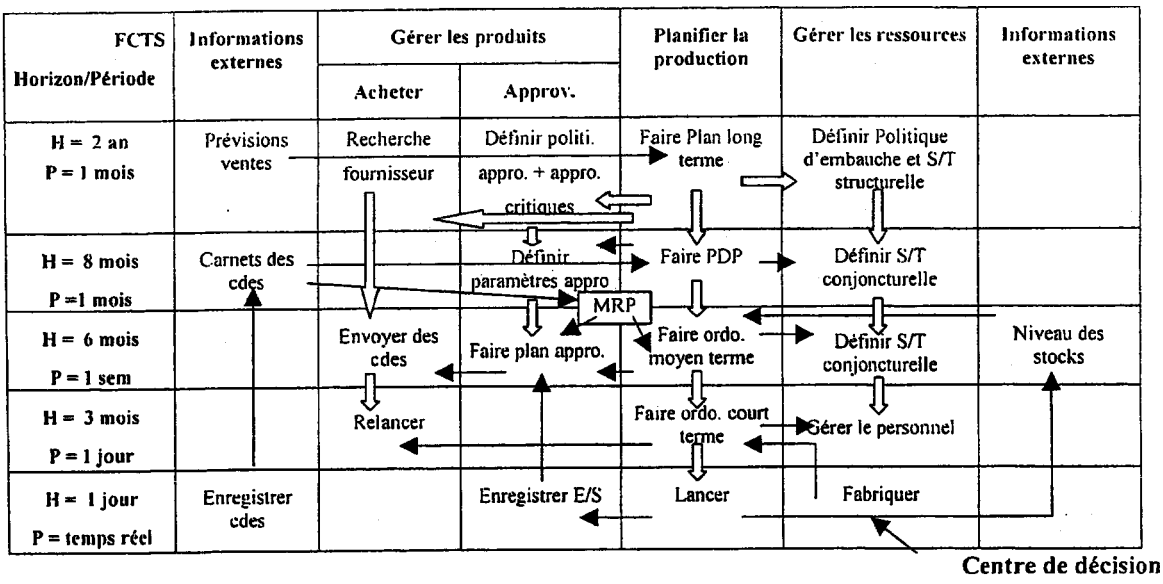

Figure II.5-5- Exemple de Grille GRAI (D'après [Roboam, 93])

Cette grille est complétée par le réseau GRAI (Figure II.5-6). Ce dernier décrit la structure de plusieurs activités dans chaque centre de décision déjà identifié dans la grille.

Les éléments fondamentaux du réseau sont les activités. Chaque activité :

- Possède un état initial et un état final,
- Requiert un support d'information, et
- Produit des résultats.

En plus, une activité de décision nécessite un support de cadre de décision.

Un résultat d'activité peut être l'entrée ou la ressource d'une autre activité.

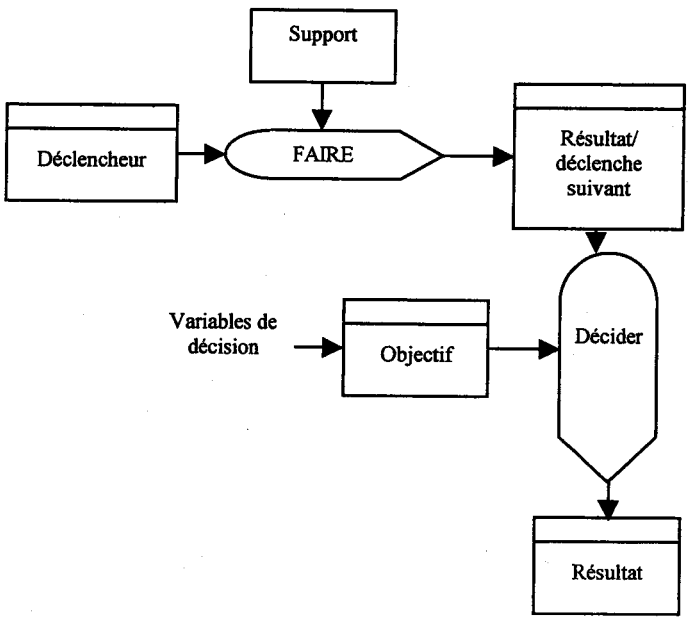

Figure II.5-6 – Exemple de réseau GRAI (d'après [Shorter, 94])

GRAI-GIM reste une architecture performante pour l'organisation des activités des centres de décisions de l'entreprise. Ce qui n'est pas le cas de CIMOSA.

Il nous paraît que les deux architectures peuvent être complémentaires du fait que l'une (GRAI-GIM) traite la décision et l'autre (CIMOSA) pilote les opérations de l'entreprise.

II.5.4. PERA

PERA (Purdue Enterprise Reference Architecture) [Williams, 92] est une méthodologie complète d'ingénierie des environnements industriels développée par le Prof. Williams, Purdue University, USA. Elle peut être généralisée au développement de tout système d'entreprise (système industriel, atelier, usine ou département de toute nature). La

méthodologie définit toutes les phases du cycle de vie d'une entité industrielle depuis sa conceptualisation jusqu'à sa mise en opération en passant par les phases de conception. La Figure II.5-7 précise l'architecture de la méthodologie, organisée suivant le cycle de vie de toute entité industrielle. Les numéros indiquent les différentes étapes de la méthode.

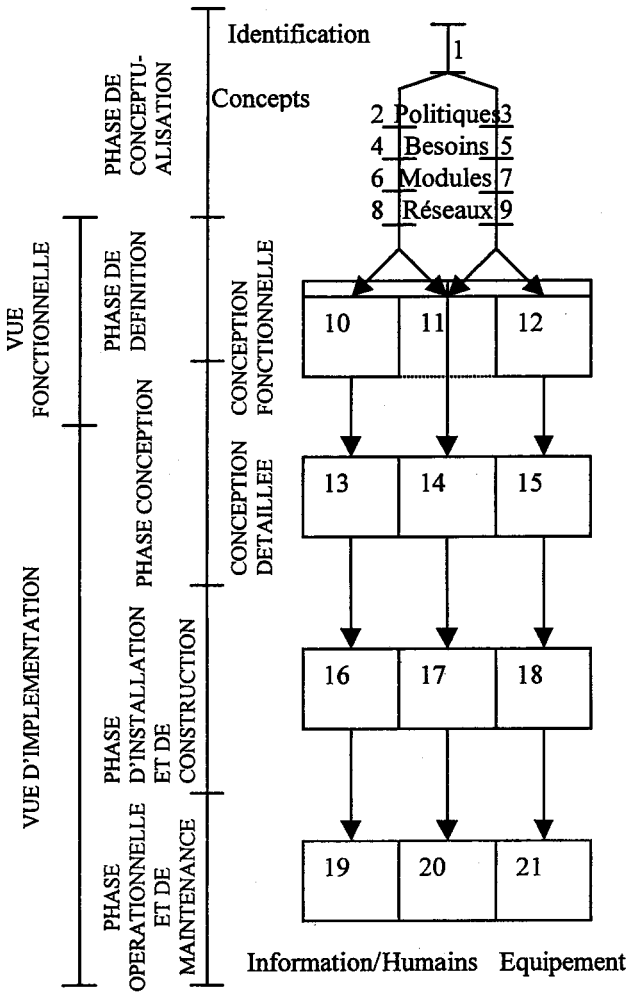

Figure II.5-7- Structure de l'Architecture PERA

L'originalité de PERA réside dans la prise en compte des aspects humains dans la méthodologie et de leur positionnement clair dans l'architecture, entre la partie commande (Systèmes d'information et de gestion) et la partie opérative (machines et équipements). Autant que, PERA admet qu'il n'existe pas de séparation nette entre chaque partie et que certaines opérations peuvent être réalisées à la fois par l'humain et la machine.

II.5.5. ARIS

ARIS (Architecture for integrated Information Systems).

Cette architecture a été développée par le professeur Scheer à l'université de Saarbruk en Allemagne [Scheer, 99].

Sa structure entière est similaire à celle de CIMOSA, mais à la place de se focaliser sur les systèmes CIM, elle traite les entreprises avec des méthodes traditionnelles orientées métier (planning de production, inventaires de contrôles, etc.). Elle se focalise surtout en ingénierie des logiciels et les aspects organisationnels de la conception des systèmes intégrés dans l'entreprise.

La Figure II.5-8 fournit une vue globale de cette architecture. Elle est structurée en quatre vues et trois niveaux de modélisation. Ces trois derniers sont ceux de CIMOSA.

Les quatre vues sont :

- **Vue fonction** : qui définit le modèle fonctionnel comme une hiérarchie de fonctions, et puis les spécifier en termes de modules de programmation pour générer un code.
- **Vue de données** : qui définit les modèles de sémantique de données (modèle entité-relation), et les traduire en schémas relationnels avant de les implémenter en systèmes physiques de bases de données.
- **Vue organisation** : qui définit la structure de l'entreprise en diagramme d'organisation, topologie réseau et son implantation. Cette vue couvre aussi l'aspect ressource.
- **Vue contrôle** : reliée aux autres vues. Dans cette vue, les processus métier et les chaînes d'activités sont rassemblés et implémentés comme des séquences logiques d'une exécution d'un programme.

ARIS est une architecture ouverte dans le sens que les formalismes utilisés dans les différents niveaux et vues de l'architecture ne sont pas fixés pour toujours. Ils peuvent être mis à jour ou étendus quand de nouvelles méthodes apparaissent.

ARIS est mise en œuvre par un outil : ARIS ToolSet. Elle est largement utilisée dans l'industrie allemande et ailleurs et est appliquée à la réingénierie des processus métier des systèmes de gestion d'informations [Scheer, 99] et notamment avec SAP R/3 pour l'ingénierie métier.

En effet, la méthode EPC (Event-drivern Process Chain) de ARIS, qui a été développée en 1992 à l'université de Saarland en Allemagne en collaboration avec SAP AG, est un élément principal dans les concepts de modélisation de SAP R/3 [Scheer, 99].

Cette méthode est basée sur les concepts des réseaux stochastiques et des réseaux de Petri [Proth, 95]. Plusieurs fonctions de cette méthode peuvent être le résultat d'événements. D'autre part, d'autres fonctions peuvent être nécessaires à mettre avant le déclenchement d'événements. Des relations logiques sont illustrées par les symboles « ET » ($\wedge$), « OU » ($\vee$) et « OU-exclusif » (XOR). La Figure II.5-9 donne un exemple de relations d'événements dans EPC. Ceci rappelle les méthodes de traitement de MERISE

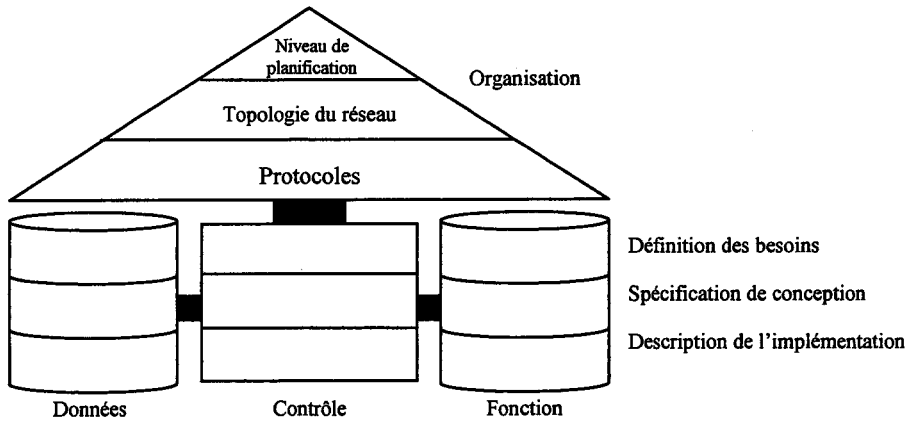

Figure II.5-8- Architecture de ARIS

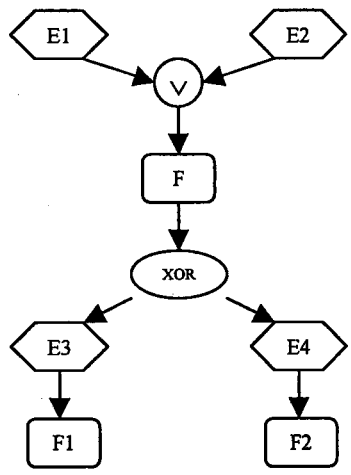

Figure II.5-9 – Relations entre événements dans EPC (d'après [Scheer, 99])

II.5.6. GERAM

GERAM (Generalized Enterprise Reference Architecture and Methodology) [IFAC, 97] est une architecture de référence développée par un groupe de réflexion sur les architectures pour l'intégration des entreprises (IFAC/IFIP Task Force on Architectures for Enterprise Integration). GERAM est en fait une généralisation de CIMOSA, de GRAI-GIM [IFAC, 97], de PERA et de quelques autres architectures (ARIS, ENV 40003 et IEM[7]). La structure de cette architecture est donnée par la Figure II.5-10 dans laquelle on reconnaît le cube CIMOSA (niveaux générique, partiel et particulier, les vues fonction, information, ressources et organisation, les niveaux de modélisation des besoins, de conception et d'implémentation), le cycle de vie dérivé de celui de PERA (à la terminologie près et en y ajoutant une phase de démantèlement) et les trois composantes de PERA (partie commande, partie humaine et partie opérative).

Cette architecture est une référence dans le domaine tant comme cadre méthodologique que pour le positionnement d'outils et de méthodes de modélisation pour l'ingénierie des systèmes industriels intégrés.

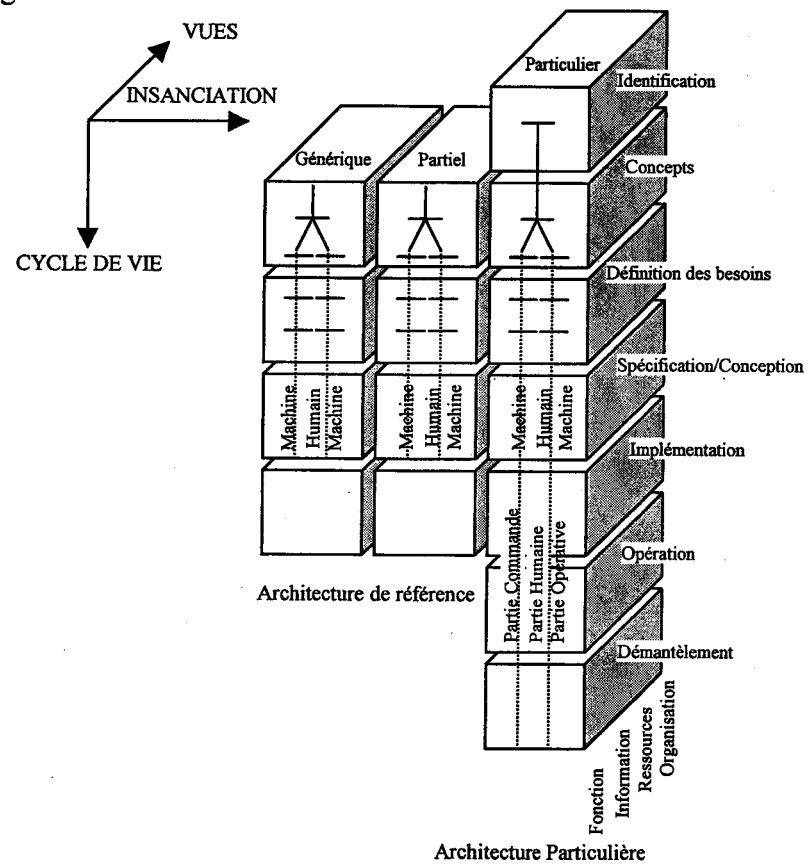

Figure II.5-10- Structure de l'Architecture GERAM

[7] Integrated Enterprise Modelling [Mertins, 93]

En résumé, nous concluons que GERAM reste une coque vide qui n'offre pas son propre langage de modélisation, sauf ceux donnés par les méthodes dont elle est issue.

II.6. CONCLUSION

Les architectures et les méthodes de modélisation sont :

- utiles du fait qu'elles permettent de capitaliser un savoir et un savoir-faire importants,
- essentiels pour aider à comprendre et clarifier l'entreprise et intégrer plusieurs concepts.

Ceci est dans le but d'améliorer les performances de l'entreprise tout en satisfaisant le client.

De plus, les petites et moyennes entreprises industrielles cherchent des modèles de référence et/ou réutilisables pour qu'elles puissent les adapter à leurs propres besoins, en instanciant ces modèles, pour créer leurs modèles particuliers ou des applications.

La plupart de ces architectures offrent des outils, des méthodes et des langages pour mieux analyser et trouver des solutions d'implémentation qui couvrent le cycle de vie entier de l'entreprise.

De par nos préoccupations, nous avons retenu CIMOSA en tant que référence de base même si celle-ci reste de nature académique, car elle se focalise sur les processus métier et les définit d'une façon proche de nos besoins (voir § chapitre I).

En effet, elle a élaboré des concepts qui permettent de diminuer la complexité de l'entreprise ou d'un système pour faciliter la modélisation, que nous jugeons intéressants pour regrouper et gérer les processus métier.

Cette modélisation couvre l'expression des besoins, la conception et la définition de l'implémentation.

Finalement, en se basant sur l'approche CIMOSA, nous proposons un méta-modèle qui permet à l'entreprise de développer ses applications par instanciation ou en se basant sur d'autres applications déjà élaborées; ce qui engendre la capitalisation et la réutilisation du savoir-faire de la modélisation métier sous forme de modèles. En effet, une analyse des objectifs de l'entreprise permet d'élaborer des processus métier en ayant pour but d'achever de tels objectifs, sans pour cela s'éloigner du cadre du projet.

Dans ce contexte, le chapitre suivant traite la façon de capitaliser et réutiliser le savoir-faire.

CHAPITRE III

CADRE DE TRAVAIL – PROJET CAS

CHAPITRE III : CADRE DE TRAVAIL – PROJET CAS

III.1. PRÉSENTATION DU PROJET

Le projet CAS « CIMOSA Application Server » vient de voir le jour lors du lancement de cette thèse, au laboratoire LGIPM[8]. Il a été proposé par M. Spadoni, Maître de conférences.

La motivation principale de ce projet est d'offrir une aide pour la modélisation et la conception des systèmes d'entreprise, en se basant sur des modèles de référence. Ces derniers peuvent représenter un référentiel de modélisation.

Un système peut être défini comme un ensemble de méthodes et de procédés destinés à assurer une fonction définie ou à produire un résultat ; il est donc important d'en définir son objectif et ses concepts. Les concepts de base d'un système sont les entités de l'entreprise, les processus et les rôles des acteurs [Eriksson, 00]. En fait, le flux de contrôle du système est décrit par les processus.

L'objectif d'un système d'entreprise est d'ordre stratégique et concerne avant tout l'amélioration des processus existants. Des enquêtes ont été effectuées qui montrent le besoin pour les entreprises d'améliorer leurs processus. Nous citons en particulier une enquête qui a été réalisée par le cabinet de Benchmark Group [JDNet, 03]. Celle-ci montre que 34% des entreprises essaient de modéliser leurs processus métier avec l'hypothèse qu'une entreprise est décomposable en processus, eux-mêmes décomposables, par niveaux successifs, en sous-processus. A cela, nous pouvons ajouter que l'entreprise elle-même fonctionne comme un processus en interaction avec ses clients, ses fournisseurs, ses concurrents, etc. Ce processus intervenant à son tour dans des processus socio-économiques plus globaux. Par une dynamique permanente d'amélioration, l'entreprise peut espérer à terme améliorer tous les processus. Dans une telle démarche, l'entreprise doit s'intéresser en priorité à quelques processus et de préférence à ceux dont l'amélioration lui apportera le plus de gain. Ce choix de processus « prioritaires » constitue une phase importante lors de la modélisation et ne doit pas être fait au hasard. L'expérience a montré que bien souvent l'observation de dysfonctionnements est à l'origine du déclenchement d'une modification de processus. Lorsque ce type de modification est réalisé au coup par coup il y a fort à parier que l'optimisation du processus sera obtenue après une série de tâtonnements coûteux et dans bien

[8] Laboratoire de Génie Industriel et Production Mécanique EA 3096

des cas démobilisateurs. Afin d'éviter cette situation, il paraît nécessaire de définir et regrouper les processus de telle façon à identifier les processus principaux et secondaires en se basant sur une typologie qui convient à l'entreprise.

Les concepts de base d'un système d'entreprises sont les entités de l'entreprise, les processus et les rôles des acteurs [Eriksson, 00]. En fait, le flux de contrôle du système est décrit par les processus. Ces derniers transforment des entrées en sorties par le biais de ressources. Ces processus sont appelés des processus opérationnels [Vernadat, 96], du fait qu'ils décrivent le comportement opérationnel du système. On les appelle aussi des processus métier. Ils ont un impact direct sur les résultats de l'entreprise, sur son avenir à long terme et la satisfaction des clients. On note qu'ils relèvent d'une problématique de maîtrise d'ouvrage et décrivent le fonctionnement de l'entreprise selon les choix de stratégie et d'organisation. Ils font référence à des activités ou des opérations affectées à des acteurs qui coopèrent selon le respect des choix d'organisation et des règles métier de l'entreprise conforme à la norme ISO 9000-2000 [Mathieu, 00]. Ils constituent les chaînes de valeur de l'entreprise.

Après avoir défini un système d'entreprise, il reste à se préoccuper de sa représentation. Cela consiste avant tout à la mise en place d'une organisation de l'information à travers un système d'information. Du fait qu'un système d'information subit un certain nombre de contraintes de son environnement l'obligeant à réagir en déclenchant des activités, il se doit d'être la représentation la plus fidèle possible de l'organisation du système d'entreprises qu'il représente. Par exemple, le processus "expédition", sous système de l'entreprise, subit une contrainte d'environnement sous forme de l'arrivée d'un bordereau d'expédition du processus "traitement des commandes". Et l'on peut descendre jusqu'au niveau détaillé de la tâche, c'est à dire le programme permettant l'exécution proprement dite de celle-ci.

Au cœur du système d'information se trouve un référentiel, qui dans notre cas regroupe différents composants [Brereton, 00] [Bachman, 00].

Ceux-ci sont distribués et exploités :

- Par des concepteurs d'applications au sein de chaque entreprise. En sortie, nous récupérons des fichiers exécutables qui pourront être partagées par des d'utilisateurs Windows.

- Par des concepteurs de systèmes d'entreprise à travers un framework (Figure III.1-1). En sortie nous récupérons des fichiers sous format xml qui pourront être partagés par des d'utilisateurs internet.

Le système d'information proposé repose sur une architecture logicielle quatre couches (Figure III.1-2) :

- couche 0, dite "**Référentiel CIMOSA**" : celle-ci est représentée par le référentiel utile à la modélisation des processus. Ce référentiel est partagé par l'ensemble de la chaîne d'entreprises. Dans le cadre de nos travaux, nous avons choisi les modèles de processus proposés par CIMOSA. Ainsi, chaque mise à jour de ces modèles bénéficie immédiatement à l'ensemble de la chaîne d'entreprises.
- couche 1, dite « **Systèmes d'entreprise** » : celle-ci est représentée par la bibliothèque de systèmes d'entreprise. Cette bibliothèque est spécifique à une entreprise et son contenu est distribué dans l'entreprise pour le développement d'applications. Dans un contexte d'entreprise étendue, l'ensemble des bibliothèques de systèmes d'entreprise reste accessible à l'ensemble des entreprises de la chaîne. Cet ensemble, que l'on peut nommer "Bibliothèque de chaîne d'entreprises", peut être exploité par des services web.
- couche 2, dite « **Applications** » : celle-ci est représentée par l'ensemble des applications développées dans l'entreprise. Chaque application ainsi conçue est partagée aux différents utilisateurs potentiels.
- couche 3, dite « **Utilisateurs** » : celle-ci est représentée par l'ensemble des utilisateurs d'une même entreprise.

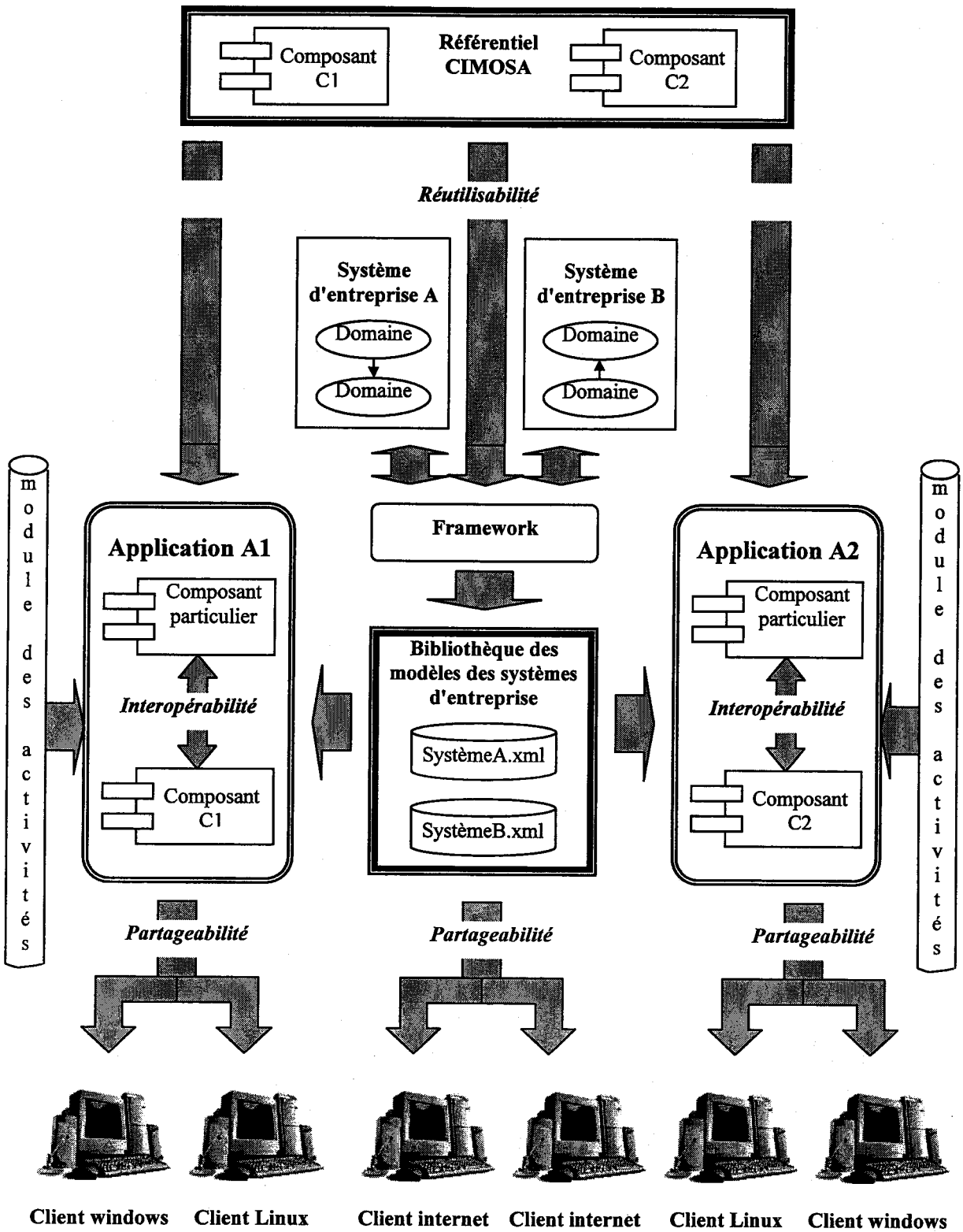

Figure III.1-1 - Présentation du projet C.A.S. (Cimosa Applications Server)

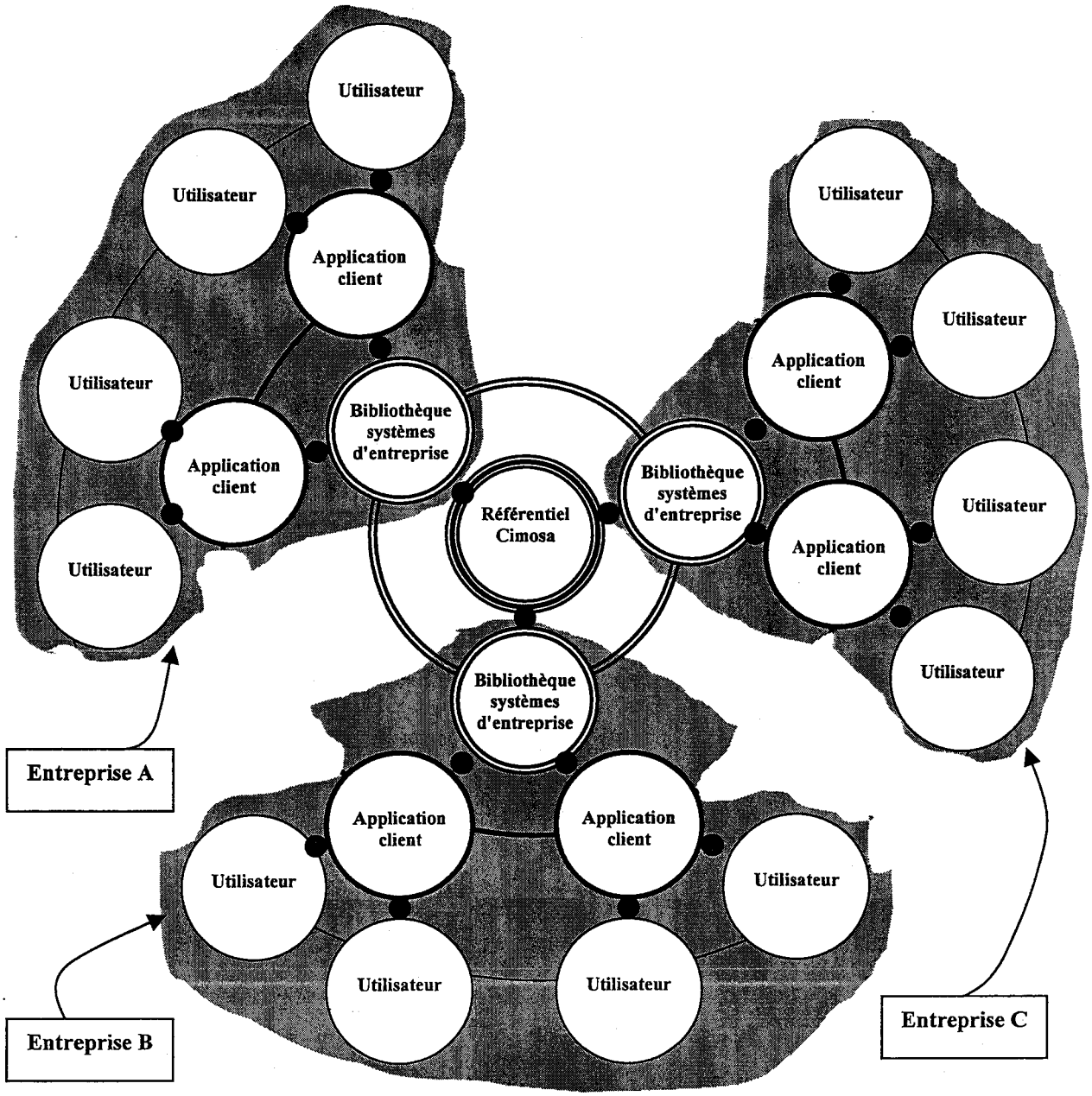

Figure III.1-2 - Architecture du système d'information associé à la modélisation de systèmes d'entreprise dans le cadre d'une chaîne d'entreprises

III.2. LE REFERENTIEL

Est défini dans cette thèse comme un ensemble de modèles de référence pour réaliser d'autres travaux de modélisation.

Pour automatiser un tel référentiel et faciliter l'utilisation des modèles de référence (appelés modèles partiels dans CIMOSA), un développement informatique peut être réalisé. Dans ce cas, les modèles de référence seront implémentés pour devenir des biens logiciels (outil

informatique, composant logiciel, application, etc.). Par conséquent, le référentiel devient une bibliothèque de ces biens logiciels qui peuvent être catalogués et décrits pour permettre une meilleure recherche et utilisation de tels biens (Figure III.2-1).

Pour que la réutilisation soit efficace, les biens logiciels doivent être constitués d'informations réutilisables, mais également celles qui facilitent sa recherche au sein de la bibliothèque.

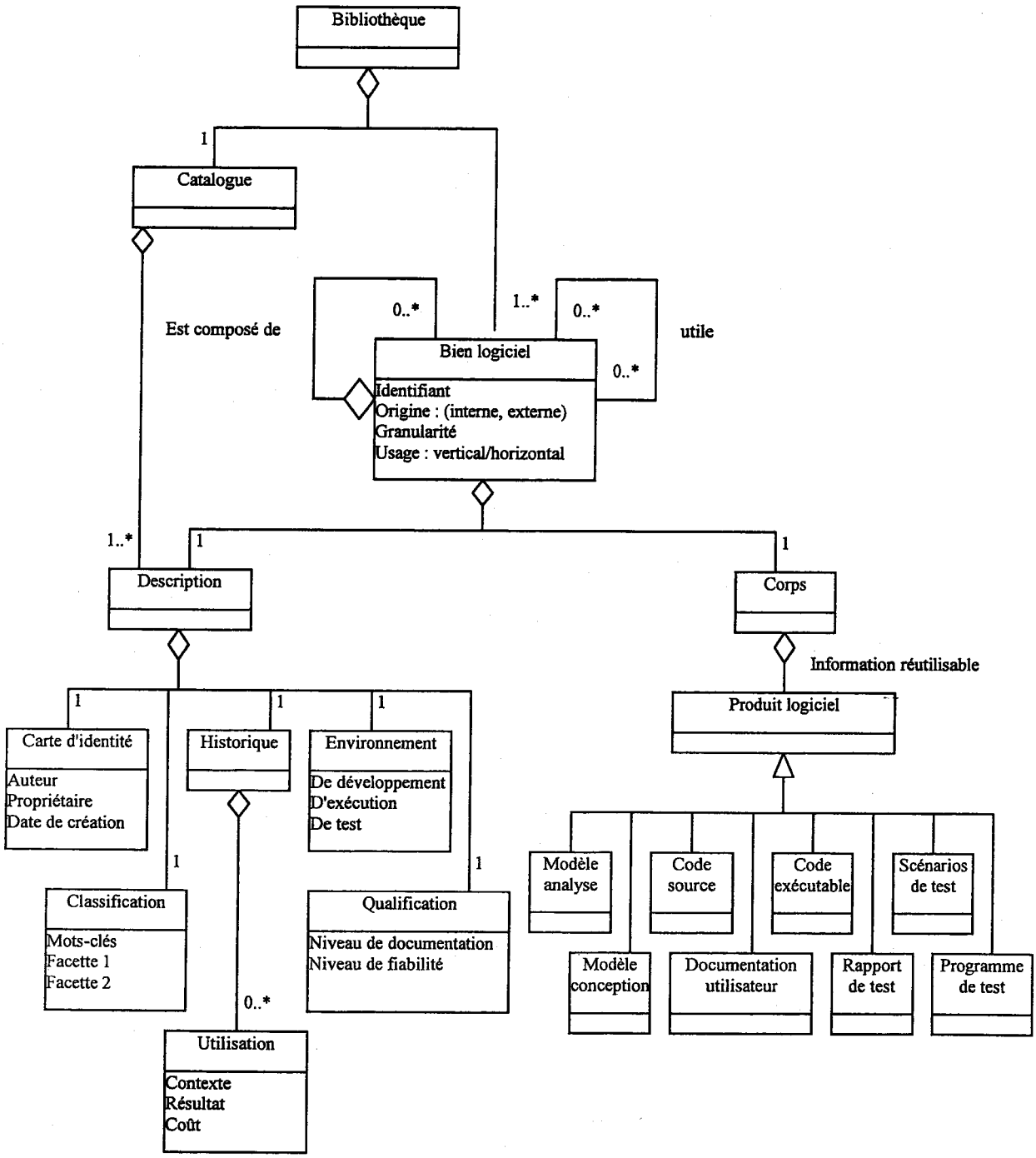

Figure III.2-1- Modèle du référentiel de biens logiciels [Ezran, 99]

L'objectif principal du référentiel dans ce projet est de permettre le développement de modèles particuliers de processus et les distribuer dans l'entreprise de façon qu'ils soient consultables par d'autres applications (par exemple, un système intégré comme un ERP).

Par conséquent, nous avons défini une démarche (Figure III.3-1) qui permet d'établir un modèle de système d'entreprise en se basant sur CIMOSA.

III.3. DESCRIPTION D'UN SYSTEME D'ENTREPRISE

Un système d'entreprise est décrit à partir d'un objectif stratégique par une succession de décompositions. L'approche CIMOSA propose quatre niveaux de décomposition : les domaines, les processus maître, les processus métier et les activités. Jusqu'à présent la notion de processus maître dans CIMOSA n'est pas clairement définie et permet une interprétation large. Nous proposons de les utiliser pour organiser les processus métier. En effet, nous constatons que les processus métier, n'ont que très peu d'influence sur le re-engineering. Par contre, les processus maître nous permettent de regroupement les processus métier et donc de les organiser. En conséquence, les processus maître influencent directement les performances de l'entreprise et le re-engineering. C'est ainsi, que nous proposons une organisation à partir des processus maître dans laquelle la notion d'objectif occupe une place centrale.

Notre démarche s'apparente à une approche globale. En effet, on commence par établir une liste de processus métier concernés par l'objectif du domaine. Ensuite, on établit un regroupement par processus maître et on fixe des objectifs à chacun d'eux. Dans une telle approche, on considère en priorité l'existant et on fait avec, quitte à diversifier les objectifs maître pour respecter l'objectif du domaine. On essaie donc de modifier au minimum l'existant.

En général, les méthodes utilisées pour organiser les processus, adoptent une approche descendante. C'est à dire que l'on va des processus globaux vers la modélisation détaillée de ceux-ci. C'est ce qu'on appelle le déploiement d'objectif [Berio, 01]. Dans une telle approche, les contraintes doivent s'adapter aux objectifs quitte à choisir d'autres processus pour les atteindre. C'est ainsi qu'on identifie dans quelle mesure les processus existants pourront satisfaire les objectifs fixés.

Présentation de la démarche suivant quatre étapes (Figure III.3-1) :

Étape 1 : Après avoir fixé un objectif système (OS), on décompose l'objectif système en sous-objectifs (ou objectifs-domaines) pour définir les domaines du système. On rappelle qu'un domaine est un élément structurant, qui permet de regrouper des processus en un module indépendant. Cela nous permet, en particulier, de s'intéresser à ce qui se passe aux interfaces des différents domaines, en l'occurrence aux informations échangées. Une représentation par domaine ne se veut pas unique ou universelle. Faisons une analogie avec le corps humain. Si l'on s'intéresse au pilotage du corps humain, on effectue un découpage systémique (système sanguin, système musculaire, système digestif, système nerveux, etc.) pour mettre en avant les informations entre le système nerveux et les autres systèmes. Si l'on souhaite mettre en évidence le fonctionnement du corps humain, on effectue un découpage par organes (cœur, foie, rein, etc.). Par contre, il paraît avoir un intérêt très limité de réaliser un découpage physique (tête, tronc, épaule, pied, etc.) car à l'interface d'information qui y circule reste pauvre et n'a pas grand intérêt.

Étape 2 : Pour chaque domaine, nous établissons une liste des différents processus métier (BP) nécessaires à l'élaboration des objectifs-domaines.

Étape 3 : A partir de la liste des processus métier, nous réalisons un ou plusieurs regroupements en processus maître. A chacun d'eux nous associons les objectifs-maîtres et les événements qui s'y rattachent.

L'ensemble des solutions possibles associées à un système d'entreprise est représenté par un graphe OU, qui défini un espace de solutions.

La solution qui répond plus à l'objectif système est retenue par le modélisateur du système.

Étape 4 : La structure du système, correspondant à la solution retenue, est définie. En effet, nous définissons les entrées/sorties, les ressources et flux de contrôle de chaque processus (processus métier et processus maîtres) du système. Nous identifions aussi les activités de chaque processus métier.

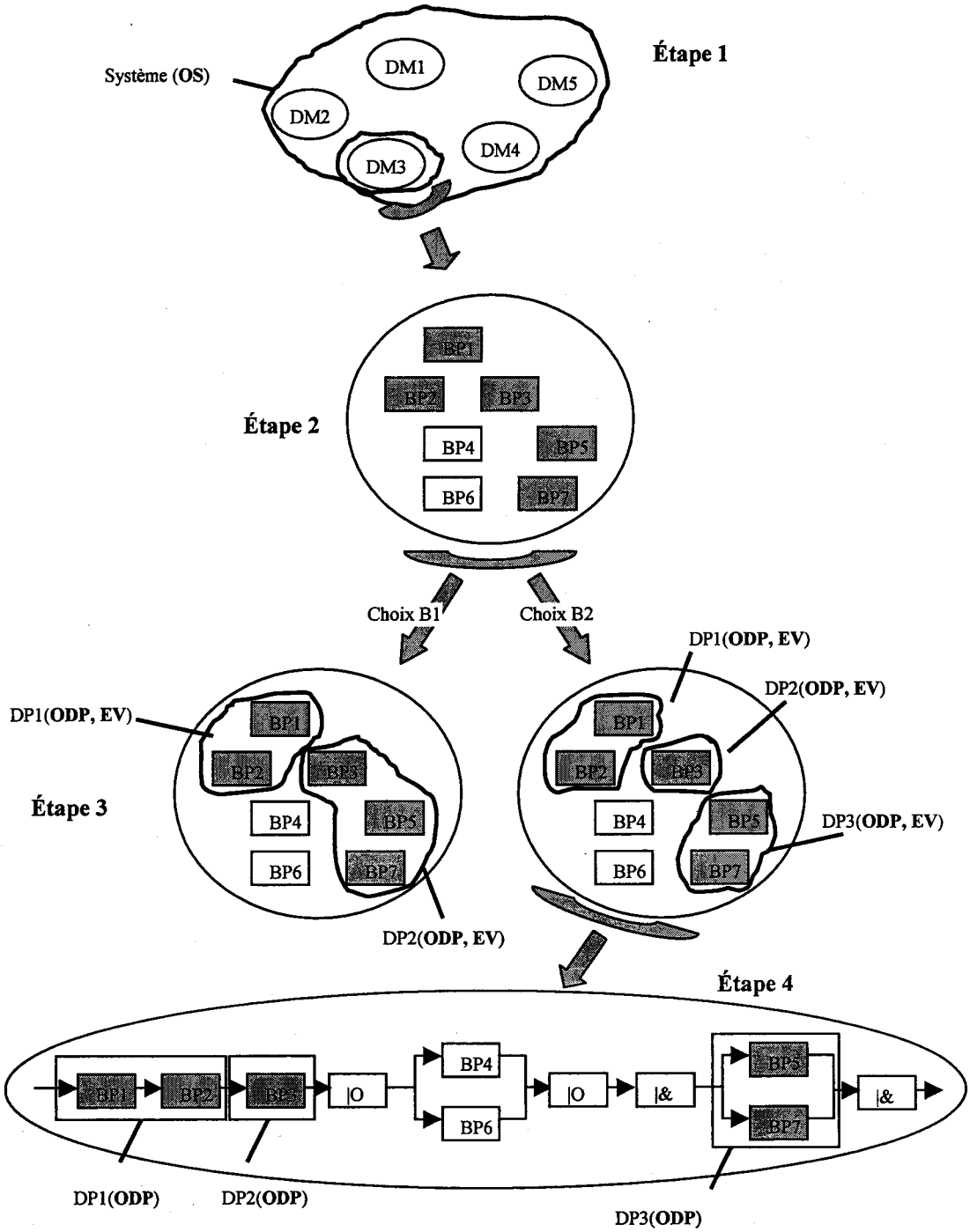

Figure III.3-1- Définition d'un système d'entreprise

III.4. CAPITALISATION D'UN SYSTEME D'ENTREPRISE

Le système d'entreprise ainsi décrit, il reste à lui associer un modèle afin de capitaliser et de manipuler les solutions qui lui sont associées. Pour cela, nous considérons qu'une solution peut être vue comme un plan d'actions, c'est à dire que chaque élément (processus) est associé à une action à exécuter. L'ensemble des solutions associées à un système d'entreprise est représenté par une famille de solutions. La recherche de solution consiste à parcourir une séquence d'actions à travers un plan d'actions. Cela nécessite de faire appel à un moteur de recherche.

Parmi les différents travaux réalisés dans le cadre de modélisation de plan d'actions, on peut citer, entre autres, les approches par réseaux de Petri [Kruth, 92], les graphes ET/OU [Eimaraghy, 92]. Pour ce qui nous concerne, nous retenons le système S.I.R.S. [Spadoni, 01]. Celui-ci modélise une famille de plans d'actions à partir de multigraphes et propose un moteur de recherche capable de gérer et capitaliser une famille de séquences d'actions (Figure III.4-1).

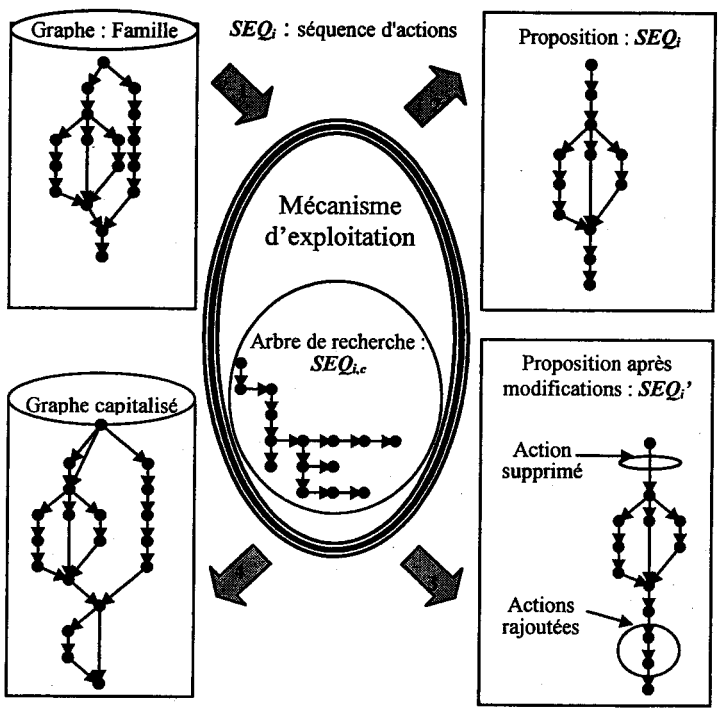

Figure III.4-1 – Moteur de recherche de graphes

Une famille de plans d'actions est implantée par un ensemble de règles (Figure III.4-2) et éditée dans un formalisme adapté [Spadoni, 02]. Cette base de règles peut être développée à partir d'un éditeur de texte ou plus simplement à partir d'une interface adaptée (Figure III.4-3) suite à une proposition (Figure III.4-4).

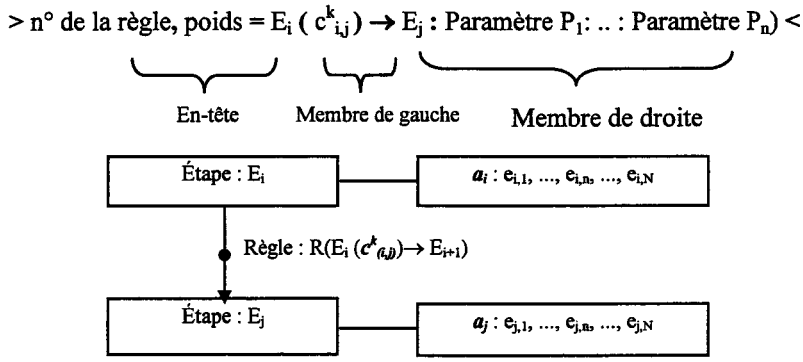

Figure III.4-2 - Syntaxe de règle représentative d'une liaison linéaire entre deux actions

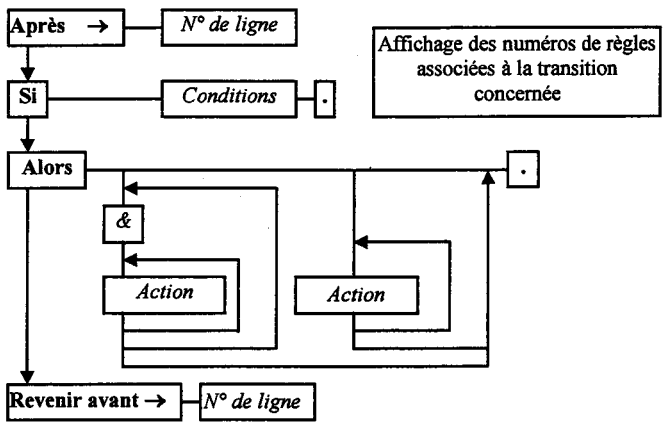

Figure III.4-3 - Modèle de l'interface de modification d'une proposition

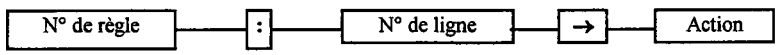

Figure III.4-4 - Syntaxe d'une proposition

Il est nécessaire d'adapter quelque peu le formalisme S.I.R.S. à la représentation d'un système d'entreprise, en particulier la structure des actions (Figure III.4-5).

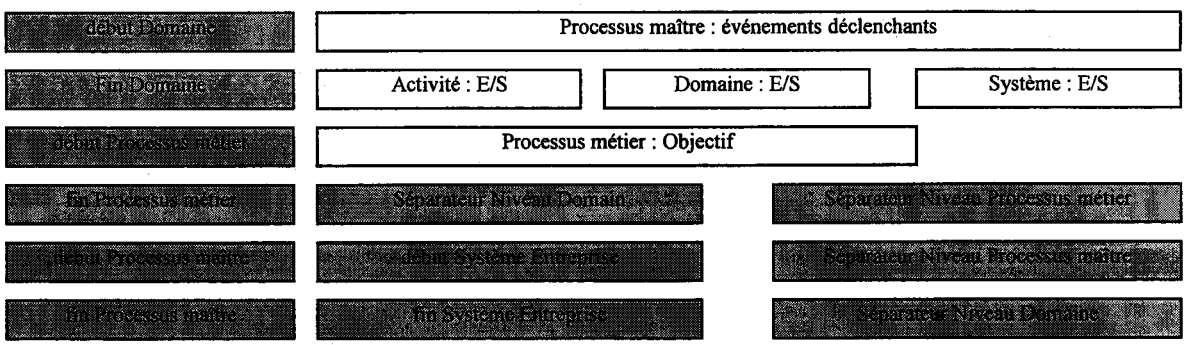

Figure III.4-5 - Récapitulatif des structures d'actions associées à un S.E.

III.5. NOS TRAVAUX DE THESE

Les travaux présentés dans le cadre de cette thèse concernent la conception du référentiel (§ chapitre IV) contenant un modèle de référence pour la modélisation des processus. Nous appelons ce modèle un méta-modèle (§ IV.3). Rappelons que ce dernier se base sur le cadre de modélisation de CIMOSA. Ce méta-modèle est utilisé par des acteurs métier pour développer par instanciation des modèles particuliers pour l'entreprise.

Le méta-modèle est focalisé autour du concept de processus métier tout en permettant aux acteurs d'entreprise de modéliser les processus en partant d'un objectif stratégique de l'entreprise.

Ensuite, les modèles particuliers de processus doivent être capitalisés et stockés d'une façon qu'ils puissent être accessibles par n'importe quelle application. Ceci est dans le but de les distribuer et les réutiliser en garantissant une interopérabilité durable et efficace.

En effet, le méta-modèle est implémenté par des composants (§ IV.4.) qui correspondent aux vues de CIMOSA. Ces derniers sont manipulés au sein d'un framework (§ chapitre V) qui intègre aussi l'application S.I.R.S. dans le but de capitaliser les modèles de systèmes d'entreprise pour mieux les réutiliser.

CHAPITRE IV

CONCEPTION D'UN REFERENTIEL POUR LA MODELISATION DES PROCESSUS D'ENTREPRISE

CHAPITRE IV : CONCEPTION D'UN REFERENTIEL POUR LA MODELISATION DES PROCESSUS D'ENTREPRISE

IV.1. INTRODUCTION

Un référentiel est un espace informatique où sont stockés des modèles de référence (génériques, partiels ou complets).

Les informations qu'il contient peuvent être partagées et réutilisées par plusieurs acteurs connectés au référentiel. En effet, chaque acteur peut récupérer des informations ou des modèles dont il a besoin pour les utiliser dans son environnement, en manipulant des applications et des outils existants. Cela signifie qu'un référentiel doit permettre la distribution des applications existantes dans l'entreprise.

Dans cette thèse, nous proposons un référentiel pour aider l'entreprise à développer des modèles particuliers, sur la base de son savoir et de son savoir-faire. Ces modèles se doivent d'être partageables entre les acteurs métier de l'entreprise et interopérables [Workflow, 96] par rapport aux applications devant les accueillir.

Ce référentiel joue le rôle d'une base de modèles génériques ou partiels de processus que nous avons appelé référentiel de composants de modélisation de l'entreprise (EMC *Enterprise Modeling Components*) [Abdmouleh, 02]. Ceux-ci permettent à l'entreprise de développer leurs propres modèles particuliers de processus de systèmes d'entreprise et les placer dans un référentiel de système d'entreprise (Figure IV.1-1).

Le référentiel se base sur un méta-modèle qui se base principalement sur l'approche CIMOSA et aide les acteurs métier à développer des modèles particuliers de processus métier par instanciation (Figure IV.1-1) [Abdmouleh, 00] [Abdmouleh, 02] [Abdmouleh, 03].

Pour implémenter ce méta-modèle, nous avons opté pour la technologie objet et plus particulièrement celle des composants [Brereton, 00] [Bachman, 00] (Figure IV.1-1).

Les modèles particuliers développés sont évalués et utilisés par des utilisateurs tels que des concepteurs, des techniciens, des ingénieurs, du personnel administratif, etc.

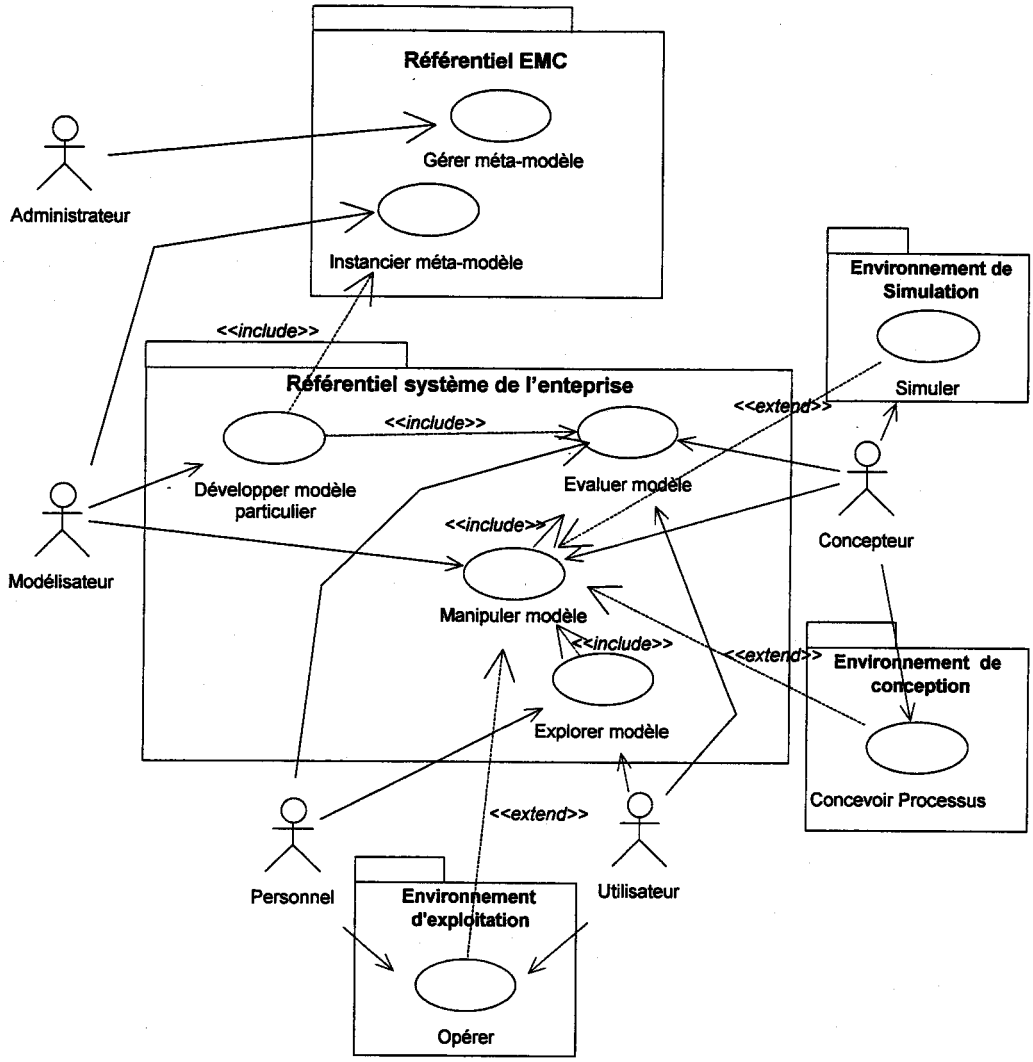

Figure IV.1-1- Interaction des acteurs avec le référentiel EMC

Dans la section suivante, nous spécifions les acteurs métier qui contribuent au développement de modèles de l'entreprise en utilisant le référentiel. Ces acteurs sont les groupes d'administrateurs et les groupes de modélisateurs.

Ensuite, nous définissons les composants métier qui implémentent le méta-modèle basé sur l'approche CIMOSA, les services qu'ils offrent et leur utilisation par les acteurs métier.

IV.2. LES ACTEURS METIER DE L'ENTREPRISE

IV.2.1. INTRODUCTION

Nous nous intéressons aux différents acteurs existants dans l'entreprise et qui peuvent, d'après nous, intervenir lors du développement de modèles particuliers de processus métier.

Les acteurs métier travaillent suivant les principes de l'ingénierie simultanée pour pouvoir créer ces modèles, en utilisant le méta-modèle du référentiel.

IV.2.2. DEFINITION

« Un acteur est un individu, un groupe ou une institution, considéré du point de vue des rôles qui lui sont impartis dans la société (dans notre cas, c'est l'entreprise). C'est aussi une personne qui joue un rôle dans une action. » [Hachette, 01]

En adoptant cette définition, nous pouvons dire que l'acteur est un élément essentiel pour la réalisation d'une action. Mais, il ne peut intervenir que si un rôle lui a été attribué en fonction de ses aptitudes (compétences/capacités).

Un acteur est quelqu'un ou quelque chose d'autonome, tel qu'une machine, une application informatique ou une personne. Il peut avoir un ou plusieurs rôles simultanément[9]. Par exemple, une personne peut jouer le rôle d'un standardiste et d'un réceptionniste en même temps ; alors que ce rôle peut être attribué à d'autres acteurs.

Il est possible aussi qu'un acteur ait plusieurs rôles, mais il ne peut les exécuter tous simultanément. Par exemple, une personne qui a le rôle de gestionnaire, pour la confirmation des factures arrivées, peut aussi avoir le rôle de réceptioniste de factures.

IV.2.3. ACTEURS METIER

Les acteurs métier qui contribuent au développement d'un modèle métier, peuvent appartenir à l'environnement d'ingénierie ou à l'environnement d'exploitation de l'entreprise [AMICE, 93].

Ils ont pour objectif de modéliser tout ou partie de l'entreprise, au sein d'une approche d'ingénierie simultanée [Jagou, 93].

En partant de la définition qu'une entreprise peut être considérée comme un ensemble d'au moins un processus, un système peut être aussi un réseau de processus communiquant entre eux par le biais d'événements et de résultats.

Par conséquent, les acteurs métier collaborent pour modéliser un système de l'entreprise (Figure IV.2-2), en se basant sur une méthodologie qui permet de les orienter et aussi de développer des modèles cohérents et efficaces. La modélisation d'un système d'entreprise est

[9] Voir Annexe B

réalisée en se basant sur le savoir et le savoir-faire des experts en modélisation des processus métier.

Cette méthodologie est intégrée dans le méta-modèle que nous proposons, basée principalement sur les concepts de CIMOSA. Ces systèmes sont ainsi appelés : Système d'entreprise (Figure IV.2-2).

La Figure IV.2-2 représente le cas d'utilisation « Modéliser un sysème d'entreprise » qui présente l'ensemble des cas d'utilisation « Instancier Méta-modèle » et « Développer modèle particulier » de la Figure IV.1-1.

Dans cette thèse, nous nous sommes limités à un nombre d'acteurs métier dans les deux environnements, en définissant des acteurs humains (Responsables métier, d'organisation, de cellule, experts, etc.) et des acteurs logiciels (ERP, SGDT/PDM). En effet, les Responsables métier, les Responsables d'organisation, les Responsables de Cellules, les Experts et les Techniciens forment les groupes de modélisateurs (Figure IV.1-1). Alors que les Responsables de système d'entreprise sont les groupes d'administrateurs du référentiel EMC (Figure IV.1-1). Dans la figure nous représentons cette spécialisation de groupes d'acteurs en utilisant la notion d'héritage comme c'est défini dans UML [Roques, 00] [Booch, 00] [Rumbaugh, 94].

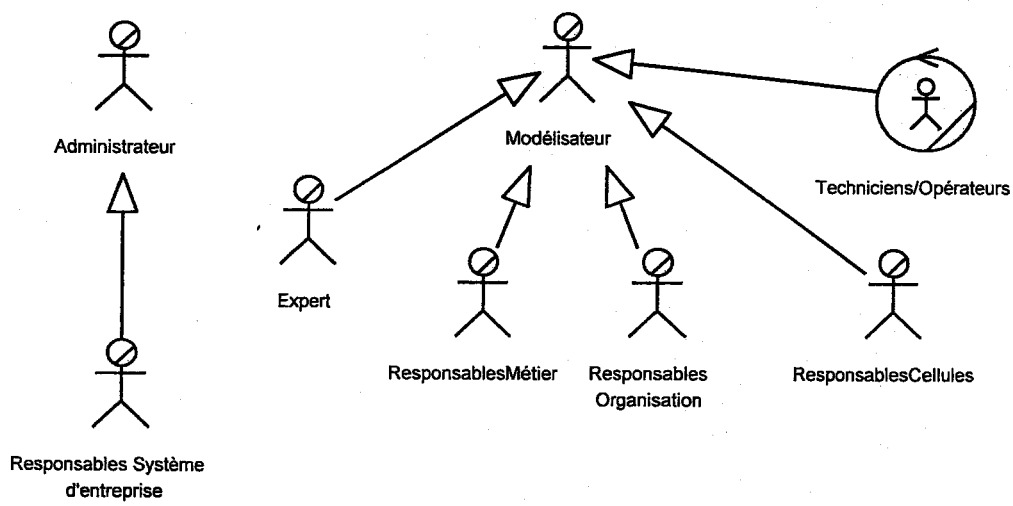

Figure IV.2-1 – Spécialisation des groupes d'acteurs

D'autres acteurs peuvent être rajoutés, comme des entreprises sous-traitantes et/ou partenaires dans le cas où l'entreprise partage son activité de modélisation avec d'autres.

Remarques :

- Le manager de l'entreprise peut être considéré en tant que responsable du Système d'entreprise, ou tout simplement en tant que responsable métier.
- Un chef service est normalement considéré en tant que responsable de cellule.
- Un ingénieur peut être un responsable métier, d'organisation, un expert ou même un responsable du Système d'entreprise.

Les acteurs métier peuvent être classés en deux catégories, comme indiqué dans [Abdmouleh, 03] :

- Acteurs métier d'ingénierie qui appartiennent à l'environnement d'ingénierie de l'entreprise, tels que le responsable du Système d'entreprise, les experts, les responsables métier, les responsables de cellules, etc.
- Acteurs métier d'exploitation qui appartiennent à l'environnement d'exploitation de l'entreprise, tels que les techniciens et les opérateurs.

Les acteurs métier d'ingénierie définissent et développent des méthodes, des modèles et des applications qui seront exploités et utilisés par les acteurs métier d'exploitation. Ils oeuvrent dans les bureaux d'études et des méthodes.

Il est donc important que les acteurs métier d'exploitation, qui agissent dans le système opérationnel, participent à la modélisation pour éclaircir les différents problèmes d'exploitation et intègrent leurs savoir-faire « de terrain » dans les modèles.

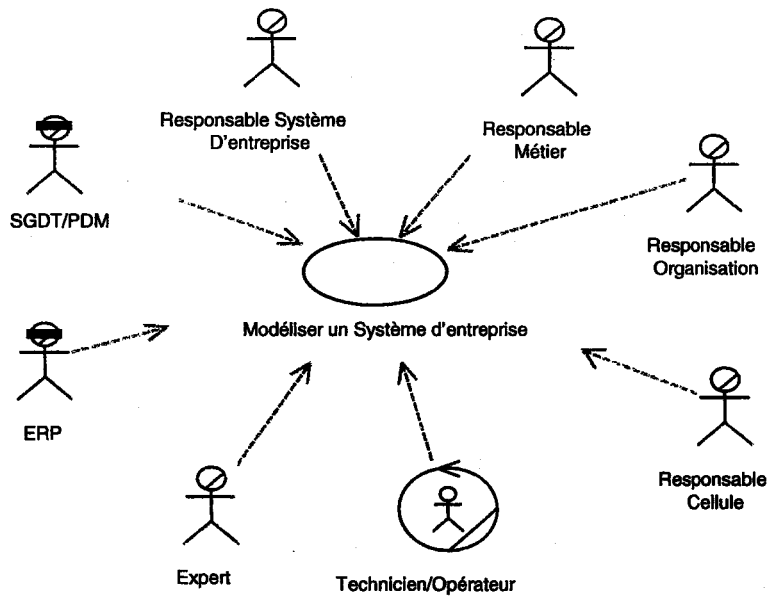

Figure IV.2-2- Acteurs métier contribuant à la modélisation d'un Système d'entreprise

Responsable Métier	Il s'occupe de définir les processus et leurs objectifs.	• Nom • Identifiant • Cellule • Rôles au sein de l'équipe [1..*] • Aptitudes [1..*] • Eléments manipulés [1..*]	Environnement d'ingénierie de l'entreprise
Responsable Organisation	Il définit la structure d'organisation. Il peut même attribuer à des acteurs, les rôles de responsables de cellules.	• Nom • Identifiant • Cellule • Rôles au sein de l'équipe [1..*] • Aptitudes [1..*] • Eléments manipulés [1..*]	
Responsable Cellule	Il distribue les responsabilités au sein de sa cellule.	• Nom • Identifiant • Cellule • Aptitudes [1..*] • Eléments manipulés [1..*]	
ERP	Permet d'avoir des informations à propos des ressources et des objets de l'entreprise.	• Port d'entrée d'information • Port de sortie d'information	
SGDT/PDM	Permet d'avoir les données techniques à propos des objets techniques et des produits dans l'entreprise.	• Port d'entrée de données techniques • Port de sortie de données techniques	
Expert	• Il contribue par son savoir et faire à la définition des étape activités et du comportement processus. • Il contribue à définir et à vali entrées/sorties des activités r dans la cellule.	• Nom • Identifiant • Cellule • Rôles au sein de l'équipe [1..*] • Aptitudes [1..*] • Eléments manipulés [1..*]	
Technicien/Opérateur	Il contribue par son savoir-faire en tant qu'agent de terrain, à décrire les problèmes rencontrés et à donner ses propositions et solutions.	• Nom • Identifiant • Cellule • Rôles au sein de l'équipe [1..*] • Aptitudes [1..*] • Eléments manipulés [1..*]	Environnement d'exploitation de l'entreprise

Tableau IV.2-1- Tâches génériques de chaque acteur métier

Un groupe de responsables du Système d'entreprise doit contribuer à toutes les étapes du cycle de vie du développement du système. Cela permet de guider les autres groupes d'acteurs dans l'application de l'approche CIMOSA.

Comme tout avant-projet, un cahier des charges fonctionnel fixant les besoins et la définition d'une stratégie d'entreprise est obligatoire.

Ensuite, nous définissons des objectifs en partant de la stratégie et des besoins de l'entreprise. Ces objectifs permettent aux responsables métier de mieux définir le système et ses processus métier en particulier, en développant une cartographie.

L'étape suivante consiste à associer des centres de décision ou des cellules d'organisation pour définir les responsabilités sur chaque processus. Le responsable cellule distribue les responsabilités au sein de sa cellule en fonction des rôles et des aptitudes de chaque acteur. Ces derniers définissent, pour chaque processus :

- Les entrées et les sorties comme indiquées dans les objectifs du processus considéré,
- Les ressources techniques et les ressources humaines requises.

Après l'étape de définition de la cartographie des processus, il est indispensable de définir les étapes de chaque processus pour définir le comportement et les fonctionnalités de l'entreprise. Ces fonctionnalités, qui sont décrites par les activités, sont définies par des experts et par des agents de terrain (techniciens et opérateurs). Les activités peuvent être décomposées, d'après CIMOSA, en opérations fonctionnelles. Ces dernières représentent le niveau de granularité en termes de fonctionnalité le plus fin du système. Elles sont réalisées par des entités fonctionnelles (humaines, machines, automates ou applications), qui sont des ressources actives, c'est-à-dire douées de capacités d'action ou de décision [AMICE, 93].

Après cette décomposition fonctionnelle, la cartographie des processus doit être mise à jour en fonction de la spécification des ressources et des entrées/sorties au niveau des activités.

La Figure IV.2-3 résume les tâches génériques expliquées ci-dessus.

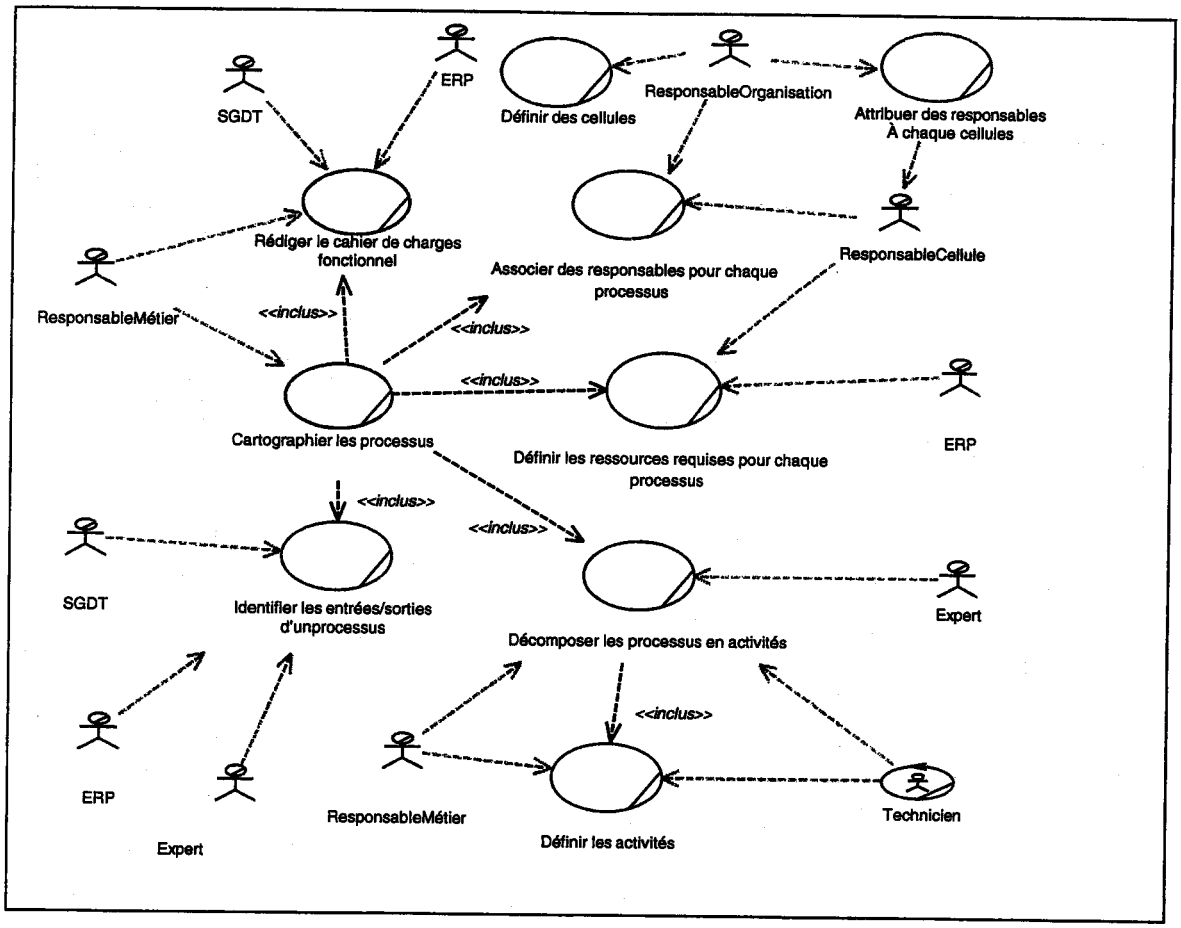

Figure IV.2-3- Cas d'utilisation de chaque acteur métier

Dans la section suivante, nous décrivons le méta-modèle que les acteurs métier utilisent pour développer un modèle de leur système, tout en réalisant leurs tâches respectives (Figure IV.2-3). Ce méta-modèle englobe les concepts permettant la définition de processus métier et des différentes entités de l'entreprise qui peuvent être en interaction avec de tels processus.

IV.3. META-MODELE

IV.3.1. INTRODUCTION

Un des objectifs de cette thèse est d'offrir aux acteurs métier une aide pour la modélisation des processus métier d'un système d'entreprise.

En se basant sur les architectures de modélisation en entreprise, les normes ISO [ISO, 02] et CEN (ENV 40003 et 12204) [Shorter, 00][CEN, 90], nous avons conçu un modèle conceptuel qui permet de développer des modèles spécifiques, par instanciation, dans différents environnements de l'entreprise.

Par conséquent, ce modèle conceptuel joue le rôle d'un méta-modèle, car c'est un modèle de référence pour le développement de modèles de processus par instanciation.

Ce méta-modèle a été enrichi en termes de le savoir-faire de conception métier des processus par le biais des patrons métier [Eriksson, 00].

Dans cette section, nous présentons et décrivons le méta-modèle. Nous proposons quatre aspects (fonctionnel, informationnel, de ressource, organisationnel) conformément aux quatres vues préconisées par CIMOSA [AMICE, 93] et ISO/CEN [CEN, 90].

IV.3.2. ASPECTS DU META-MODELE

En partant de notre définition du processus (§II.2.1), nous pouvons identifier et caractériser les concepts qui permettent de définir un processus métier.

a) Aspects fonctionnels

Nous traitons les aspects fonctionnels du méta-modèle en réutilisant les concepts (constructs) de la vue fonction de CIMOSA [CIMOSA, 94]. Nous représentons ces constructs par une ou plusieurs classes d'objets et les évaluons.

Les différentes définitions sont présentées au moyen de tableaux dans lesquels sont utilisés des abréviations. Celles-ci sont explicitées dans le Tableau IV.3-1 :

OBM	***OBjectif Métier***	*D*	***Durée***
SBP	*Sous-processus métier (**Sub-Business Process**)*	*CRp*	*Cellule d'organisation **Responsable***
EF	***Entrée Fonctionnelle***	*CAtO*	*Cellule d'organisation ayant l'**Autoritaire Opérationnelle***
SF	***Sortie Fonctionnelle***	*CAtC*	*Cellule d'organisation ayant l'**Autorité de Conception***
EC	***Entrée Contrôle***	*Rp*	***Responsable***
SC	***Sortie Contrôle***	*AtO*	***Autorité opérationnelle***
RS	***ReSsource***	*AtC*	***Autorité de conception***

Tableau IV.3-1- Abréviations

a.i) Processus métier

Le concept principal de cet aspect est celui de processus métier. Nous le représentons par la classe d'objet nommé 'Processus Métier' (BP : *Business Process).*

D'après la définition que nous avons proposée, celui-ci reçoit des objets en entrée et leur ajoute de la valeur, par le moyen de ressources, tout en fournissant des objets de sortie (produits/services) remplissant les besoins et les exigences d'un client (atteindre les objectifs) interne ou externe à l'entreprise. Il ne peut être déclenché que par des événements internes et/ou externes à l'entreprise. Chaque processus est en communication avec d'autres processus et peut être décomposé en sous-processus et en activités. Une activité transforme une entrée en sortie sous l'influence d'objets de contrôle en utilisant les ressources requises et disponibles pendant une durée bien définie.

Donc, un processus métier peut être défini par ses ***étapes*** (activités) (§(2) et §a.ii), son ***comportement*** (§(3) et §a.iii), son ***environnement*** d'application (domaines) (§(1), ses ***entrées et sorties***, et les ***ressources*** requises pour atteindre ses ***objectifs*** (Figure II.2-3).

(1) Environnement d'un processus métier

CIMOSA offre le concept de 'Domaine' (DM) pour appréhender la complexité de l'entreprise ou du système et permet le regroupement des processus métier en zones fonctionnelles. CIMOSA fournit aussi le concept de 'processus maître' (DP : *Domain Process*) qui est considéré en tant que processus métier de plus haut niveau dans la décomposition des processus métier d'un domaine.

Nous utilisons le concept de processus maître en tant que moyen logique pour regrouper des processus métier en tenant compte des objectifs métier à un niveau global.

Un domaine rassemble donc les processus métier regroupés en processus maîtres. Dans notre méta-modèle, les concepts 'Domaine' et 'processus maître' sont représentés par des classes d'objets. Par conséquent, la classe 'processus maître' est une composition de la classe 'processus métier'. La classe 'domaine' est une composition de la classe 'processus métier' par le biais de la classe 'processus maître' (Figure IV.3-1).

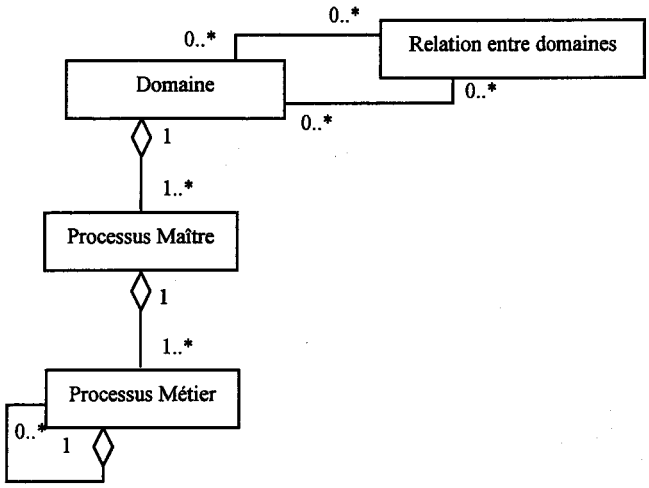

Figure IV.3-1 – Regroupement des processus métier

(2) Étapes d'un processus métier

La structure d'un processus est définie par ses étapes, qui peuvent être des sous-processus et/ou des activités [Marchall, 99].

Dans notre méta-modèle, nous avons défini les étapes d'un processus métier en tant qu'activités d'entreprise (EA : *Enterprise Activity*) et/ou processus métier. Donc un processus métier est un ensemble non vide d'activités interconnectées et un ensemble de processus métier. Nous pouvons noter cela comme suit :

$$BP_i=\{EA_{i,j}\} \cup \{SBP_{i,k}\} \text{ avec } \{EA_{i,j}\} \cup \{SBP_{i,k}\} \neq \varnothing$$

Tels que

BP_i : un processus (i)

$EA_{i,j}$: les activités EA_j appartenant au processus BP_i

$SBP_{i,k}$: sous-processus métier SBP_k appartenant au processus BP_i

avec $1 \leq i \leq n$; $1 \leq j \leq m$; $0 \leq k \leq n$

n : nombre de processus métier existants dans le système

m : nombre d' activités existantes dans le système

(3) Comportement d'un processus métier

Le déclenchement d'un processus métier est défini par le comportement de son processus parent. Celui-ci peut être un processus maître ou un processus métier.

Un processus maître est déclenché par des événements internes ou externes à son domaine (Figure IV.3-2).

Le comportement d'un processus maître ou d'un processus métier est défini par des ***règles de comportement*** (BR : *Behavior Rules*) qui vérifient et valident le lancement d'autres processus et/ou des activités.

Une règle de comportement est de la forme canonique suivante:

SI [conditions] ALORS [actions]

Le déclenchement d'un processus maître est lui aussi défini par une règle de comportement qui déclenche des actions (processus/activité) après vérification de l'existance d'un ensemble d'événements. Ceci est de la forme de :

SI [DEBUT AVEC {Evénement(i)}] ALORS [actions]

Il peut être déclenché au moyen d'une conjonction de plusieurs événements (liés par l'opérateur logique ET).

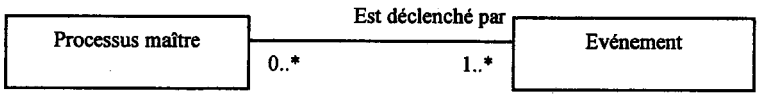

Figure IV.3-2 – Déclenchement d'un processus maître

Un processus métier peut être déclenché directement après le lancement du processus maître parent. Ceci peut être décrit par la syntaxe suivante :

SI [DEBUT] ALORS [actions]

Remarque :

- *Les règles de comportement sont décrites en détail par Berio et Vernadat [Berio, 01].*
- *Le concept d'événement est représenté par une classe d'objet Événement. Celle-ci a pour attributs ceux proposés par CIMOSA.*

(4) Caractéristiques d'un Processus Métier

Un processus métier peut être caractérisé par :

- Un nom,
- Un identifiant (une référence),
- Un objectif métier,
- La liste de ses sous-processus métier,
- La liste de ses activités,
- La liste de ses états de transition,
- Le processus parent (processus maître ou processus métier en cas de sous-processus métier),
- La liste et l'enchaînement des règles de comportement,
- La liste des entrées fonctionnelles,
- La liste des sorties fonctionnelles,
- La liste des entrées de contrôle,
- La liste des sorties de contrôle,
- La liste des ressources,
- Le coût,

La liste des états de transition ou états de fin (ES : *Ending Status*) [CIMOSA, 94] permet de décrire les états à la fin du processus métier. Ceux-ci peuvent être ***un abandon, une attente, une suspension et la fin d'exécution*** du processus [Marchall, 99]. Ces états peuvent être représentés graphiquement par un diagramme d'états-transitions (state-chart ou Réseau de Petri) [Booch, 00].

Par conséquent, la classe du Processus Métier peut avoir les attributs suivants :

Attributs de la classe Processus Métier		**Caractéristiques d'un processus métier**
Attribut	***Type***	
Nom	Chaîne de caractères	Nom
$BP_{i,j,k}$	Chaîne de caractères	Identifiant
OBM	Objet	Objectif métier
DP	Objet	Processus parent
SBP_i	[0,n] Objet	Liste des sous-processus métier
EA_i	[1,n] Objet	Liste des activités
ES_i	[1,n] Chaîne de caractères	Liste des états de transition
BR_i	[1,n] Objet	Liste et enchaînement des règles de comportement

EF_i	[0,n] Objet	Liste des entrées fonctionnelles	L'union ne peut être vide
SF_i	[0,n] Objet	Liste des sorties fonctionnelles	
EC_i	[0,n] Objet	Liste des entrées de contrôle	
SC_i	[0,n] Objet	Liste des sorties de contrôle	
RS_i	[1,n] Objet	Liste des ressources	
Coût	Nombre	Le coût du processus métier	

Ici, le coût d'un processus est défini comme la somme des coûts de ses étapes (activités et/ou sous-processus), des activités de maintenance (entretiens, interventions, etc.), des transitions entre les étapes et les temps masqués (TM). On le note ainsi :

$$\text{Coût Processus} = \sum_{i=1}^{n} CoûtEA_i + \sum_{j=1}^{m} CoûtSBP_j + \sum_{k=1}^{p} CoûtActivitéMaintenance_k + \sum_{l=1}^{q} CoûtTransition_l + \sum_{h=1}^{r} CoûtTM_h$$

(5) Méthodes de la classe Processus Métier

Les méthodes de la classe Processus métier sont les différentes opérations/fonctions qu'une occurrence Processus Métier peut exécuter (ne pas confondre avec les activités).

Ces méthodes peuvent permettre :

- de décrire le développement des règles de comportement du processus métier : *DevComportement(BR)*,
- de décrire l'exécution des règles de comportement en tenant compte de l'enchaînement défini : *Exécution(BR_i)*,
- d'ajouter des activités : AjouterActivité($EA_{i,j,k,l}$),
- d'ajouter des sous-processus métier : *AjouterProcessus(SBP_i)*.

On rappelle que les règles de comportement permettent de décrire la façon dont s'exécute un du processus (lancement des activités et des sous-processus) et d'identifier ainsi son début et sa fin.

a.ii) Activité d'entreprise

Une activité d'entreprise est définie en tant que tâche réalisée dans l'entreprise au moyen de ressources, au cours du temps, pour réaliser un objectif. Elle consiste à transformer les états

d'objets d'entrée en états d'objets de sortie sous certaines contraintes, et produit ainsi des informations et des événements.

En se basant sur les approches CIMOSA et SADT/IDEF0, une activité peut admettre deux types d'entrée :

- les entrées fonctionnelles définies par les états des objets à transformer (exemple : pièce brute, pièce semi-finie),
- les entrées de contrôle définies par des états d'objets et/ou des informations (exemple : données, mesures).

De même, nous pouvons avoir deux types de sorties : les sorties fonctionnelles et les sorties de contrôle.

D'après CIMOSA, l'exécution d'une activité est réalisée par des opérations fonctionnelles, qui représentent la plus petite unité fonctionnelle dans l'entreprise. Ces opérations fonctionnelles sont des actions élémentaires pouvant être exécutées par des entités fonctionnelles (ou ressources actives c'est-à-dire instrumentées). Une opération peut être 'entrer_pièce' ou 'sortir_pièce' pour un stock ou, 'lire_données' ou 'insérer_données' dans un serveur de données.

Nous utilisons la forme canonique donnée par CIMOSA :

EF.OF(liste de paramètres)

Avec EF : nom de l'entité fonctionnelle *(functional entity)*

OF : nom de l'opération fonctionnelle *(functional operation)*

Pour associer les opérations fonctionnelles d'une activité aux entités fonctionnelles qui sont les ressources validant les aptitudes de l'activité, on peut le récapituler dans un tableau. Par exemple, soit l'activité A, qui nécessite les opérations fonctionnelles OF1, OF2 et OF3. Celles-ci sont exécutées par quatre entités fonctionnelles EF1, EF2, EF3 et EF4, telles que EF1 et EF2 exécutent OF2, EF3 exécute OF1 et EF4 exécute OF3. Cela peut être représenté par le Tableau IV.3-2 :

EF.OF	OF1	OF2	OF3
EF1		Paramètres2	
EF2		Paramètres3	
EF3	Paramètres1		
EF4			Paramètres4

Tableau IV.3-2 – Opérations fonctionnelles et entités fonctionnelles

Les paramètres déterminent les vues d'objet VO_k qu'il faut passer à l'entité opérationnelle EF_j pour exécuter l'opération fonctionnelle OF_i

Alors, nous pouvons exprimer la relation $EF_j.OF_i$ comme suit :

$EF_j.OF_i=\{VO_k\}$; $1 \leq k \leq n$

Telle que n est le nombre de vues d'objet dans une relation $EF_j.OF_i$.

Dans notre méta-modèle, une activité et une opération fonctionnelle sont respectivement représentées par une classe d'objet (Figure IV.3-3).

Nous pouvons exprimer une activité EA_i en fonction d'une opération fonctionnelle $FO_{i,j}$.

$$EA_i=\{FO_{i,j}\} ,$$

Avec

EA_i : une activité (i)

$FO_{i,j}$: les opérations fonctionnelles FO_j appartenant à l'activité EA_i

Et $1 \leq i \leq n$; $1 \leq j \leq m$

n : nombre d'activités existantes dans le système

m : nombre d'opérations fonctionnelles existantes dans le système

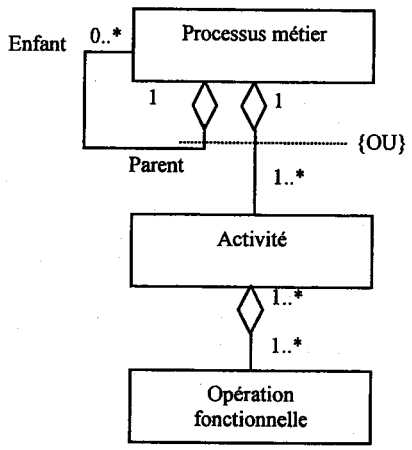

Figure IV.3-3 – Décomposition fonctionnelle d'un processus métier

Remarque :

Les paramètres des opérations fonctionnelles sont définis par une classe d'objet Paramètre.

Une activité d'entreprise est caractérisée par les attributs suivants :

- Un nom,
- Une identification (référence),
- Les entrées/sorties fonctionnelles et de contrôle,
- Fonctionnement de l'activité,
- Les états de transition de l'activité,
- L'ensemble des aptitudes requises,
- La durée,
- Le coût,
- L'objectif,

Par conséquent, les attributs de la classe 'Activité' sont :

<table>
<tr><th colspan="2">Attributs de la classe Activité</th><th colspan="2" rowspan="2">Caractéristiques d'une activité</th></tr>
<tr><th>Attribut</th><th>Type</th></tr>
<tr><td>Nom</td><td>Chaîne de caractères</td><td colspan="2">Nom</td></tr>
<tr><td>$EA_{i,j,k,l}$</td><td>Chaîne de caractères</td><td colspan="2">Identifiant</td></tr>
<tr><td>OBA</td><td>Objet</td><td colspan="2">Objectif de l'activité</td></tr>
<tr><td>$BP_{i,j,k}$</td><td>Objet</td><td colspan="2">Processus parent</td></tr>
<tr><td>OF_i</td><td>[1,n] Objet</td><td colspan="2">Liste des opérations fonctionnelles employées par l'activité</td></tr>
<tr><td>ES_i</td><td>[1,n] Chaîne de caractères</td><td colspan="2">Liste des états de transition</td></tr>
<tr><td>BR_i</td><td>[1,n] Objet</td><td colspan="2">ceci peut être un algorithme</td></tr>
<tr><td>EF_i</td><td>[0,n] Objet</td><td>Liste des entrées fonctionnelles</td><td rowspan="3">L'union est non vide</td></tr>
<tr><td>SF_i</td><td>[0,n] Objet</td><td>Liste des sorties fonctionnelles</td></tr>
<tr><td>EC_i</td><td>[0,n] Objet</td><td>Liste des entrées de contrôle</td></tr>
<tr><td>SC_i</td><td>[0,n] Objet</td><td colspan="2">Liste des sorties de contrôle spécifiques</td></tr>
<tr><td>RS_i</td><td>[1,n] Objet</td><td colspan="2">Liste des ressources</td></tr>
<tr><td>EnsApt</td><td>Objet</td><td colspan="2">Ensemble des aptitudes/compétences requises pour la réalisation de l'occurrence de l'activité en cours, dans le but d'achever l'objectif de l'activité</td></tr>
<tr><td>D</td><td>Entier (secondes)</td><td colspan="2">La durée</td></tr>
<tr><td>Coût</td><td>Nombre</td><td colspan="2">Le coût de l'activité</td></tr>
</table>

Les méthodes de la classe Activité sont les différentes opérations qu'une occurrence activité peut exécuter.

Ces méthodes peuvent décrire :

- Le fonctionnement de l'activité : *Exécution(BR_i)*,
- La définition des aptitudes/compétences requises pour l'exécution de l'occurrence de l'activité *DefinirAptitudes()*.

En fait, les entrées/sorties et les ressources des activités appartenant à un processus métier composent les entrées/sorties et les ressources de celui ci.

Cela signifie que les entrées des activités lancées au début du processus métier représentent les entrées de celui-ci. Et les sorties des dernières activités exécutées sont les sorties du processus métier. De même, l'ensemble des ressources des activités utilisées compose les ressources nécessaires pour le processus métier.

De même, l'agrégation des objectifs d'activités compose l'objectif global du processus métier.

a.iii) Règle de comportement

Les règles de comportement (§ Annexe A) sont utilisées pour pouvoir définir le flux de contrôle associé aux processus. En fait, elles définissent les décisions logiques requises pour déterminer les séquences des processus et des activités, dans le but d'exécuter les tâches générales d'un processus maître ou d'un processus métier.

Ces règles peuvent être utilisées pour le déclenchement de processus maître/métier, le déclenchement forcé, le déclenchement conditionnel, le déclenchement parallèle, le déclenchement en rendez-vous, la définition de boucle et la définition de clôture de processus [Berio, 99].

Elles peuvent être représentées par des classes d'objet ou par un module.

a.iv) Domaine

Comme cité précédemment, le concept de 'Domaine' (DM) permet de regrouper un ensemble de processus en un modèle indépendant de manière à gérer la complexité du système à modéliser. Ce qui est intéressant dans ce concept, c'est qu'il facilite le partitionnement des processus, pour réaliser l'un des objectifs du système. Ainsi, un domaine est une agrégation de processus. Un processus doit donc appartenir à un seul domaine.

Cependant, un domaine regroupe des processus maîtres (*Domain Processes DP*), ainsi nous pouvons exprimer un domaine DM_i comme suit :

$$DM_i=\{DP_{i,j}\}$$

Avec $DP_{i,j}$ les processus maîtres (j) appartenant au domaine (i).

Et $1 \leq i \leq n$; $1 \leq j \leq m$

n : nombre de domaines existants dans le système

m : nombre de processus existants dans le système

(Voir Figure IV.3-1)

Le concept de Domaine est caractérisé par les propriétés suivantes:

- Identificateur,
- Nom,
- Description du domaine,
- Liste des processus maîtres.

La classe d'objets Domaine peut donc avoir les attributs suivants :

Attributs de la classe Domaine		**Caractéristiques d'un Domaine**
Attribut	***Type***	
Nom	Chaîne de caractères	Nom
DM_i	Chaîne de caractères	Identifiant
Descript	Chaîne de caractères	Description du domaine
$DP_{i,j}$	[1,n] Objet	Liste des processus maîtres

Une classe d'objet Domaine peut aussi assurer les fonctions/méthodes suivantes :

- Définir les processus maîtres de l'instance Domaine : *AjouterProcessus(DP_i)*,
- Supprimer les processus maîtres de l'instance Domaine : *SupprimerProcessus(DP_i)*.

Les domaines communiquent entre eux par l'envoi et la réception d'événements et/ou d'objets. Cette communication est définie dans CIMOSA par une relation entre domaines.

a.v) Relations entre domaines

La classe Relation-Domaines définit les interactions entre deux domaines en interaction.

Elle indique aussi les événements et les vues d'objet qui peuvent être échangés entre deux domaines, en spécifiant les intrants et les extrants par rapport aux domaines.

Attribut de la classe Relation-Domaines		**Caractéristiques de la classe Relation-Domaines**
Attribut	***Type***	
Nom	Chaîne de caractères	Nom
R-DM	Chaîne de caractères	identifiant
AtC_i	[1,n] Objet	Liste des autoritaires de conception
Description	Chaîne de caractères	Courte description de l'instance de la Relation-Domaines
DM-1	[1,1] Objet	Nom de la première instance de domaine incluse dans l'instance de la relation
DM-2	[1,1] Objet	Nom de la deuxième instance de domaine incluse dans l'instance de la relation. DM-2 ≠ DM-1
VO_i	[0,n] Objet	Liste d'instances des vues d'objet incluses contenant :
Id	Chaîne de caractères	Identificateur de l'instance de la vue d'objet
Nom	Chaîne de caractères	Nom de l'instance de la vue d'objet
VO_i_DM-S	[1,1] Objet	L'instance du domaine source de laquelle les occurrences de l'instance de la vue d'objet sont envoyées.
VO_i_DM-C	[1,1] Objet	L'instance du domaine cible à laquelle les occurrences de l'instance de la vue d'objet sont envoyées.
Fréquence	Entier	La fréquence d'échange des occurrences des vues d'objet
EV_i	[0,n] Objet	Liste d'instances des événements inclus contenant :
Id	Chaîne de caractères	Identificateur de l'instance d'événement
Nom	Chaîne de caractères	Nom de l'instance d'événement
EV_i_DM-S	[1,1] Objet	L'instance du domaine source de laquelle les occurrences de l'instance d'événement sont envoyées.
EV_i_DM-C	[1,1] Objet	L'instance du domaine cible à laquelle les occurrences de l'instance d'événement sont envoyées.
Fréquence	Entier	La fréquence d'échange des occurrences des événements

Cette classe doit permettre d'indentifier :

- les deux domaines DM-1 et DM2 : *IdentifierDM(DM$_i$-1, DM$_j$-2).*
- les vues d'objet incluses dans la relation : *IdentifierVO (Nom, Id, VO$_i$_DM-S, VO$_i$_DM-C, Fréquence).*
- les événements inclus dans la relation : *IdentifierVO (Nom, Id, EV$_i$_DM-S, EV$_i$_DM-C, Fréquence).*

a.vi) Objectifs

Comme défini par les patrons métier 'Structure de Base d'un Processus' et 'Objectif Métier' (§ Annexe B), chaque processus doit réaliser un objectif. Puisque chaque objectif peut être décomposé en sous-objectifs, comme décrit dans le patron 'Décomposition d'un Objectif Métier' (§ Annexe B) – Alors, une agrégation des objectifs est possible.

Lorsque tous les sous-objectifs sont atteints, l'objectif composite est atteint. On note qu'un objectif peut être composé de sous-objectifs spécifiques aux activités/sous-processus correspondants. Cela est appelé, dans UML [Booch, 00], la composition, et se représente graphiquement comme dans la Figure IV.3-4:

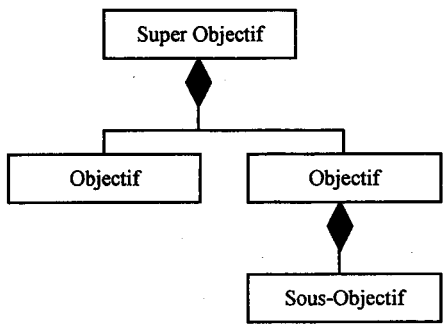

Figure IV.3-4- Décomposition d'objectifs

Cette représentation peut être raffinée en introduisant les termes 'Parent' et 'Enfant', pour distinguer les composites de leurs composants, comme défini par le patron de conception 'Composite' [Gamma, 95]. Rappelons qu'un objectif (1) peut être composé de plusieurs (0..*) sous-objectifs.

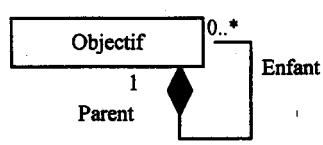

Figure IV.3-5- Application du patron de conception 'Composite'

Un objectif est caractérisé par :

- Un identificateur,
- Un nom,
- Description de l'objectif,
- Parent de l'objectif.

Ceci permet de définir les attributs de la classe d'objets Objectif :

Attributs de la classe Objectif		**Caractéristiques d'un Objectif**
Attribut	***Type***	
Nom	Chaîne de caractères	Nom
OB_i	Chaîne de caractères	Identifiant
Descript	Chaîne de caractères	Description de l'objectif
Parent	Chaîne de caractères	Parent de l'objectif

La classe Objectif peut avoir les méthodes suivantes :

- *DecomposObjectif(Objectif$_i$)* : permet de décomposer un objectif en sous-objectifs,
- *SupprimObjectif(Objectif$_i$)* : supprimer des sous-objectifs.

a.vii) Structure de l'aspect fonctionnel

Dans nos travaux, le méta-modèle que nous proposons permet de modéliser les processus des systèmes de l'entreprise. Du fait que les domaines peuvent diviser un système en parties pour faciliter la gestion de sa complexité, nous pouvons définir un système en tant qu'ensemble de domaines. Or, le système peut être défini par des domaines existants ou qui ne seront pas modélisés à l'aide du méta-modèle. Nous les appelons des Domaines Non-CIMOSA. Alors que les domaines qui sont modélisés par le méta-modèle sont des Domaines CIMOSA.

C'est ainsi que nous proposons le méta-modèle de la Figure IV.3-6.

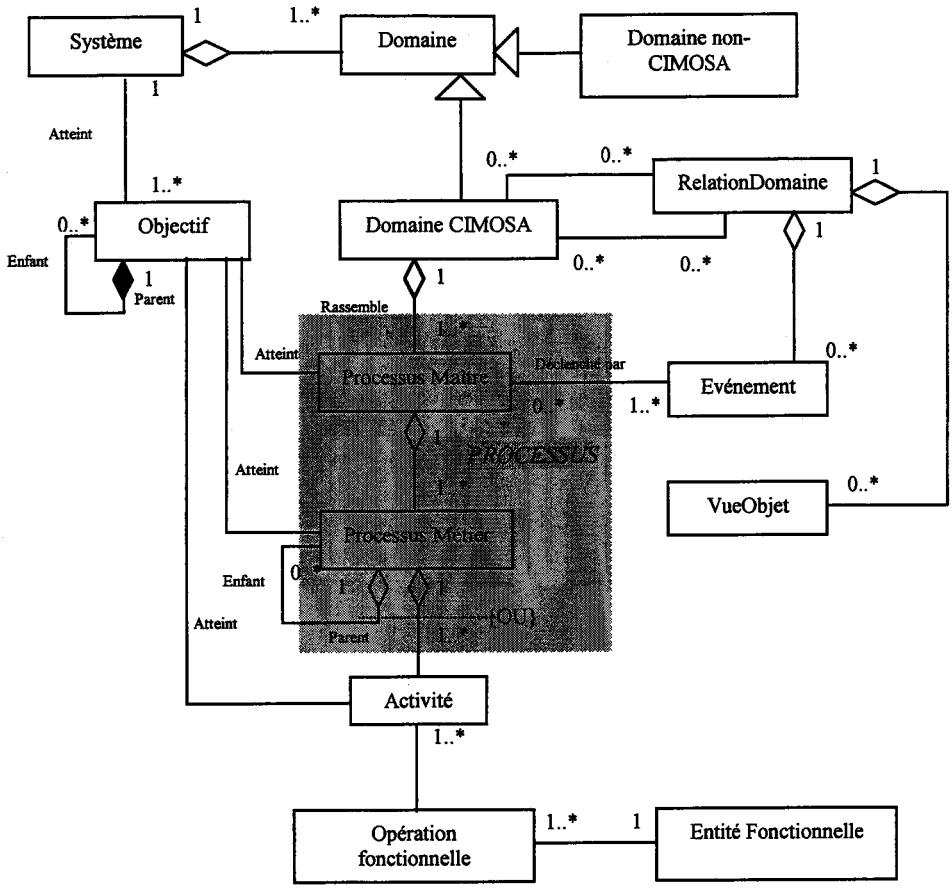

Figure IV.3-6 – Méta-modèle (modèle fonctionnel)

a.viii) Conclusion

Ce méta-modèle fonctionnel permet aux acteurs métier de les guider lors d'une modélisation.

Une méthodologie à suivre pour utiliser ce méta-modèle fonctionnel est nécessaire. Par contre, elle dépend de la façon d'implémentation du modèle. Elle est décrite dans la section (IV.4) qui traite l'implémentation du méta-modèle par l'approche de composants.

Dans ce qui suit, nous définissons les concepts restants qui définissent les processus métier, selon les autres aspects de modélisation.

b) Aspects informationnels

Un processus ne peut être exécuté sans l'utilisation de ressources [CIMOSA, 94], [Eriksson, 00] – machines, opérateurs, équipements – pour pouvoir transformer des états d'entrée, appelés vues d'objet (VO) ou « Object Views » dans CIMOSA [CIMOSA, 94], en états de sortie. Les vues d'objet représentent un ou des états d'une ou plusieurs entités, appelées objets d'entreprise (OE) ou « Enterprise Objects ». Un objet d'entreprise peut être de l'information ou un objet physique (un produit, une matière, une personne) [CIMOSA, 94].

Prenons comme exemple un centre de formation. Ce dernier reçoit des personnes (stagiaires) non formées (vue d'objet). La formation (processus) sera assurée par des formateurs (ressources humaines). Quand la formation est à terme, ces stagiaires sont formés. On remarque que les stagiaires non formés ont changé d'état à la fin du processus de formation. Ils sont le résultat du processus parce que son objectif (former des stagiaires) est atteint. On peut définir ces stagiaires comme des objets humains d'entreprise.

Dans le cas où le stagiaire ne réussit pas le test final de la formation, il peut être formé à nouveau ou recalé. Donc, le résultat du test est un indicateur de performance qui montre si l'objectif du processus est atteint (réussir la formation de stagiaire).

Par conséquent, pour définir les états d'entrée/sortie (vues d'objet) d'un processus, il faut identifier le ou les objets qui peuvent avoir ces états.

Ces vues d'objet peuvent être regroupées dans un schéma conceptuel qui, par analogie avec l'approche système d'information peut être identifié à un modèle conceptuel de type entité-relation.

b.i) Objets d'entreprise

D'après CIMOSA, un objet d'entreprise est une entité (produit, planning, nomenclature) du monde réel de l'entreprise, c'est à dire une chose ayant une identité, et un cycle de vie qui lui sont propres.

Nous représentons ce concept par deux classes d'objet :

- Objet d'entreprise,
- Elément d'information, qui définit les propriétés d'une occurrence d'Objet d'entreprise.

Le concept Objet d'entreprise est caractérisé par :

- Un identificateur,
- Un nom,
- La famille de l'objet,

Ainsi, une classe d'objet 'Objet d'entreprise' aura les attributs suivants :

Attributs de la classe Objetentreprise		**Caractéristiques d'un Objet d'entreprise**
Attribut	***Type***	
Nom	Chaîne de caractères	Nom
EO_i	Chaîne de caractères	Identifiant
Famille	Chaîne de caractères	
EI_i	[1,n] Objet	Ensemble des éléments d'information définissant les propriétés de l'occurrence courante de l'objet d'entreprise

Cette classe permet de réaliser les fonctions suivantes :

- Ajouter un élément d'information : *AjouterElementInformation(EI_i)*,
- Ou supprimer un élément : *SupprimerElementInformation(EI_i)*,
- Ajouter des vues d'objets représentant l'objet d'entreprise étudié : *AjouterVueObjet()*,
- Ou dissocier les vues d'objets représentant l'objet d'entreprise étudié : *AjouterVueObjet()*,

La classe d'objet ElementInformation est définie dans ce qui suit.

b.ii) Vues d'objet

Une vue d'objet est une manifestation de l'état d'un ou plusieurs objets d'entreprise à un instant donné. Elle peut être physique (objet concret, comme une pièce brute, une pièce usinée, un tour) ou informationnelle (nomenclature d'un produit, ISBN d'un livre). Une vue d'objet agit sur un ou plusieurs objets d'entreprise pour en extraire une représentation (vue) adaptée à l'utilisateur. On peut donc définir autant de vues d'objet que nécessaire pour un même objet ou un ensemble d'objets sans avoir à manipuler la représentation complète de chaque objet, selon l'environnement.

Nous représentons ce concept par deux classes d'objet :

- Vue d'objet,
- Elément d'information qui représente les propriétés de l'occurrence Vue d'objet.

Une vue d'objet est caractérisée par :

- Un identificateur,
- Un nom,

- La famille de la vue d'objet,
- La nature de la vue d'objet (informationnelle ou physique),
- La cardinalité,
- La persistance de la vue d'objet (permanent, temporaire),

Ceci permet de définir les attributs de la classe d'objet VueObjet.

Cette classe d'objet permet d'exécuter les opérations (méthodes) suivantes :

- Ajouter un élément d'information qui définit une propriété de la vue d'objet : *AjouterElementInformation(EI_i),*
- Supprimer un élément d'information : *AjouterElementInformation(EI_i).*

Nous pouvons exprimer des occurrences de vues d'objet VO_i en fonction d'objets d'entreprise OE_j, et de leurs états correspondants :

$$(VO_i) = [etat_{i,j}] \bullet (OE_j)$$

Avec :

(VO_i) : vecteur des occurences des vues d'objets

$[etat_{i,j}]$: matrice définissant les états représentées par chacune des occurences des vues d'objets, de chaque occurence de l'objet d'entreprise

$[OE_j]$: vecteur des occurences des objets d'entreprise

Attributs de la classe VueObjet		**Caractéristiques d'une Vue d'objet**
Attribut	***Type***	
Nom	Chaîne de caractères	Nom
VO_i	Chaîne de caractères	Identifiant
Famille	Chaîne de caractères	
Nature	Chaîne de caractères	Physique ou informationnel
Cardinalité	Entier	Nombre des occurrences de la vue d'objet
Persistance	Chaîne de caractères	(permanent, temporaire)
EI_i	[1,n] Objet	L'ensemble des éléments d'informations définissant les propriétés de l'occurrence courante de la vue d'objet

b.iii) Éléments d'information

Le concept élément d'information est défini par :

- Son nom,
- Son identificateur,
- Son type (nombre, caractère, ...),
- Les contraintes (règles d'intégrité) applicables à cet élément.

D'où les attributs de la classe ElementInformation :

Attributs de la classe ElementInformation		**Caractéristiques d'un élément d'information**
Attribut	***Type***	
Nom	Chaîne de caractères	Nom
EI	Chaîne de caractères	Identifiant
TypeElément	Chaîne de caractères	(Entier, réel, caractère, tableau, ...)
Liste de contraintes	[0,n] chaîne de caractères	

b.iv) Schéma conceptuel

Les vues d'objet, accompagnées par une classe d'objet Relation, sont utilisées pour développer un schéma conceptuel. Ce dernier permet de spécifier la structure du système d'information de l'entreprise qui se matérialise par la réalisation d'une ou plusieurs bases de données.

Nous définissons le schéma conceptuel par :

- Son nom,
- Son identificateur,
- Une liste d'entités qui définit les vues d'objet utilisées,
- La liste des relations entre les entités,

Donc, en représentant ce concept par une classe d'objet SchemaConceptuel, les attributs sont :

Attributs de la classe SchemaConceptuel		**Caractéristiques d'un schéma conceptuel**
Attribut	***Type***	
Nom	Chaîne de caractères	Nom
SC	Chaîne de caractères	Identifiant
ListeEntités	[2,n] Objet	Des vues d'objet

ListeRelations	[1,n] Objet	Les relations entre les entités

Pour pouvoir gérer les entités et les relations d'un schéma conceptuel, la classe d'objet correspondante permet les opérations (méthodes) suivantes :

- Ajouter une entité : *AjouterVueObjet()*,
- Supprimer une entité : *SupprimerVueObjet()*,
- Ajouter une relation entre deux ou plusieurs entités : *AjouterRelation()*,
- Supprimer une relation : *SupprimerRelation()*.

b.v) Relations

Les objets relations entre les entités sont représentés par la classe d'objet Relation, ayant pour attributs :

Attributs de la classe Relation		**Caractéristiques d'une Relation**
Attribut	***Type***	
Nom	Chaîne de caractères	Nom
R	Chaîne de caractères	Identifiant
EntitéSource	[1,1] Objet	Des vues d'objet
EntitéCible	[1,1] Objet	Des vues d'objet

b.vi) Structure de l'aspect informationnel

Après la définition des classes représentant les différents concepts des aspects informationnels du méta-modèle, nous représentons ces classes par la Figure IV.3-7:

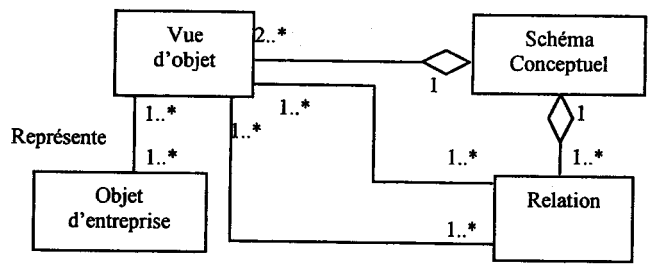

Figure IV.3-7 – Aspects informationnels du méta-modèle

b.vii) Conclusion

Les aspects informationnels du méta-modèle permettent d'aider les modélisateurs à modéliser les informations et les données des objets techniques existant dans l'entreprise, tout en développant des schémas conceptuels dans le but de réaliser des bases de données. Ces objets peuvent être aussi des moyens et ressources de l'entreprise. Leurs états (vues d'objet)

permettent de spécifier les entrées/sorties des processus et des activités modélisées par le modèle fonctionnel (aspects fonctionnels du méta-modèle).

c) Aspects relatifs aux ressources

En spécifiant le modèle fonctionnel, on note que se sont les ressources qui exécutent les activités pour traiter les objets d'entreprise via leurs vues d'objet, dans le but d'atteindre un objectif.

Ces ressources sont représentées par des classes abstraites. Par exemple, un outil informatique (ressource) permet d'exécuter l'activité 'documenter des informations'. Or, pour spécifier les ressources, CIMOSA a défini le concept d'entité fonctionnelle (EF) (ressource active) qui exécute des opérations fonctionnelles pour accomplir une activité. Par exemple, un logiciel informatique, exemple MsWord (entité fonctionnelle), permet de taper, sauvegarder et restituer des textes (opérations fonctionnelles), par une secrétaire (entité fonctionnelle). Chacune de ces opérations fonctionnelles appartiennent à l'activité 'documenter des informations'. L'entité fonctionnelle 'MsWord' est la spécification de la ressource 'Outil informatique'.

En utilisant le diagramme de classes, l'aspect ressource du méta-modèle se présente comme suit (Figure IV.3-8):

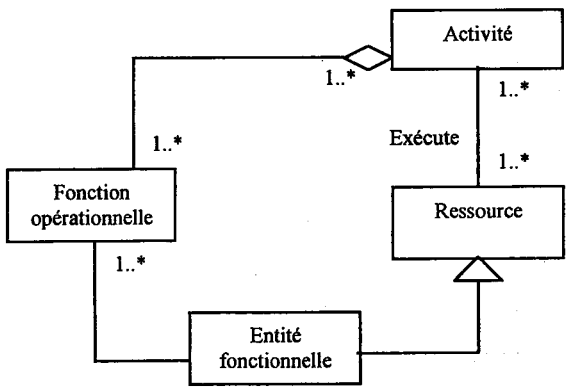

Figure IV.3-8- Aspects ressource

Dans CIMOSA, une ressource RC_i peut être spécifiée en termes d'entités fonctionnelles EF_j et de composants ressources (ressources passives telles que des réglets, des outils d'usinage).

$$RC_i=\{EF_j\}\cup\{RessourceComposant_k\}$$

Avec $1 \le i \le n$; $1 \le j \le m$; $1 \le k \le p$ *et* $n = m + p$

n : nombre des ressources existantes.

m : nombre de ces ressources identifiées comme des entités fonctionnelles EF_j

c.i) Ensembles d'aptitudes

Avant d'associer des ressources aux activités, il est obligatoire d'étudier les aptitudes ou compétences requises pour réaliser de telles activités. Cet ensemble d'aptitudes ou de compétences permet de trouver facilement les ressources correspondantes à toutes les aptitudes/compétences requises.

Ceci nous permet d'ajouter un concept dans les aspects fonctionnels qui détermine les aptitudes/compétences requises par chaque activité.

Par contre, il est important d'identifier l'ensemble des aptitudes/compétences acquises par chaque ressource. Cela permet une association des ressources aux activités, par correspondance entre l'ensemble d'aptitudes/compétences d'une activité et ceux des différentes ressources.

Ainsi, nous utilisons le concept 'Ensemble d'Aptitudes/compétences' (EAP), qui est caractérisé par :

- Identifiant,
- Nom,
- Aptitudes/compétences fonctionnelles : liste des aptitudes/compétences liées à la fonction à réaliser,
- Aptitudes/compétences relatives aux objets : liste des aptitudes/compétences relatives aux objets à manipuler,
- Aptitudes/compétences de performance : liste des aptitudes/compétences relatives aux performances de la fonction à réaliser,
- Aptitudes/compétences opérationnelles : liste des aptitudes/compétences de nature opérationnelle,
- Compétences : liste des compétences relatives au savoir, savoir-faire et savoir-être acquis ou exigés d'un individu (pour les ressources humaines).

Nous pouvons exprimer un ensemble d'aptitudes EAP_i par quatre axes :

- Aptitudes/compétences fonctionnelles :

a_x.**f** (a_x : Aptitude, **f** : vecteur relatif aux aptitudes/compétences fonctionnelles)

- Aptitudes/compétences relatives aux objets :

  a_y.**b** (a_y : Aptitude, **b** : vecteur relatif aux aptitudes/compétences des objets)

- Aptitudes/compétences de performance :

  a_z.**m** (a_z : Aptitude, **m** : vecteur relatif aux aptitudes/compétences de performance)

- Aptitudes/compétences opérationnelles :

  a_w.**r** (a_w : Aptitude, **r** : vecteur relatif aux aptitudes/compétences opérationnelles)

Donc :

$$EAP_i=\{a_x\}.\mathbf{f} \cup \{a_y\}.\mathbf{b} \cup \{a_z\}.\mathbf{m} \cup \{a_w\}.\mathbf{r}$$

Avec $1 \leq x, y, z$ *ou* $w \leq n, m, k, p$

n, m, k, p : Nombres des aptitudes/compétences correspondant à un axe.

Cette expression peut s'écrire sous la forme d'une matrice :

$$EAP_i=[a_{gi}] \; ; 1 \leq g \leq 4 \text{ et } 1 \leq i \leq m$$

Avec

g : nombre des axes d'aptitudes/compétences.

i : nombre des ensembles d'aptitudes/compétences EAP_i

Une activité EA_i peut avoir un Ensemble d'Aptitudes/compétences EAP_i. Cela s'ajoute aux opérations fonctionnelles OF_j :

$$EA_i=\{OF_j\} \cup [a_{gi}]$$

Avec $1 \leq i \leq n$

n : nombre des activités existantes dans le système.

Une ressource RC_i peut être exprimée, aussi, en fonction de l'Ensemble d'Aptitudes/compétences EAP_l , comme suit :

$$RC_i=\{EF_j\} \cup \{RessourceComposant_k\} \cup [a_{gi}]$$

Avec $1 \leq i \leq n$

n : nombre des ressources existantes.

Ainsi, en définissant les deux concepts « ensemble d'aptitudes/compétences » et « aptitude » par des classes d'objet, leurs attributs peuvent être les suivants :

Attributs de la classe EnsembleAptitudes/compétences		**Caractéristiques d'un ensemble d'aptitudes/compétences**
Attribut	***Type***	
Nom	Chaîne de caractères	Nom
EAP_i	Chaîne de caractères	Identifiant
ListeAptitudes/compétences	[1,n] Objet	

Et

Attributs de la classe Aptitude		**Caractéristiques d'une aptitude**
Attribut	***Type***	
Nom	Chaîne de caractères	Nom
Apt	Chaîne de caractères	Identifiant
Descript	Chaîne de caractères	Description de l'aptitude
Famille	Chaîne de caractères	fonctionnelle, objet, performance, opérationnelle

La classe d'objet EnsembleAptitudes/compétences permet :

- D'ajouter des aptitudes/compétences : *AjouterAptitude(Apt)*,
- De supprimer une aptitude : *SupprimerAptitude(Apt)*.

Remarque :

La classe d'ensemble d'aptitudes/compétences fournies par une ressource s'appelle : EnsembleAptitudes/compétences_Ressource, et celle requise par une activité est : EnsembleAptitudes/compétences.

Un traitement approfondi des concepts d'aptitudes/compétences et de l'ensemble d'aptitudes a été donné par Harzallah [Harzallah, 00] essentiellement dans son modèle CRAI (Competency Resource Adspect Indivual) [Harzallah, 02]. Mais, du fait que cette thèse est

focalisé essentiellement sur la modélisation des processus, nous nous limitons aux définitions données dans CIMOSA [CIMOSA, 94] en utilisant l'approche objet.

c.ii) Ressources

Nous représentons le concept de ressource par une classe d'objet Ressource ayant pour attributs :

Attributs de la classe Ressource		**Caractéristiques d'une ressource**
Attribut	***Type***	
Nom	Chaîne de caractères	Nom
RS	Chaîne de caractères	Identifiant
Famille	Chaîne de caractères	
Classe	Chaîne de caractères	Entité fonctionnelle ou composant ressource
AdresseLogique	Chaîne de caractères	Adresse logique de l'occurrence courante de la ressource (cas de ressources informatiques)
AdressePhysique	Chaîne de caractères	Adresse physique de l'occurrence courante de la ressource
AdresseElectronique	Chaîne de caractères	Adresse électronique de l'occurrence courante de la ressource (s'il y en a une)
Quantité	Entier	Nombre d'occurences ressources existantes appartenant à la Famille
EnsembleAptitudes	Objet	L'ensemble d'aptitudes fourni par l'occurrence courante

Cette classe d'objet aide les modélisateurs à définir l'ensemble des aptitudes/compétences d'une occurrence ressource, de la supprimer et de spécifier la ressource en entité fonctionnelle ou en composant ressource. Dans le cas d'une entité fonctionnelle, la ressource assure la gestion des opérations fonctionnelles :

- Spécifier l'occurrence ressource : *SpécifierRessource()*,
- Ajouter des opérations fonctionnelles, dans le cas d'entité fonctionnelle : *AjouterOperationFonctionnelle()*,
- Supprimer une opération fonctionnelle : *SupprimerOperationFonctionnelle()*.

c.iii) Structure de l'aspect ressource

Finalement, la structure de l'aspect ressource du méta-modèle est représentée par la Figure IV.3-9 :

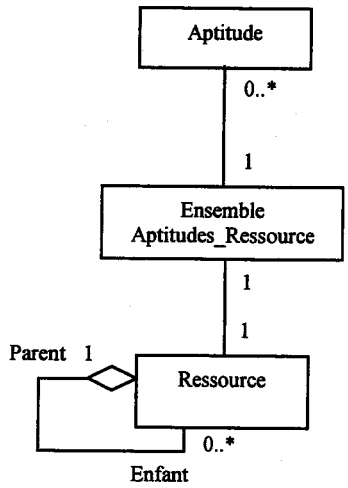

Figure IV.3-9 – Aspects de ressource du méta-modèle

c.iv) Conclusion

Les aspects ressource du méta-modèle permettent de gérer les moyens et les ressources de l'entreprise, tout en définissant leurs aptitudes (compétences/capacités). Ceci facilite l'identification et la spécification des ressources d'une activité par le biais de leurs ensembles d'aptitudes requises.

d) Aspects organisationnels

Un processus est manipulé par des personnes de l'entreprise. Ces personnes ont une certaine autorité/responsabilité envers celui-ci. Cependant, le processus peut être placé sous l'autorité d'une personne, d'un système informatique, d'une machine, etc. Nous pouvons généraliser ces entités en tant que 'Acteurs' (Figure IV.3-10).

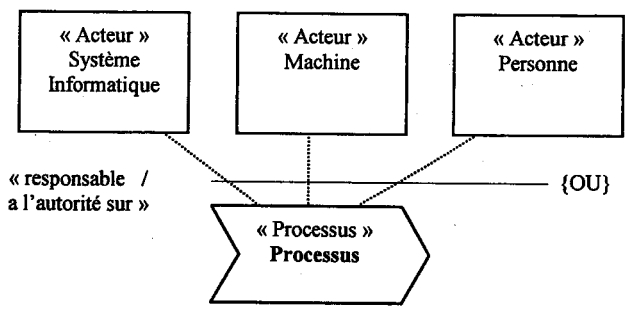

Figure IV.3-10- Notion d'Acteur

d.i) Unités d'organisations

Par définition, « un acteur est une personne ou une entité matérielle autonome, telle qu'une personne ou une machine. Le patron Acteur-Rôle (§ Annexe B), définit un deuxième concept associé au concept 'Acteur' (AT), c'est le concept de 'Rôle' (RO), qui décrit une action prise par un acteur.

Par exemple, une entreprise emploie ERIC comme chef du service de traitements de commandes ; ainsi, ERIC est un acteur (entité fonctionnelle) jouant le rôle de Chef-service (unité d'organisation) de traitement de commandes.

Dans cette section, nous supposons qu'un acteur est considéré en tant qu'entité utilisée dans un département, un service ou une direction pour jouer un rôle et accomplir des tâches (activités/processus). De ce fait, le concept « Acteur » est considéré en tant que ressource. Alors que le concept « Rôle » est la définition d'une unité d'organisation [CIMOSA, 94].

L'allocation de ressources (gestion des ressources) consiste alors à décider d'affecter un acteur A à une unité d'organisation (UO) à la date t. Cette affectation peut changer pour concerner par exemple un acteur B.

Une unité d'organisation (UO) est caractérisé par :

- Identificateur,
- Nom,
- Responsabilités : liste des responsabilités associées à l'unité,
- Autorités : liste des autorités allouées à cette unité,
- Affectée-à : département, service ou direction auquel appartient cette unité.

Nous modélisons ce concept par la classe d'objet UnitéOrganisation, ayant les attributs suivants :

Attributs de la classe UnitéOrganisation		**Caractéristiques d'une unité d'organisation**
Attribut	***Type***	
Nom	Chaîne de caractères	Nom
UO	Chaîne de caractères	Identifiant
Famille	Chaîne de caractères	
CRp_i	[1,n] Objet	Liste des cellules d'organisation responsables de l'occurrence courante d'une unité d'organisation
$CAtO_i$	[1,n] Objet	Liste des cellules d'organisation ayant l'autorité opérationnelle sur l'occurrence courante d'une unité

		d'organisation
$CAtC_i$	[1,n] Objet	Liste des cellules d'organisation ayant l'autorité de conception sur l'occurrence courante d'une unité d'organisation
$EntitéRp_i$	[0,n] Objet	Liste des entités sous la responsabilité de l'occurrence courante de l'unité d'organisation
$EntitéAtO_i$	[0,n] Objet	Liste des entités sous l'autorité de l'occurrence courante de l'unité d'organisation
$EntitéAtC_i$	[1,n] Objet	Liste des entités conçues/maintenues par l'occurrence courante de l'unité d'organisation
CO	[1,1] Objet	Cellule d'organisation (§d.ii)): département, service ou direction auquel appartient cette occurrence d'unité d'organisation
Acteur (ressource)	[1,n] Objet	Acteurs affectés à cette unité d'organisation

Pour pouvoir définir les affectations d'acteurs à une unité d'organisation, nous avons défini les méthodes suivantes de la classe UnitéOrganisation :

- Affecter un acteur : *AjouterActeur(AT)*,
- Supprimer un acteur : *SupprimerActeur(AT)*.

Pour chaque unité d'organisation définie dans une cellule d'organisation (§d.ii), des aptitudes sont requises par des activités pour en associer des acteurs (unité d'organisation). Nous appelons ces aptitudes : EnsembleAptitudes_Organisation.

D'où, nous définissons une méthode permettant de définir l'ensemble d'aptitudes requises pour cette unité d'organisation : *DéfinirEnsembleAptitudes()*.

d.ii) Cellules d'organisation

CIMOSA définit une cellule d'organisation (CO) comme des entités d'organisations composées (c'est-à-dire agrégées). Il peut s'agir de postes (composées d'un ou plusieurs rôles), de services, de départements, de divisions ou de directions, mais aussi des équipes de projet ou toute autre structure transversale [Vernadat, 96].

Une cellule d'organisation est caractérisée par :

- Identification,
- Nom,

- Description,
- Niveau : précisant la position de la cellule dans la structure organisationnelle (poste, service, département, division ou direction, ou une entité),
- Responsable : Acteur humain responsable de la cellule,
- Responsable-pour : liste des entités de l'entreprise (processus/activités, ressources, objets d'entreprise) placées sous la responsabilité de cette cellule,
- Autorité-sur : liste des entités de l'entreprise (processus/activités, ressources, objets d'entreprise) placés sous l'autorité de cette cellule, en définissant les autorités,
- Affectée-à : cellule parent,
- Contient : unités d'organisation et cellules d'organisation composant cette cellule,
- Reliée-à : cellules en relation avec celle-ci.

En représentant le concept de cellule d'organisation par la classe d'objet CelluleOrganisation, nous définissons les attributs de celle-ci comme suit :

Attributs de la classe CelluleOrganisation		**Caractéristiques d'une cellule d'organisation**
Attribut	***Type***	
Nom	Chaîne de caractères	Nom
CO	Chaîne de caractères	Identifiant
Famille	Chaîne de caractères	
CRp_i	[1,n] Objet	Liste des cellules d'organisation responsables de l'occurrence courante d'une cellule d'organisation
$CAtO_i$	[1,n] Objet	Liste des cellules d'organisation ayant l'autorité opérationnelle sur l'occurrence courante d'une cellule d'organisation
$CAtC_i$	[1,n] Objet	Liste des cellules d'organisation ayant l'autorité de conception sur l'occurrence courante d'une cellule d'organisation
$EntitéRp_i$	[0,n] Objet	Liste des entités sous la responsabilité de l'occurrence courante de la cellule d'organisation
$EntitéAtO_i$	[0,n] Objet	Liste des entités sous l'autorité de l'occurrence courante de la cellule d'organisation
$EntitéAtC_i$	[1,n] Objet	Liste des entités conçues/maintenues par l'occurrence courante de la cellule d'organisation
COParent	[1,1] Objet	Cellule d'organisation à laquelle appartient cette occurrence de cellule d'organisation
Relié	[0,n] Objet	Cellules en relation avec cette occurrence
Niveau	Chaîne de caractères	la position de la cellule dans la structure

		organisationnelle
Responsable	Objet	Acteur humain responsable de la cellule

Cette classe assure les opérations suivantes :

- Ajouter une occurrence CelluleOrganisation enfant à l'occurrence CelluleOrganisation en cours: *AjouterCO()*,
- Supprimer une occurrence CelluleOrganisation enfant de l'occurrence CelluleOrganisation en cours: *SupprimerCO()*,
- Ajouter une occurrence UnitéOrganisation à l'occurrence CelluleOrganisation en cours: *AjouterUO()*,
- Supprimer une occurrence UnitéOrganisation de l'occurrence CelluleOrganisation en cours : *SupprimerUO()*,

Nous pouvons exprimer une cellule d'organisation CO_i en fonction des unités d'organisation $UO_{i,j}$ qu'elle gère :

$$CO_i=\{UO_{i,j}\}_{1\le i\le n\,;\,1\le j\le m}$$

Avec

n : nombre de cellules d'organisation existantes dans l'entreprise

m : nombre de unités d'organisation existantes dans la cellule CO_i

Nous exprimons aussi une unité d'organisation $\{UO_{i,j}\}$ en fonction de ses acteurs $AT_{i,j,k}$.

$$\{UO_{i,j}\}= AT_{i,j,k\ 1\le i\le n\,;\,1\le j\le m;\,1\le k\le p}$$

Avec

p : nombre d'unités d'organisation $UO_{i,j}$ ayant pour acteurs $AT_{i,j,k}$ qui appartient à la cellule CO_i.

Les cellules d'organisation représentent les entités d'organisation d'une structure d'organisation.

Des relations de hiérarchie ou de distribution peuvent exister entre les différentes entités.

d.iii) Structure d'organisation

Ce concept est représenté par une classe d'objet. Il est utilisé pour déterminer les entités d'organisation nécessaires pour le système à modéliser.

La classe d'objet StructureOrganisation est caractérisée par :

- Un nom (chaîne de caractère)
- La liste des cellules d'organisation ([1,n] Objet)
- La liste des relations entre les cellules ([1,n] Objet)

Les méthodes nécessaires pour cette classe sont :

- *AjouterCO()* : ajouter une cellule d'organisation dans la structure,
- *SupprimerCO()* : supprimer une cellule d'organisation de la structure,
- *AjouterRelation()* : ajouter une relation entre deux cellules d'organisation,
- *SupprimerRelation()* : supprimer une relation existante entre deux cellules.

d.iv) Relations d'organisation

Cette relation peut être décrite par une classe d'objet RelationOrganisation, qui nous définissons dans cette thèse que par :

- Un nom (chaîne de caractères),
- Un identificateur (chaîne de caractère),
- Une première cellule d'organisation,
- Une deuxième cellule d'organisation.

Cette classe peut avoir d'autres attributs, mais nous nous contentons de cette structure très basique dans cette thèse.

d.v) Structure des aspects organisationnels

Les aspects organisationnels sont représentés par la Figure IV.3-11 :

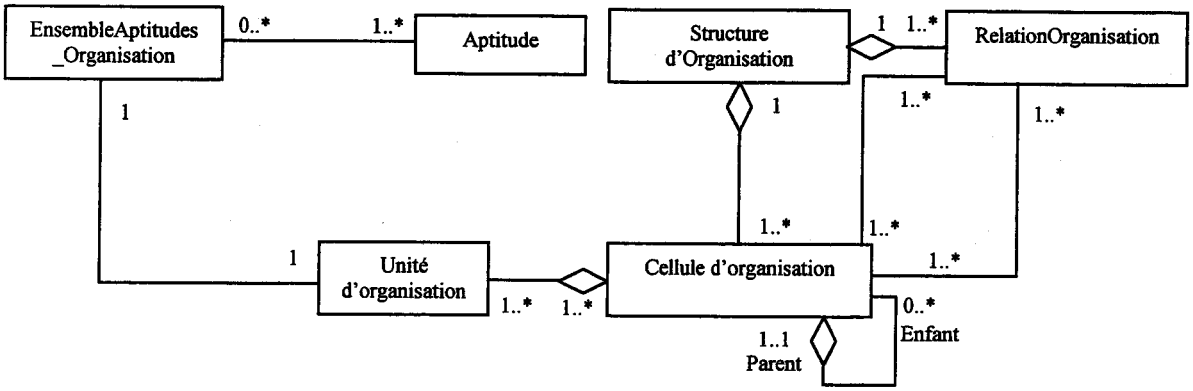

Figure IV.3-11- Aspect Organisationnel

La Figure IV.3-11 donne le diagramme de classes qui définit une décomposition de cellules d'organisation d'une structure d'organisation (définies dans le Patron 'Organisation' comme des unités d'organisation § Annexe B). Dans chaque cellule, des rôles ont été définis par des unités responsables de la cellule et appartenant à la même cellule ou à un niveau plus haut. Ces rôles sont associés à des unités d'organisation (§ Annexe B) appartenant à la cellule.

d.vi) Conclusion

Ces aspects permettent de définir une structure organisationnelle nécessaire pour la coordination d'un système, tout en définissant les responsabilités et les autorités de chacune des entités d'organisation dans la structure d'organisation nécessaire au système, ainsi que les acteurs affectés à chaque unité d'organisation. Les unités et cellules d'organisation sont à rapprocher des centres de décisions dans la méthode GRAI.

IV.3.3. STRUCTURE DU META-MODELE

En récapitulant les quatre aspects traités ci-dessus, nous obtenons la structure complète du méta-modèle (Figure IV.3-12).

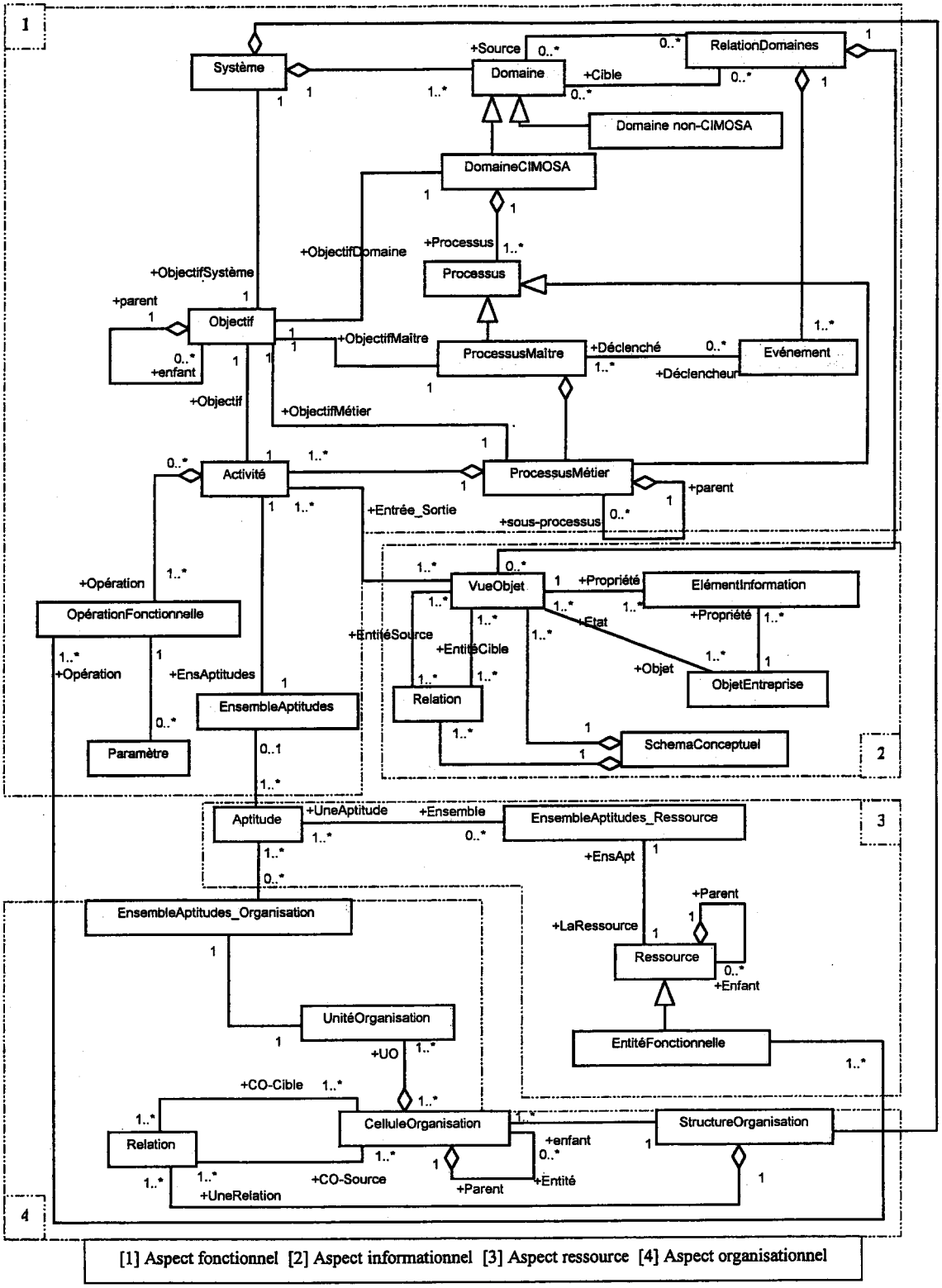

Figure IV.3-12- Méta-Modèle basé sur l'approche CIMOSA

IV.3.4. CONCLUSION

Le méta-modèle proposé se base sur les concepts de CIMOSA, ceux-ci ont été enrichis en utilisant des patrons métier [Eriksson, 00]. Par conséquent, le méta-modèle intègre le savoir et le savoir-faire de la modélisation métier et permet leur réutilisation dans le but de développer des modèles de systèmes. Ces derniers sont orientés processus métier.

Chacun des aspects du méta-modèle aide à réaliser des modèles décrivant un point de vue particulier du système. En effet, les *aspects fonctionnels du méta-modèle* permettent de modéliser les processus métier et leurs étapes.

Les *aspects informationnels du méta-modèle* aident à développer des bases de données et définir les objets d'entreprise que le système utilisera. Les états (vues d'objet) de ces derniers peuvent être utilisés pour identifier et spécifier les entrées/sorties des activités et des processus modélisés à l'aide des aspects fonctionnels.

Les *aspects de ressource du méta-modèle* permettent de gérer les moyens de l'entreprise (ressources) nécessaires au fonctionnement du système, et de les classer en ressources actives (entités fonctionnelles) et passives (composants ressources). Ils permettent aussi de définir les aptitudes de chaque moyen et les opérations fonctionnelles qu'ils peuvent assurer (dans le cas d'une entité fonctionnelle). Ces aspects permettent aussi d'attribuer des ressources aux activités par le biais de leurs aptitudes requises.

Enfin, les *aspects organisationnels* définissent la structure organisationnelle nécessaire pour la coordination du système. Il permet aussi de définir les responsabilités opérationnelles, les autorités opérationnelles et les autorités de conception de chaque entité (processus, activité, objet, ressource, etc.) du système, et de les attribuer aux entités d'organisation (cellules d'organisation, unités d'organisation).

Néanmoins, ce méta-modèle n'offre qu'une structure statique permettant un développement statique des modèles du système.

De ce fait, la description dynamique du méta-modèle permet le développement de modèles décrivant le comportement d'un système. Par conséquent, nous avons essayé de définir le comportement de différents concepts du méta-modèle lors de son implémentation. Cette dernière est réalisée par une approche permettant la ***réutilisation*** du savoir-faire et la collaboration entre les modélisateurs. Cette approche doit laisser le choix, aux modélisateurs, de modéliser n'importe quels aspects (fonctionnels, informationnels, ressources et/ou organisationnels) du système, séparément ou avec d'autres aspects et dans l'ordre de son

choix. Chaque aspect doit ***collaborer*** avec les autres, tout en étant ***interopérable*** avec les plates-formes d'applications (exemple : plates-formes informatiques).

Cette approche doit aussi permettre d'utiliser les aspects du méta-modèle par n'importe quel modélisateur, en tenant compte de sa localisation géographique. Ceci peut être résolu par une ***portabilité*** facile et rapide des concepts de ces aspects.

En faisant le bilan des caractéristiques (réutilisation, indépendance, collaboration, interopérabilité et portabilité) de cette approche d'implémentation, nous arrivons à la définition d'un composant (§IV.4.2)

IV.4. LES COMPOSANTS MÉTIER DU RÉFÉRENTIEL

IV.4.1. INTRODUCTION

De nos jours, les techniques basées sur les composants deviennent essentielles pour les exigences de qualité d'un logiciel. La première règle est de permettre l'utilisation d'un composant logiciel sans se référer à sa composition interne (son implémentation). La seule chose qu'il faut savoir à propos d'un composant est son interface publiée, qui spécifie les services fournis aux utilisateurs. Il est important de savoir qu'un composant n'est pas limité aux techniques d'implémentation qui utilisent DCOM[10], EJB[11], CORBA[12] [Geib, 97], voire même XML[13] [Young, 00]. Un composant peut avoir n'importe quelle forme d'implémentation garantissant les principes de ***séparation*** d'interfaces et de ***standardisation***. Ces formes d'implémentation peuvent être de nature binaire, de classes, de spécifications, des documents, du code source, des interface (exemple : CORBA, COM/COM+/ActiveX™ [14], EJB, etc.) ou un framework [Kobryn, 00]. Ainsi, une application générique peut être considérée comme un composant logiciel.

Les composants métier que nous proposons représentent les quatre aspects du méta-modèle proposé précédemment. Autrement dit, ils implémentent physiquement le méta-modèle de la section IV.3.

De ce fait, les acteurs métier manipulent ces composants pour avoir accès aux concepts définis dans le méta-modèle et ce, dans le but de développer un ou plusieurs modèles particuliers d'un système ou de l'entreprise.

L'approche par composants que nous proposons permet la réutilisation du savoir-faire de la modélisation métier, sous forme de composants autonomes, réutilisables et partageables par les différents acteurs métier.

Avant de présenter les composants proposés, nous donnons les points essentiels à prendre en compte avant tout développement de composant.

---

[10] Distributed Component Object Model, © 1997 Microsoft Corporation, http://www.microsoft.com/com/tech/dcom.asp
[11] Enterprise JavaBeans, Sun Microsystems, http://www.sun.fr
[12] Common Object Request Broker Architecture, OMG (Object Management Group), http://www.omg.org/
[13] eXtensible Markup Language, W3C, http://www.xml.org/
[14] Component Object Model, Microsoft Corporation ©, http://msdn.microsoft.com/

IV.4.2. DEFINITION

Il y a plusieurs définitions d'un composant, surtout en génie logiciel. Mais, les principes essentiels qui reviennent toujours sont l'***indépendance***, la ***portabilité***, la ***réutilisation*** des services [Brown, 97], l'***interopérabilité*** [Dogac, 98], l'***évolution*** et la ***collaboration*** avec les autres composants [Sprott, 00]. L'indépendance ne signifie pas nécessairement qu'un composant n'a pas de dépendance avec les autres composants [Brereton, 00].

Szyperski [Szyperski, 98] définit un composant comme une unité de composition qui peut être déployée indépendamment et qui peut être le sujet d'une composition par une partie tierce.

Une autre définition donnée par UML suppose qu'un composant est une partie physique et remplaçable d'un système, qui rassemble une implémentation et fournit une réalisation d'un ensemble d'interfaces [Booch, 00].

Sprott [Sprott, 00] décrit un composant en tant qu'une entité séparée et encapsulée. Ceci facilite la gestion, l'évolution et la collaboration avec les autres composants. La ***granularité***, la ***limite*** de ***portée*** et la cohésion interne sont des attributs importants du composant. Un composant élémentaire est facilement extensible et ne possède pas plusieurs relations, mais il a besoin de plus de gestion dans son environnement. Alors qu'un composant offrant beaucoup de services est facile à gérer, mais il aura besoin de plus de maintenance et d'efforts de modification et d'implémentation, parce que la portée de la finalité est plus vaste.

Parfois les termes objet et composant sont souvent considérées comme similaires voire interchangeables. Dans l'approche orientée objet, un composant peut être vu comme une collection d'objets, dans laquelle les objets coopèrent entre eux et sont étroitement liés.

IV.4.3. CARACTERISTIQUES D'UN COMPOSANT

Un composant est spécifié par des fonctions au moyen desquelles il communique avec son environnement, et par d'autres attributs, appelés « attributs extra-fonctionnels », qui définissent le comportement du composant.

Les composants communiquent avec leur environnement via des interfaces. Donc, il doit avoir une interface clairement spécifiée, alors que l'implémentation doit être encapsulée et ne doit pas être consultée à partir de l'environnement.

La caractéristique la plus importante d'un composant est la séparation de l'interface et de l'implémentation.

Un composant interopère avec d'autres composants ou avec des applications dans une architecture prédéfinie [Sprott, 00]. Ceci signifie qu'un composant peut uiliser les fonctions ou services d'un autre composant. L'interopérabilité peut se limiter à la communication entre composants mais en général va bien au-delà.

Dans ce qui suit, nous essayons de définir l'interface et l'implémentation d'un composant.

Une interface d'un composant doit être standard, de ce fait un composant peut être largement réutilisable.

La Figure IV.4-1 illustre un méta-modèle UML pour la spécification sémantique d'un composant [Crnkovic, 02]. Dans ce méta-modèle, un composant est déterminé par son interface (qui fournit et requiert des services) et par des contraintes. Chaque interface consiste à un ensemble d'opérations. En plus, un ensemble de pré-conditions et de post-conditions est associé à chaque opération. Les pré-conditions et les post-conditions dépendent souvent de l'état maintenu par le composant. Au-delà de la spécification du comportement d'un seul composant, les contrats sont utilisés pour spécifier les interactions parmi les groupes de composants. Un contrat liste les contraintes globales qu'un composant peut maintenir invariantes (l'invariant). Pour chaque opération, un contrat liste aussi les contraintes qui doivent être satisfaites par le client (pré-condition), et celles établies en retour (post-conditions). La pré-conditions, les invariants et les post-conditions constituent la spécification du comportement d'un composant.

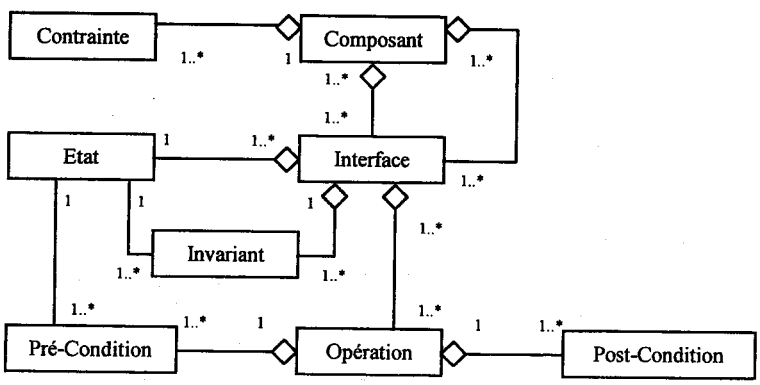

Figure IV.4-1- Méta-modèle UML pour la spécification d'un composant logiciel (d'après [CrnKovic, 02])

IV.4.4. CAS DE NOTRE ÉTUDE

Après développement du méta-modèle de la section IV.3, nous proposons cinq composants qui implémentent celui-ci ; quatre correspondent aux quatre aspects (fonctionnels,

informationnels, ressources et organisationnels) du méta-modèle. Le dernier est un complément pour les aspects fonctionnels et qui permet aux acteurs métier de modéliser le flux de contrôle des processus déjà conçus.

Avant de présenter chaque composant, nous les résumons dans le tableau suivant :

Composant	Aspects méta-modèle correspondant	Rôle
Gestion et Conception de Processus	Fonctionnels	Modélisation de la structure fonctionnel d'un système d'entreprise en définissant ses objectifs, ses domaines et ses processus et en identifiant ses activités.
Comportement		Décrire le flux de contrôle de chaque processus du système.
Gestion d'Information	Informationnels	Identifier les entrées/sorties des processus du système sous forme de vues d'objets.
Gestion de Ressources	Ressources	Identifier les moyens requis pour la réalisation de chaque processus du système.
Structure d'Organisation	Organisationnels	Définir les cellules et les unités d'organisation ayant les responsabilités et les autorités sur les processus, les domaines et le système.

Tableau IV.4-1 – Résumé des composants métier proposés

Dans ce qui suit, nous présentons chaque composant par :

- Son diagramme de cas d'utilisation qui représente les fonctions de génériques assurées par le composant.
- Son implémentation :
  - Par un diagramme de classes,
  - Par un digramme de collaboration représentant les interactions entre les classes,
- Son interface :

- Qui est représentée par des classes d'interfaces et les scénarions de leur utilisation par des acteurs métier dans des diagrammes de séquences.

IV.4.5. COMPOSANTS DE GESTION ET DE CONCEPTION DES PROCESSUS

a) Objectifs

Ce composant permet aux acteurs métier de développer des modèles d'un système par ses processus métier et ses fonctionnalités, appelés activités.

En effet, en partant du cahier de charges, le composant permet de définir l'objectif du système et de le décomposer en domaines tout en décomposant l'objectif-système (§ Annexe B).

Les acteurs de type responsables CIMOSA et responsables métier, après la définition du système et des domaines, développent une cartographie des processus métier des différents domaines.

Les processus métier étant définis, les responsables métier, les experts et les techniciens collaborent pour définir les fonctionnalités (activités) du système. Ces fonctionnalités forment les étapes des processus métier.

La Figure IV.4-2 résume ces tâches génériques par un diagramme de cas d'utilisation.

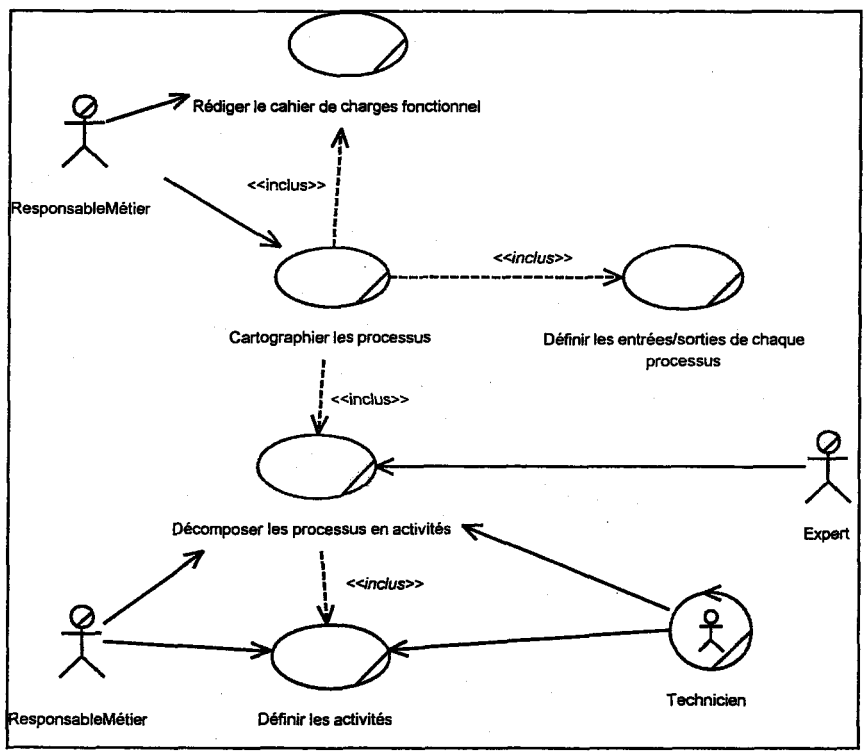

Figure IV.4-2 – Cas d'utilisation du composant gestion et conception de processus

b) Concepts

En partant des aspects fonctionnels du méta-modèle (Figure IV.3-12), les attributs et les méthodes des concepts de ces aspects sont représentés par les Figures IV.4-3 et IV.4-4.

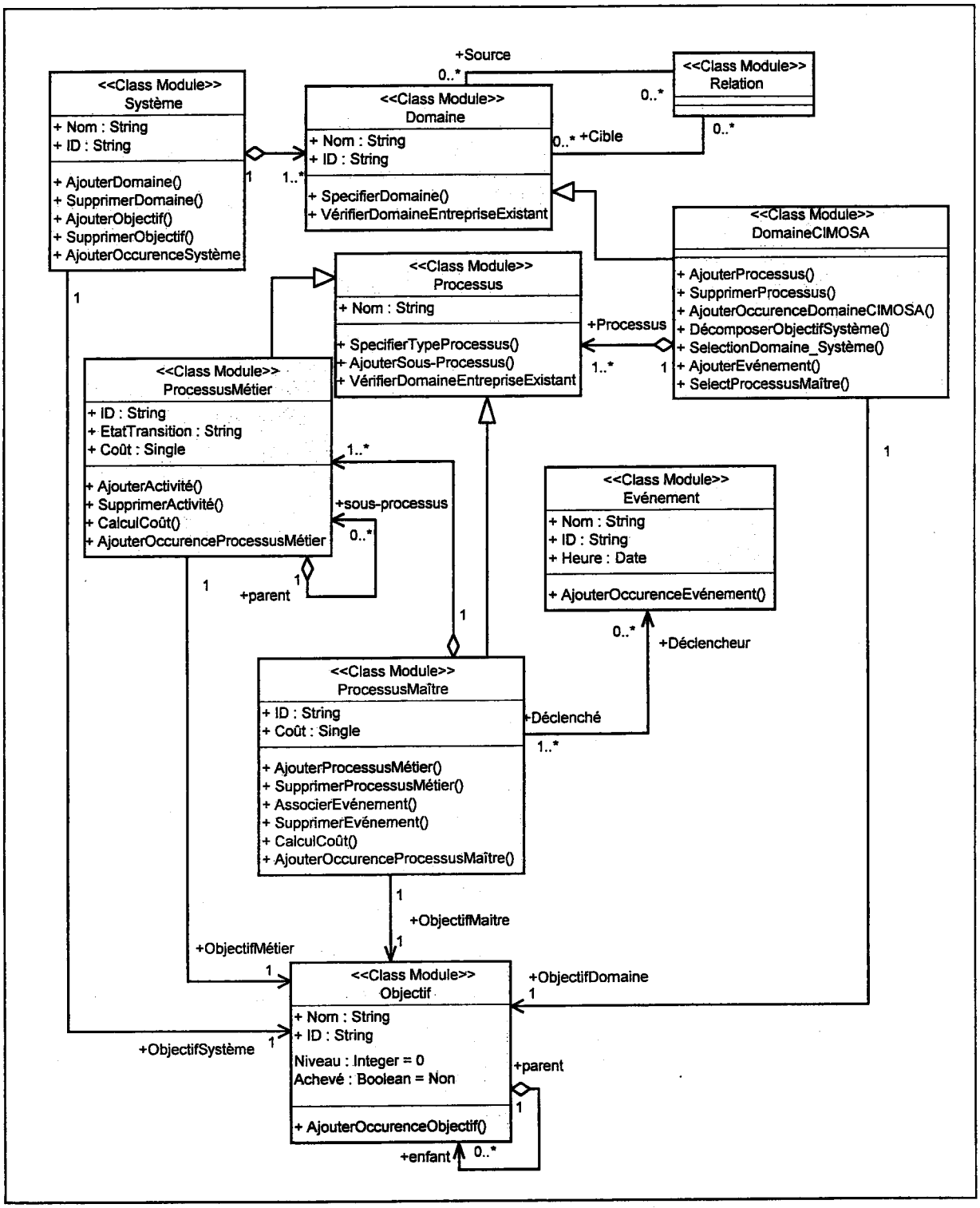

Figure IV.4-3 – Diagramme de classes du composant Gestion et conception de Processus

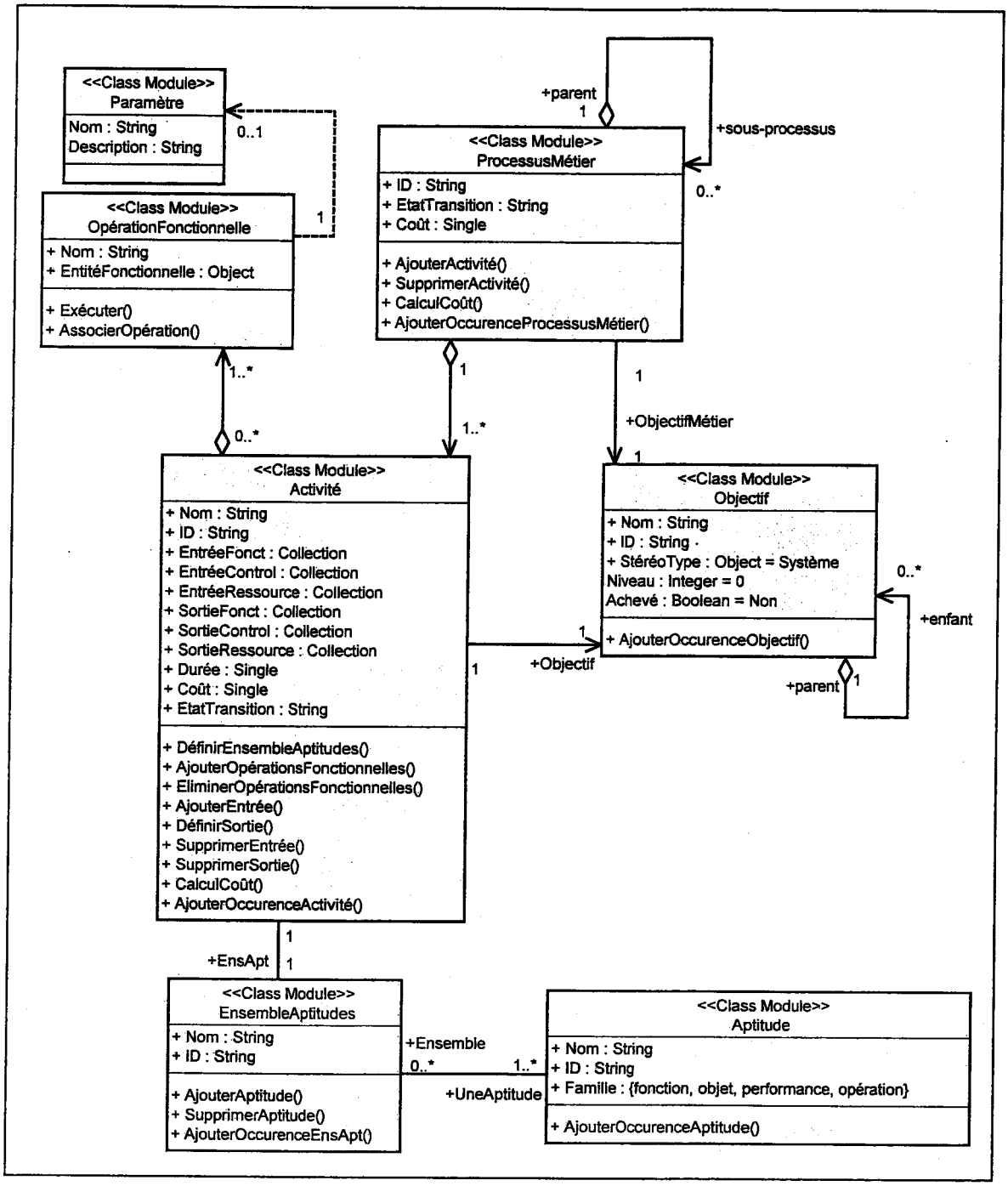

Figure IV.4-4 – Diagramme de classes du composant Gestion et conception de Processus (suite)

c) Collaboration entre les concepts du composant

Les différentes méthodes des classes des Figures IV.4-3 et IV.4-4 permettent d'identifier les interactions entre les instances de ces classes comme représenté dans la Figure IV.4-5.

La classe processus permet d'identifier des processus d'un domaine, puis de définir son type (processus métier ou processus maître) en fonction de ses objectifs. La définition du type d'un processus ne peut avoir lieu qu'après identification complète des processus du domaine, qui est suivie par le regroupement des processus en processus parent, jusqu'au niveau le plus haut. Les processus parents sont des processus maîtres, alors que les processus des niveaux les plus bas sont les processus métier.

Il est possible de rectifier les regroupements des processus métier en processus maîtres, en supprimant et ajoutant des processus.

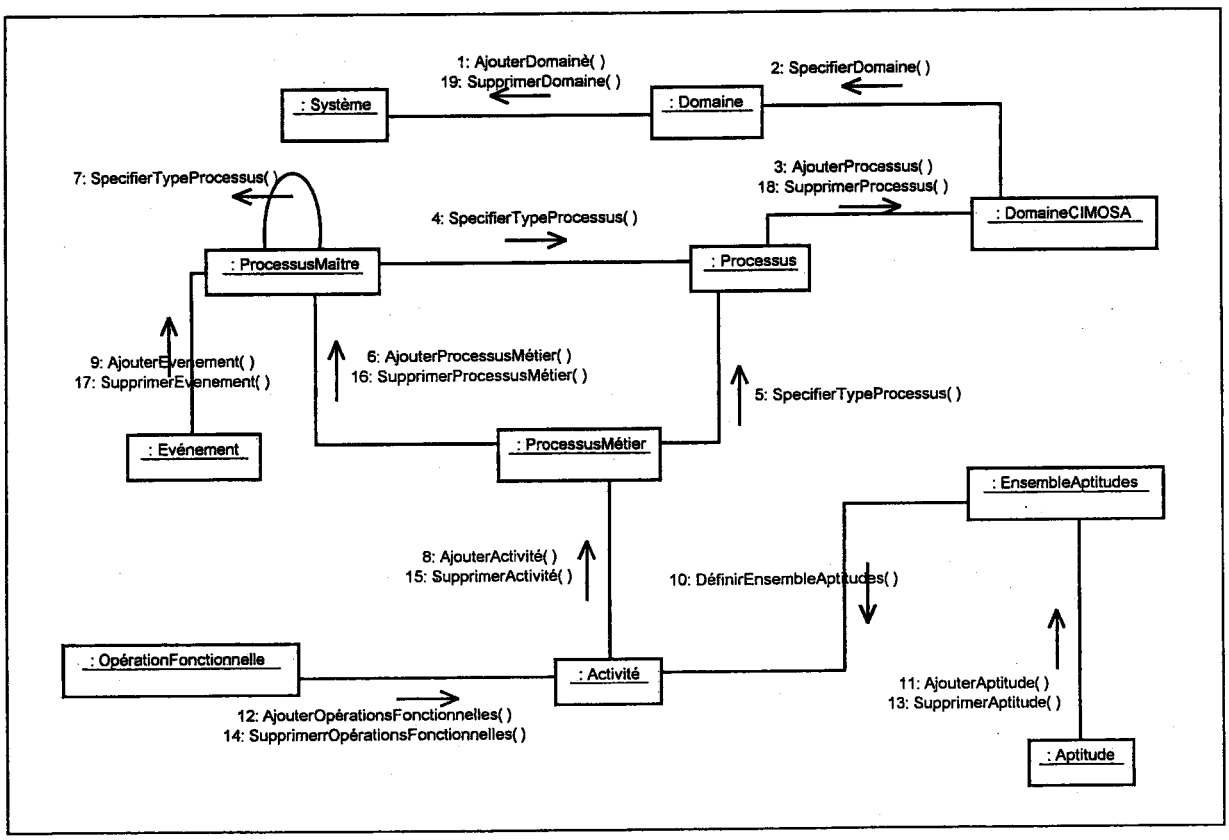

Figure IV.4-5 – Collaboration entre instances

Comme cité dans les cas d'utilisation du composant, les responsables métier et responsables CIMOSA effectuent la définition du système et de ses domaines, ensuite des processus de chaque domaine, en spécifiant les processus métier (cartographie des processus).

La définition des entrées/sorties et des ressources des processus métier est une étape importante lors de la préparation de la cartographie [Mougin, 02].

d) Interfaces du composant

Les acteurs utilisent l'interface du composant (Figure IV.4-6) pour manipuler ses concepts. L'interface est représentée par les classes interfaces suivantes :

- La classe interface 'ISystème' permet aux responsables métier et aux responsables CIMOSA d'identifier et de définir un système, ses objectifs, la liste de domaines qu'il peut contenir et les relations entre eux. Cette classe permet la manipulation d'objets Système, Relation et Domaine CIMOSA (Figure IV.4-7).
- La classe interface 'IGestionProcessus' permet aux responsables métier de gérer les processus métier et les regrouper en processus maîtres (Figure IV.4-8).
- La classe interface 'IConceptionProcessus' permet aux responsables métier, aux experts, aux responsables cellule et aux responsables d'organisation de définir les processus maîtres et métier, de décrire leurs objectifs correspondants, de définir les activités de chaque processus métier, d'identifier les entrées et les sorties des processus/activités et les cellules et acteurs responsables d'eux (Figures IV.4-9, IV.4-10 et IV.4-11).
- 'IActivité' est une interface permettant aux responsables métier, techniciens, experts, et responsables d'organisation de définir et de décrire les activités, les aptitudes requises et les ressources correspondantes tout en identifiant les opérations fonctionnelles achevant chaque activité (Figure IV.4-10).

Ces quatre classes interfaces permettent de définir l'interface du composant et permettent aux différents acteurs métier de gérer et de concevoir des objets processus, activités, objectifs, domaines, etc. Or, le composant ne permet que la description statique des processus et des activités. Par conséquent, nous avons développé un cinquième composant qui permet la description du comportement d'un processus par les règles de comportement (§IV.4.9).

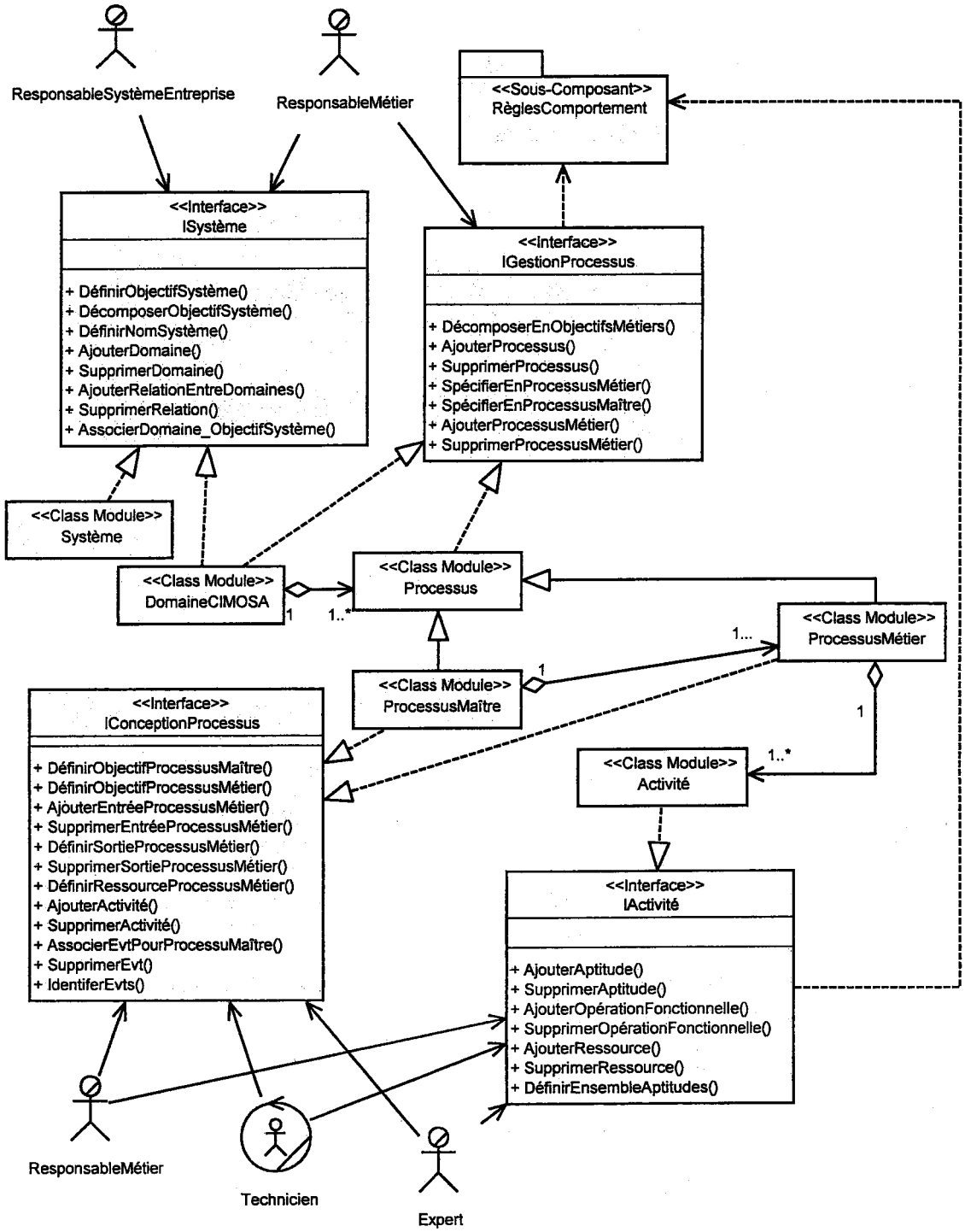

Figure IV.4-6 – Interface du composant de gestion et de conception des processus

Pour décrire les scénarios possibles lors de l'utilisation de chacune des interfaces par le modélisateur, nous avons représenté ceux-ci par des diagrammes de séquence.

En effet, dans la IV.4.8 les responsables métier décomposent chaque objectif correspondant à un domaine en objectifs métier. En partant de ces objectifs, ils définissent des processus et les spécifient en processus métier. Ensuite, ils regroupent ces processus en processus maîtres.

Il est possible qu'un processus métier soit redéfini en tant que processus maître. Ceci est dans le cas où il est le processus du plus haut niveau du domaine et pourrant être déclenché par des événements.

De même, les Figures IV.4-7, IV.4-9 et IV.4-10 représentent des séquences définies par des opérations en ordre.

Ces diagrammes de séquence permettent de décrire la méthodologie que nous proposons pour utiliser le composant Gestion & Conception de Processus, tout en identifiant les acteurs métier qui doivent y avoir accès.

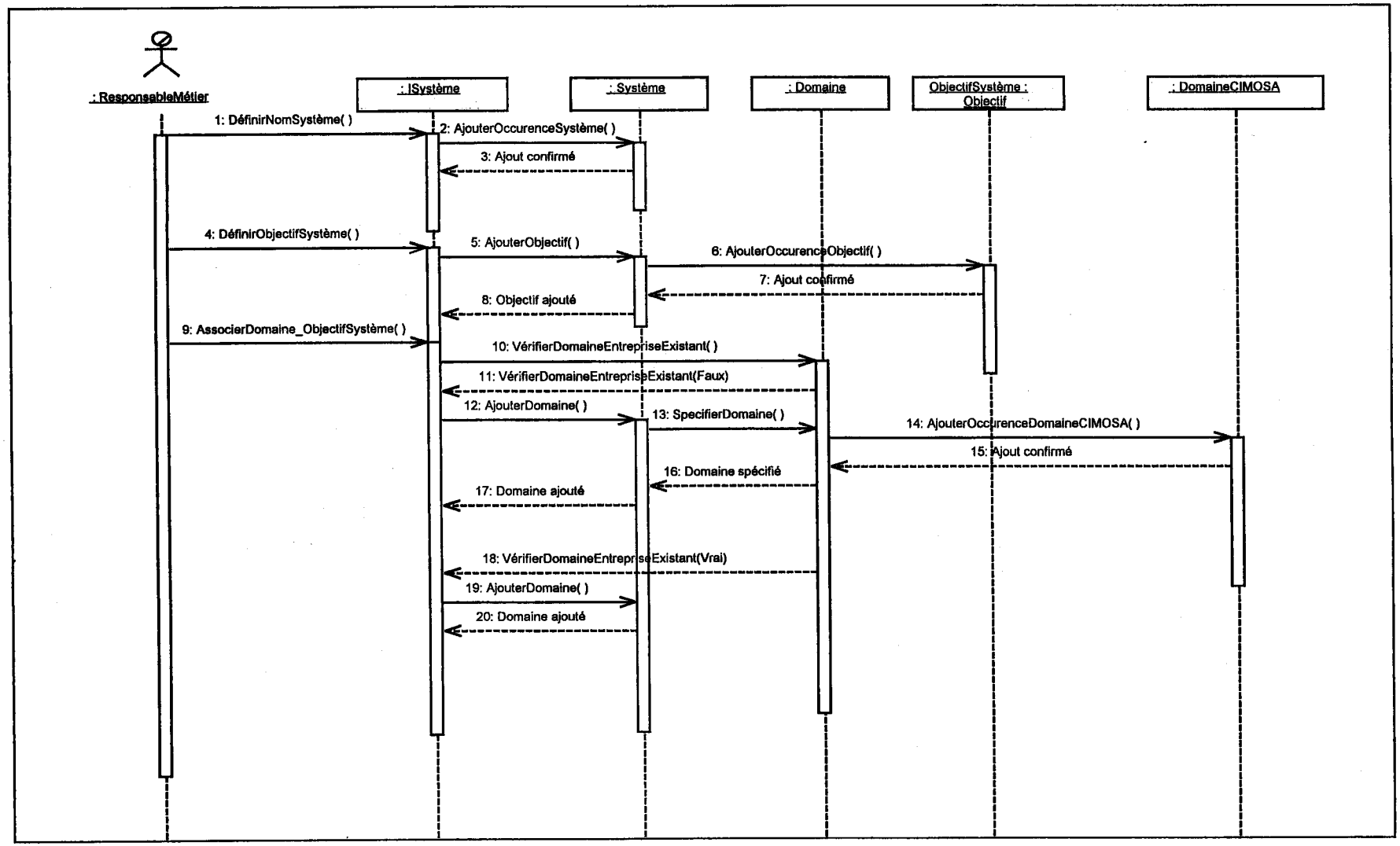

Figure IV.4-7 – Utilisation de la classe interface ISystème

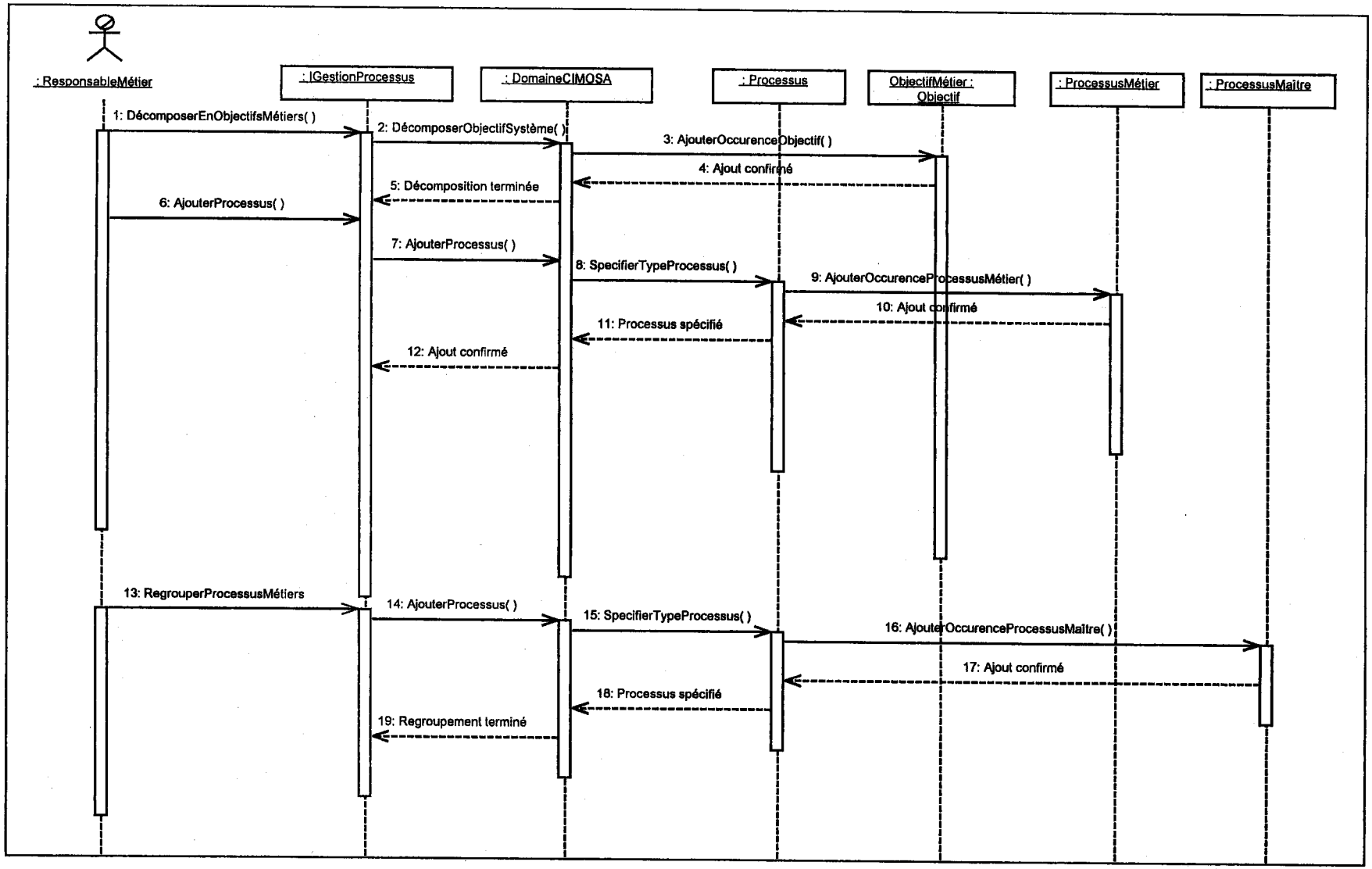

Figure IV.4-8 – Utilisation de la classe interface IGestionProcessus

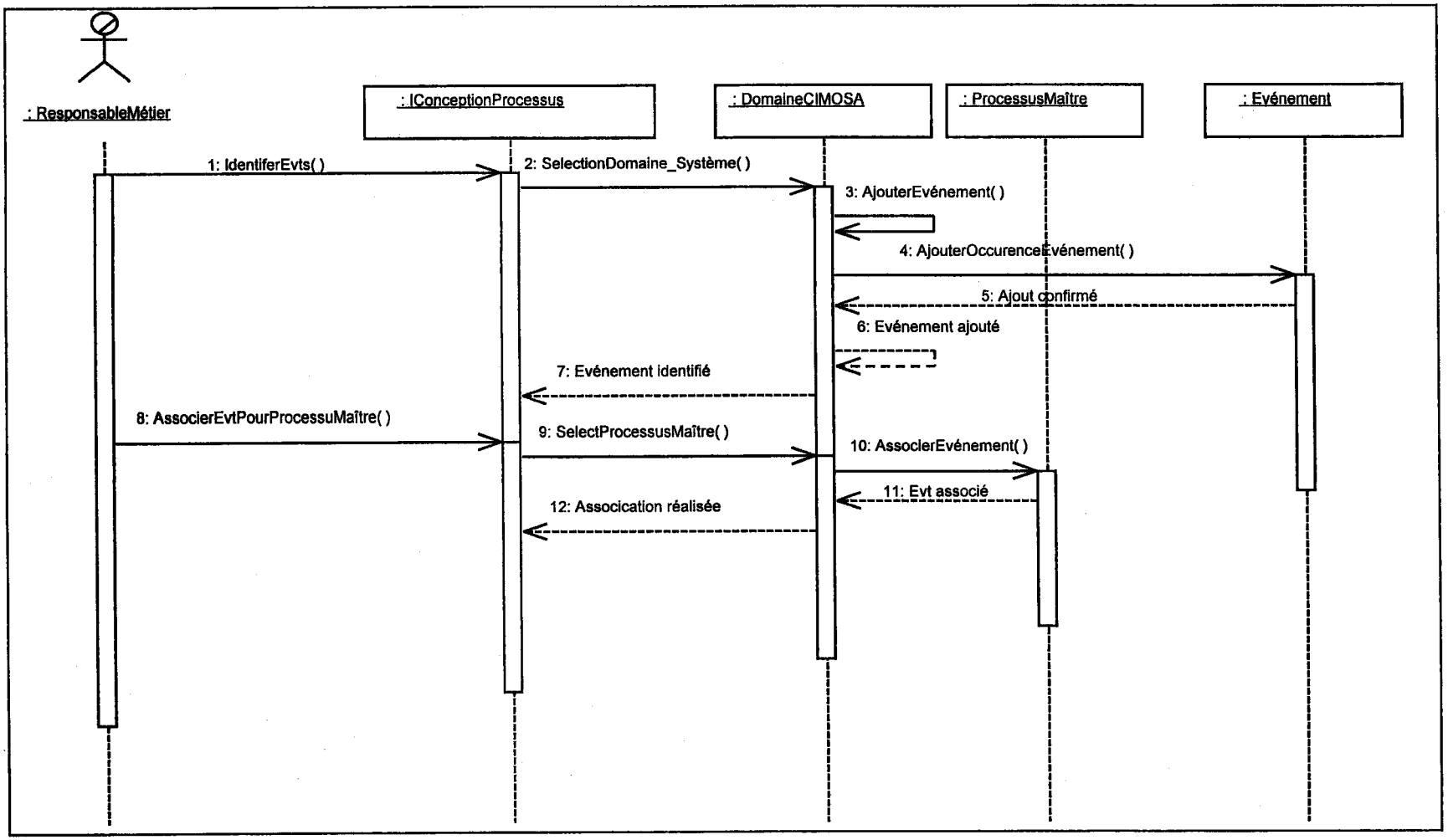

Figure IV.4-9 – Utilisation de la classe interface IConceptionProcessus

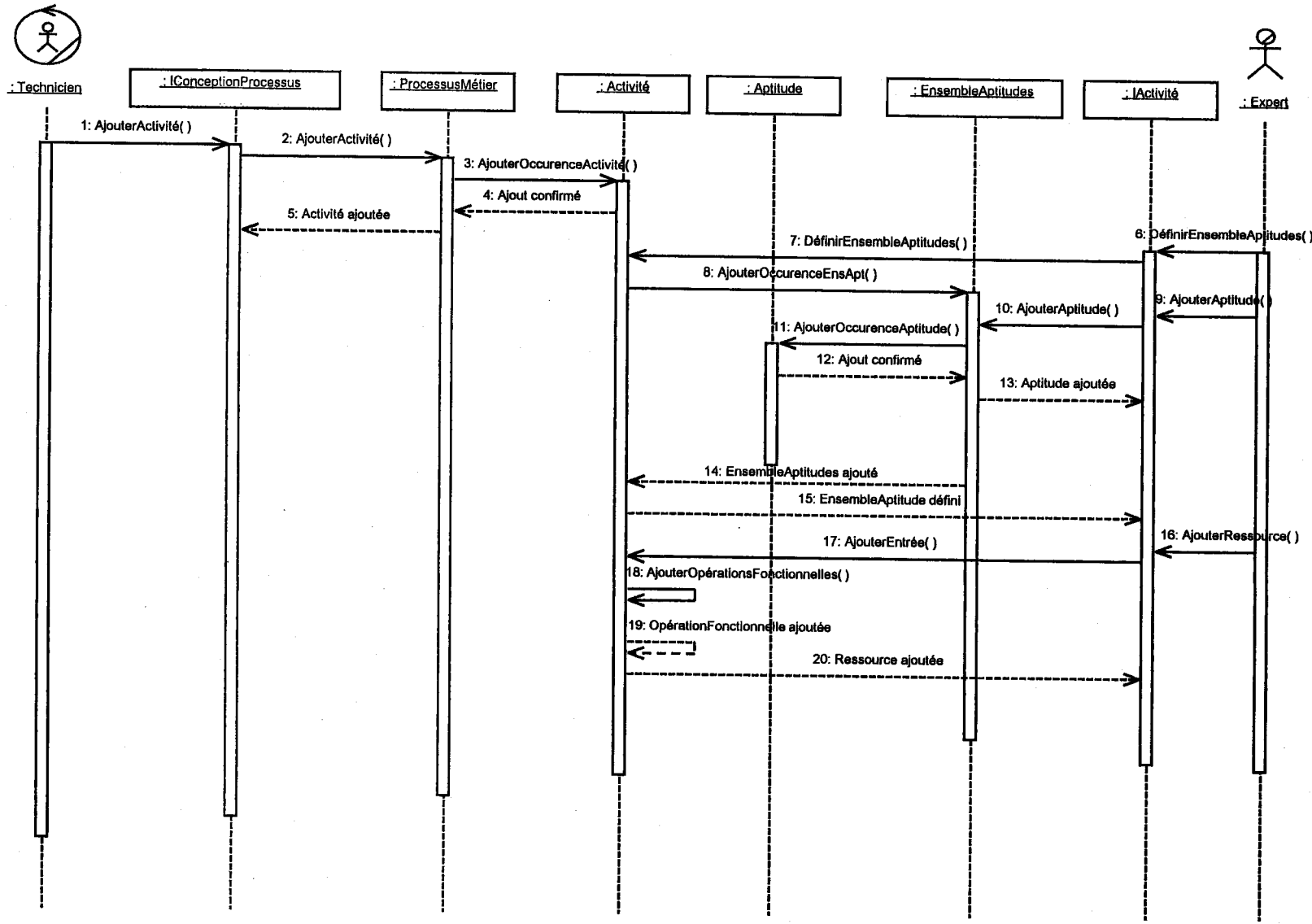

Figure IV.4-10 – Utilisation de la classe interface IActivité

En final, nous pouvons représenter le composant de Gestion & de Conception de Processus et ses interfaces comme proposé dans la Figure IV.4-11 :

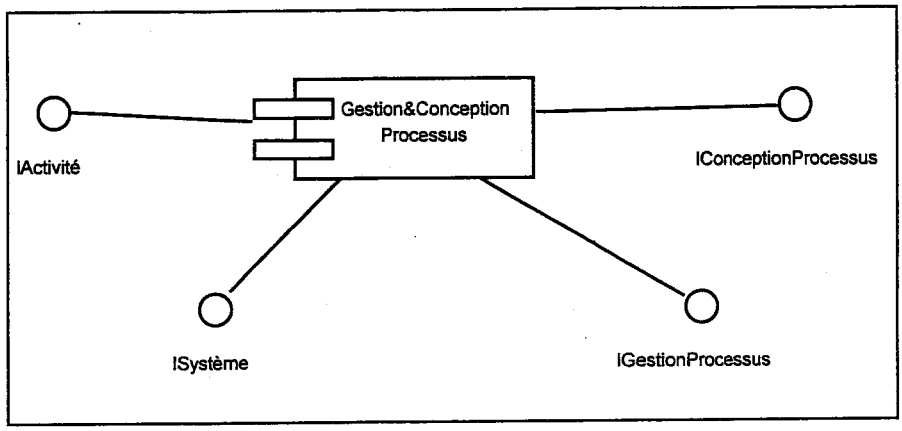

Figure IV.4-11 – Composant de « Gestion & de Conception de Processus »

IV.4.6. COMPOSANT DE GESTION DE L'INFORMATION

a) Objectifs

Ce composant permet aux techniciens de récupérer les informations et les données d'entreprise pour décrire les objets d'entreprise, leurs propriétés et leurs états.

Ces objets d'entreprise sont gérés par les responsables métier pour développer des schémas conceptuels, dans le but de réaliser des bases de données.

La Figure IV.4-12 représente le digramme de cas d'utilisation qui définit les tâches génériques à réaliser par les acteurs métier. Ce diagramme est la spécialisation du cas d'utilisation « Identifier les entrées/sorties d'un processus » de la Figure IV.2-2.

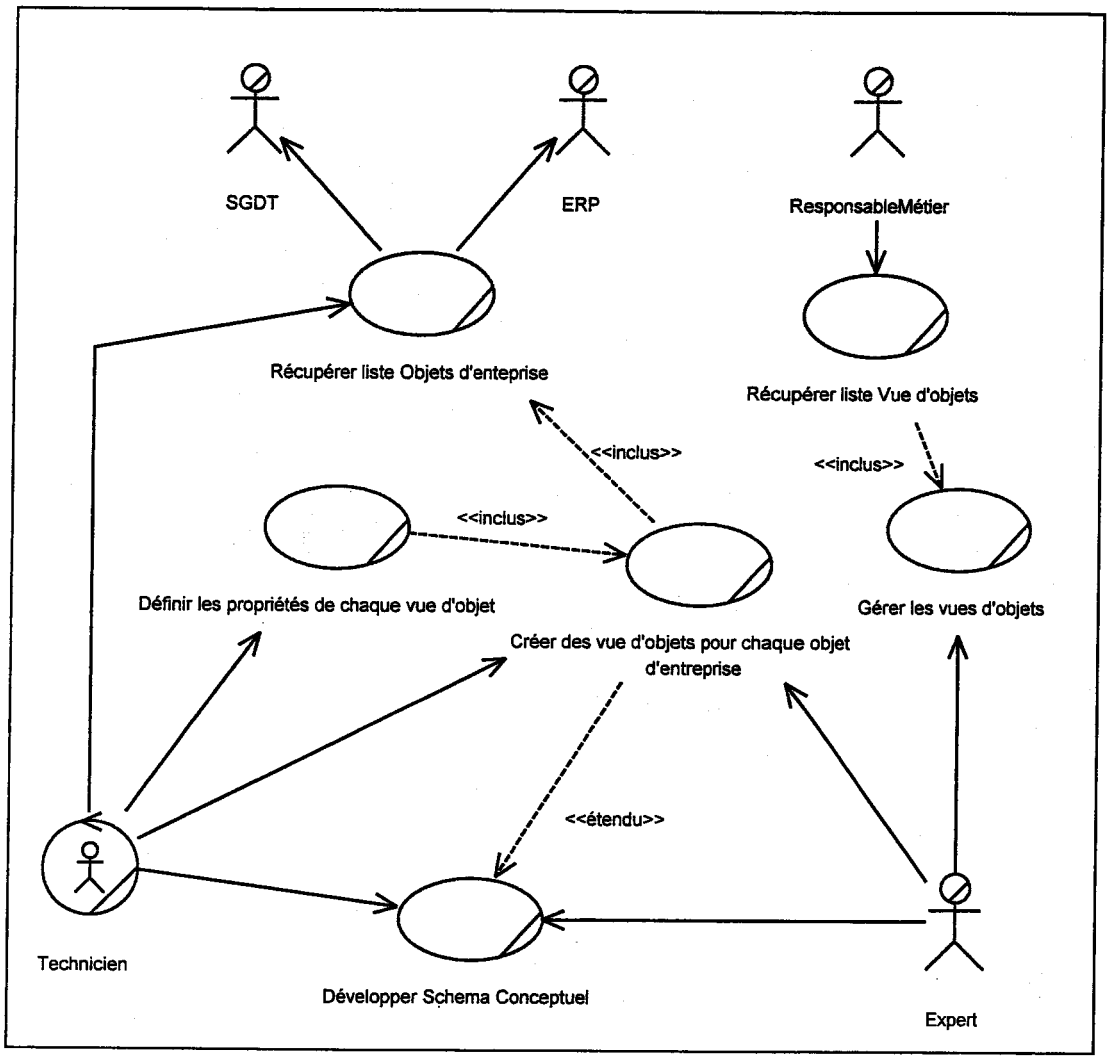

Figure IV.4-12 – Cas d'utilisation du composant Gestion de l'Information

b) Concepts

Les aspects informationnels du méta-modèle rassemblent les concepts implémentés par ce composant.

La Figure IV.4-13 représente les attributs et les méthodes de ces concepts.

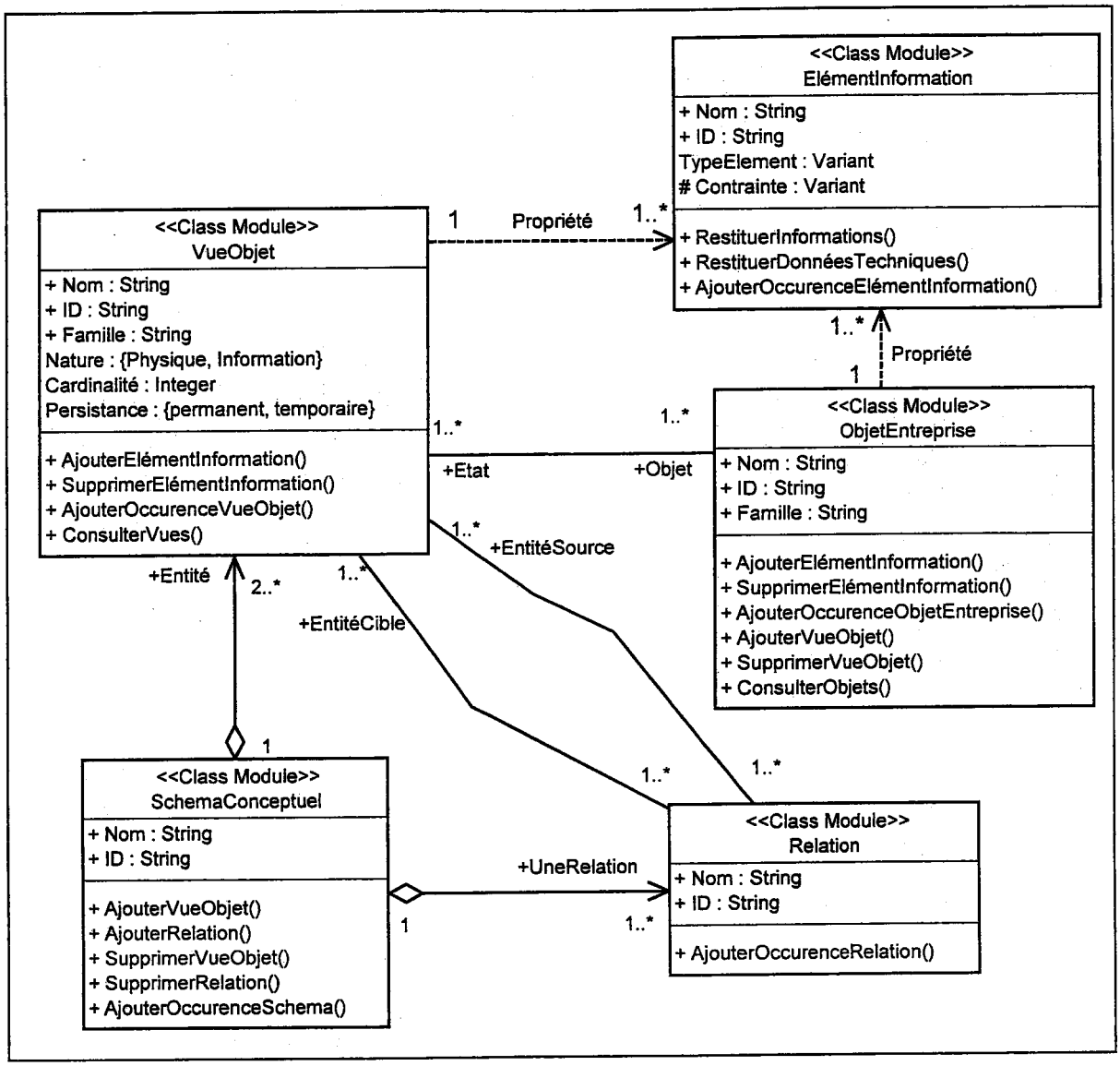

Figure IV.4-13 – Diagramme de classes du composant Gestion de l'Information

c) Collaboration entre les concepts du composant

Ces concepts sont représentés par des classes d'objets. Les instances de ces classes collaborent entre eux par envoi de messages et appel de méthodes (Figure IV.4-14).

En effet, après l'identification des objets d'entreprise, des vues d'objet sont définies pour représenter un ou plusieurs de ces objets.

Ces vues d'objet sont utilisées en tant qu'entités pour créer des schémas conceptuels.

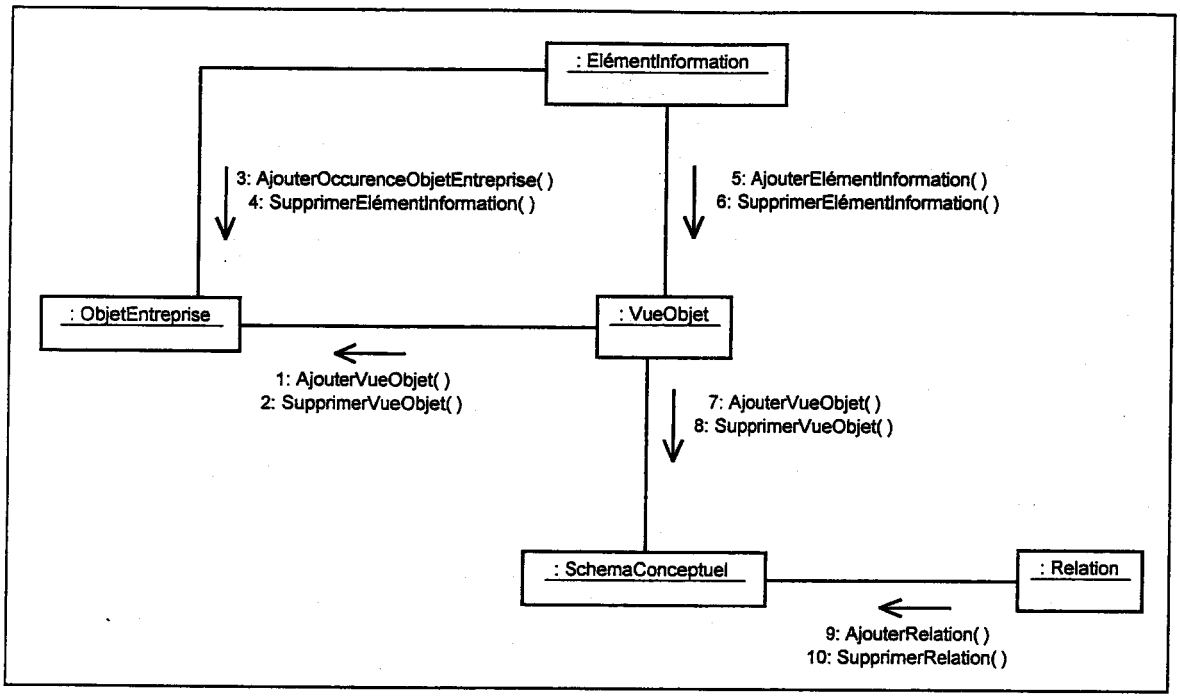

Figure IV.4-14 – Collaboration entre instances

d) Interfaces du composant

Pour avoir accès à ce composant, les acteurs manipulent son interface. Cette dernière est présentée par les deux classes interfaces (Figure IV.4-15) suivantes :

- IGestionInformation qui aide les experts et les techniciens à créer des schémas conceptuels tout en établissant la liste des objets d'entreprise et créeant les vues d'objet qui leur correspondent et de les stocker dans des bases de données techniques ou dans des ERP (Figure IV.4-16).
- IElémentInformation qui permet aux techniciens de définir les propriétés des objets d'entreprise et les vues d'objet(Figure IV.4-17).

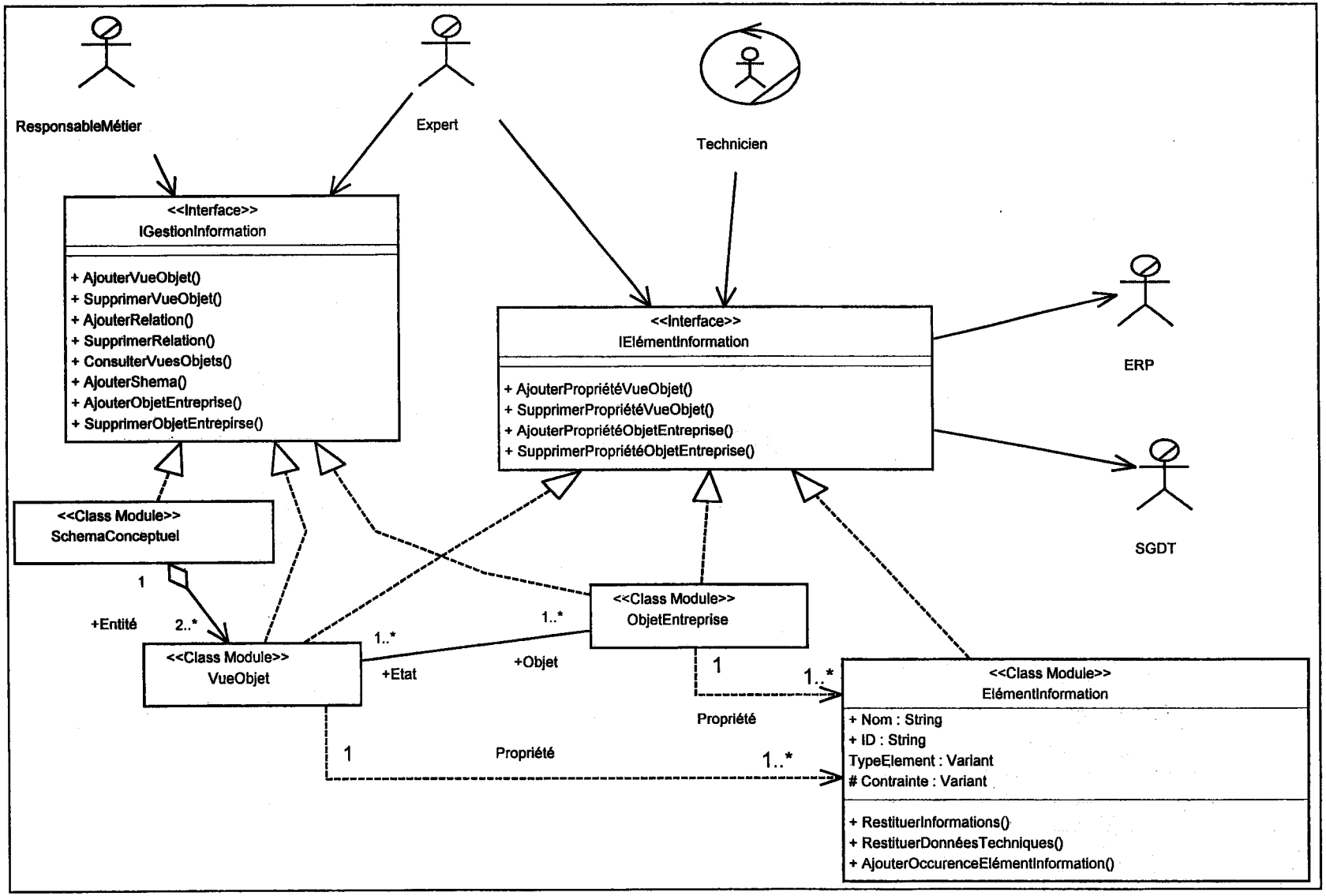

Figure IV.4-15 – Interface du composant Gestion de l'Information

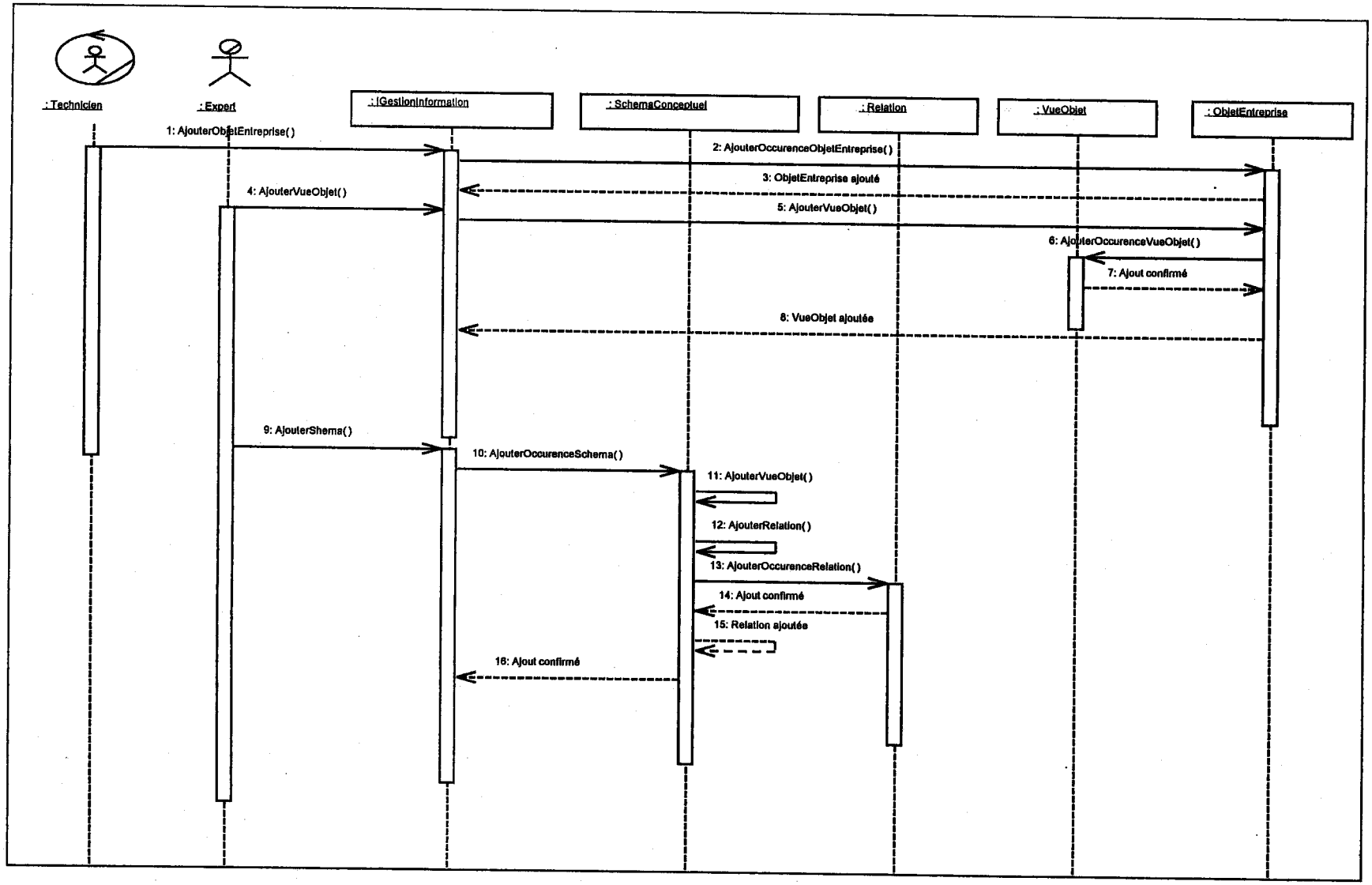

Figure IV.4-16 – Utilisation de la classe interface IGestionInformation

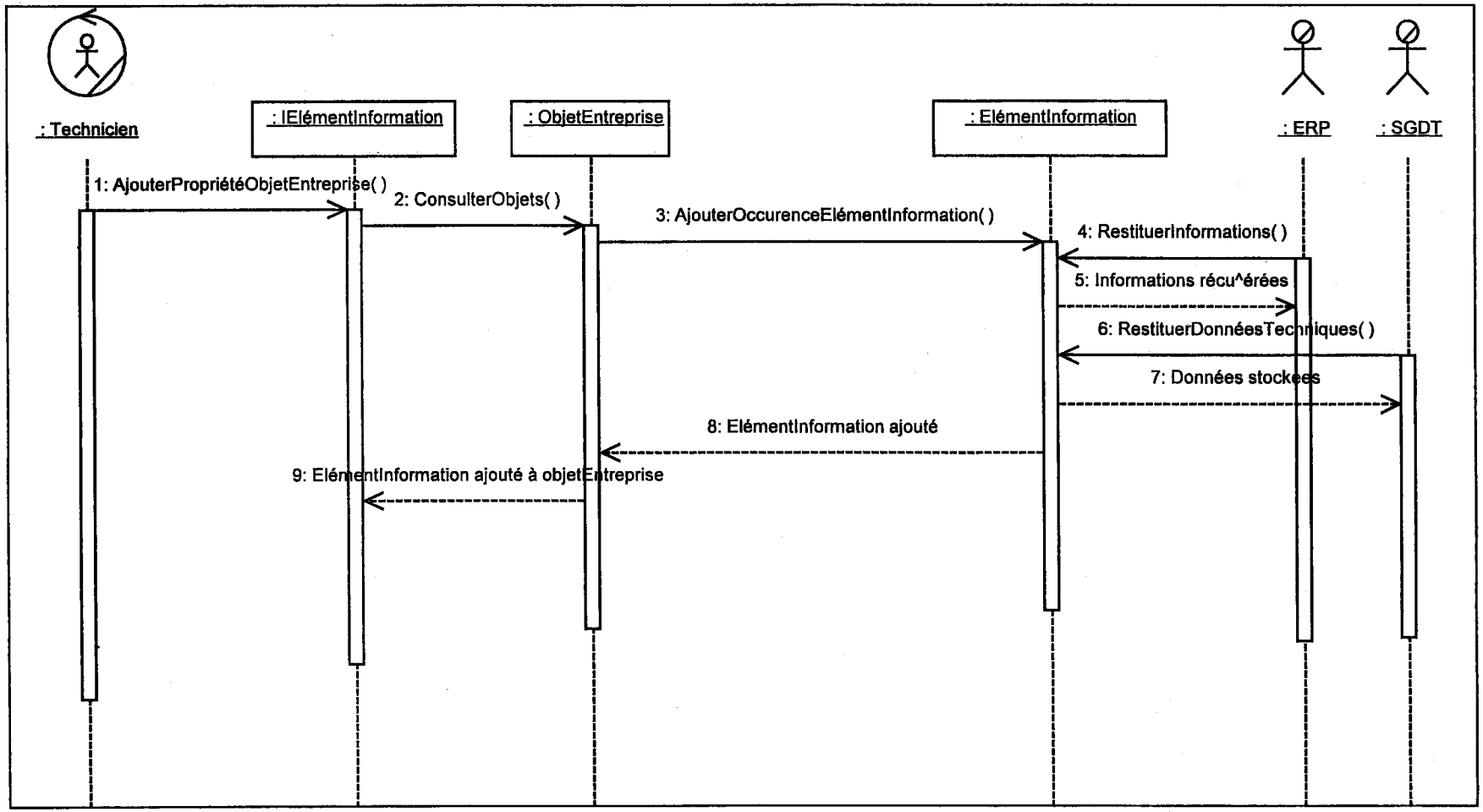

Figure IV.4-17 – Utilisation de la classe interface IElementInformation

Finalement, nous représentons le composant de gestion de l'information et son interface (Figure IV.4-18)

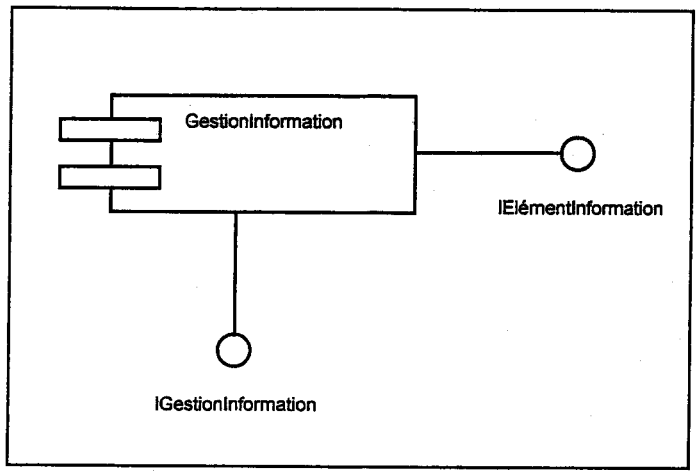

Figure IV.4-18 – Composant « Gestion d'Information »

Ce composant peut être utilisé pour associer les vues d'objet en tant qu'entrées et sorties des activités d'entreprise.

IV.4.7. COMPOSANT DE GESTION DES RESSOURCES

a) Objectifs

L'objectif de ce composant est de permettre aux acteurs métier de définir les ressources d'entreprise, pour servir aux activités. Cette association est en fonction des aptitudes requises.

En effet, les techniciens et les experts récupèrent les informations des ressources pour définir des instances de ressources et définissent leurs aptitudes (Figure IV.4-19).

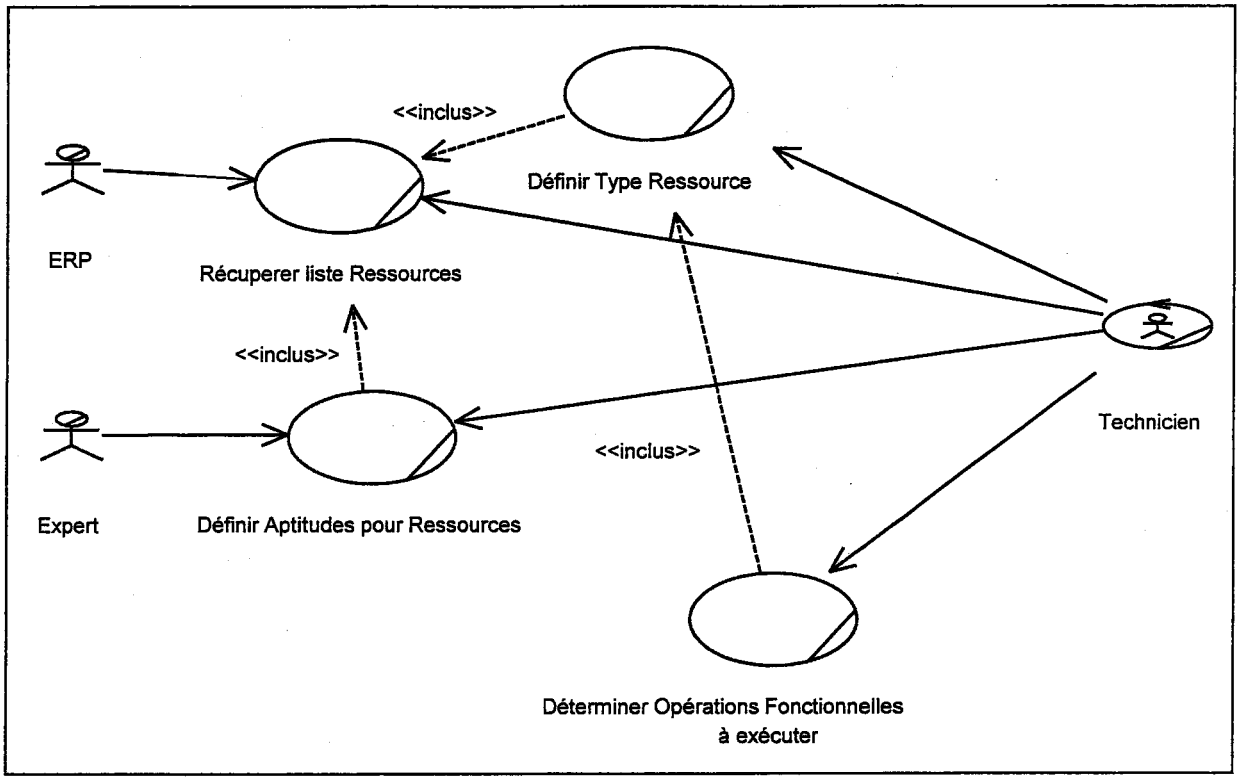

Figure IV.4-19 – Cas d'utilisation du composant Gestion de Ressources

Le diagramme de cas d'utilisation de la Figure IV.4-19 est la spécialisation du cas d'utilisation « Définir les ressources requises pour chaque processus » de la figure IV.2-2.

b) Concepts

La Figure IV.4-20 montre les attributs et les méthodes des classes qui représentent les concepts des aspects ressources du méta-modèle.

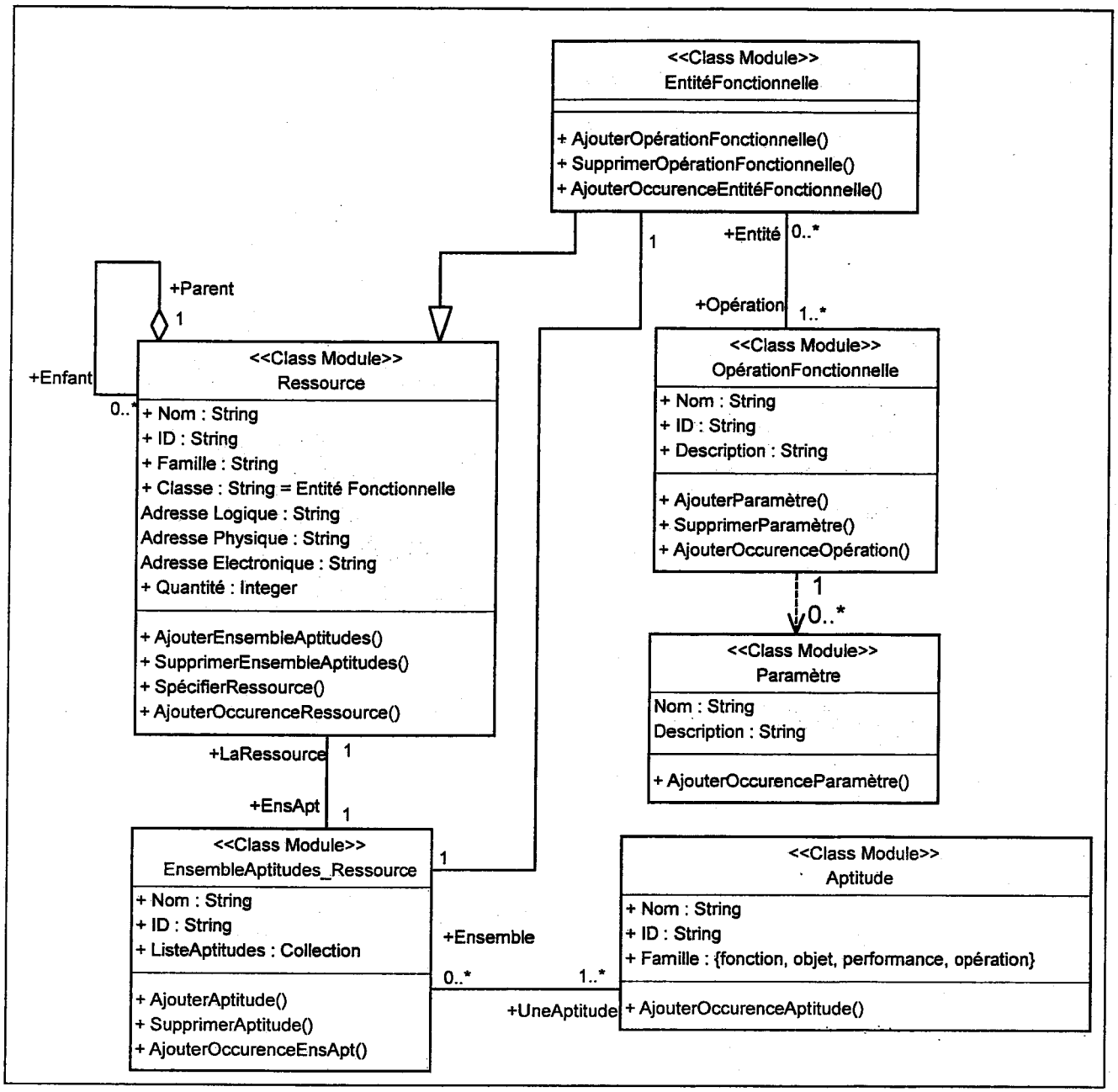

Figure IV.4-20 – Concepts du composant Gestion de Ressources

c) Collaboration entre les concepts du composant

Les instances de classes de ce composant collaborent entre eux par des appels des méthodes (Figure IV.4-20) définis ci-dessus.

Après la définition des ressources, les ensembles d'aptitudes peuvent être définis. Ensuite, les entités fonctionnelles sont identifiées parmi ces ressources. Enfin, les opérations fonctionnelles qui doivent être assurées par les entités fonctionnelles sont identifiées.

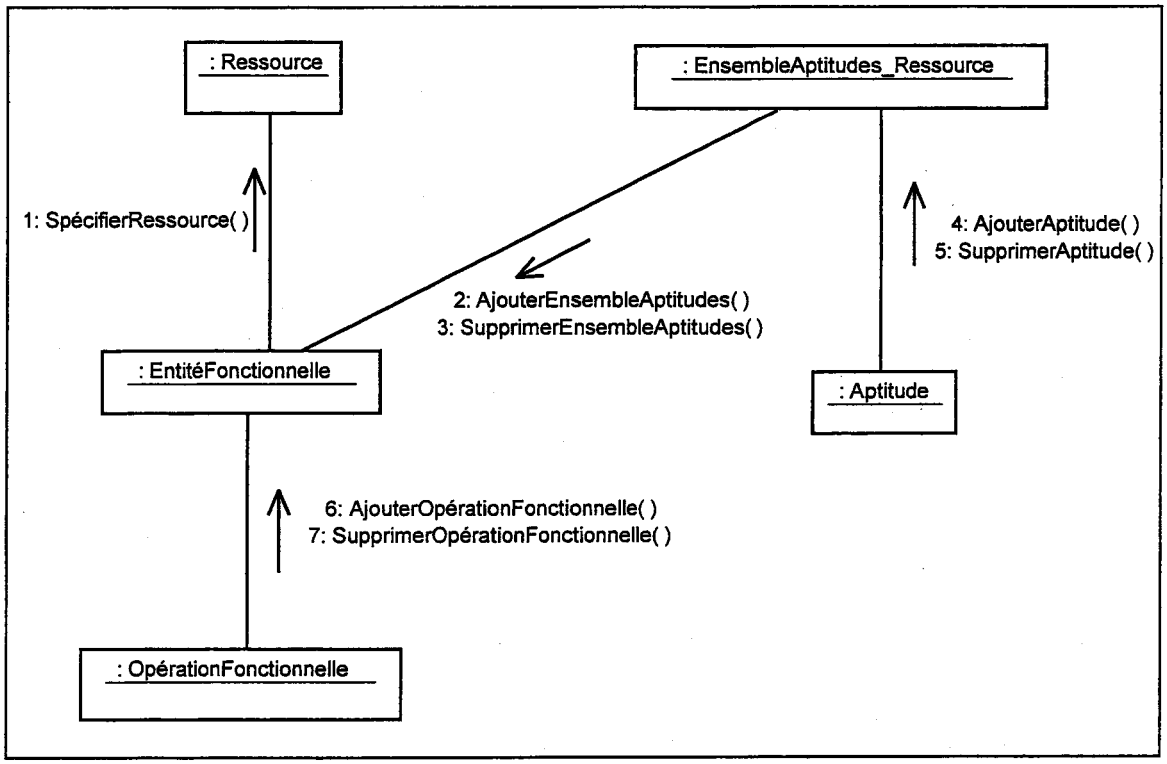

Figure IV.4-21 – Collaboration entre les instances

d) Interfaces du composant

La récupération des informations concernant les moyens de l'entreprise et la création d'instances ressources sont réalisées par les techniciens, par le biais de la classe interface IInformationRessource (Figures IV.4-22 et IV.4-23).

Ensuite, les experts définissent les aptitudes de chaque instance ressource, à travers la classe interface IGestionRessources. Cette même interface est utilisée par les techniciens pour identifier les entités fonctionnelles et définir leurs opérations fonctionnelles (Figures IV.4-22 et IV.4-24).

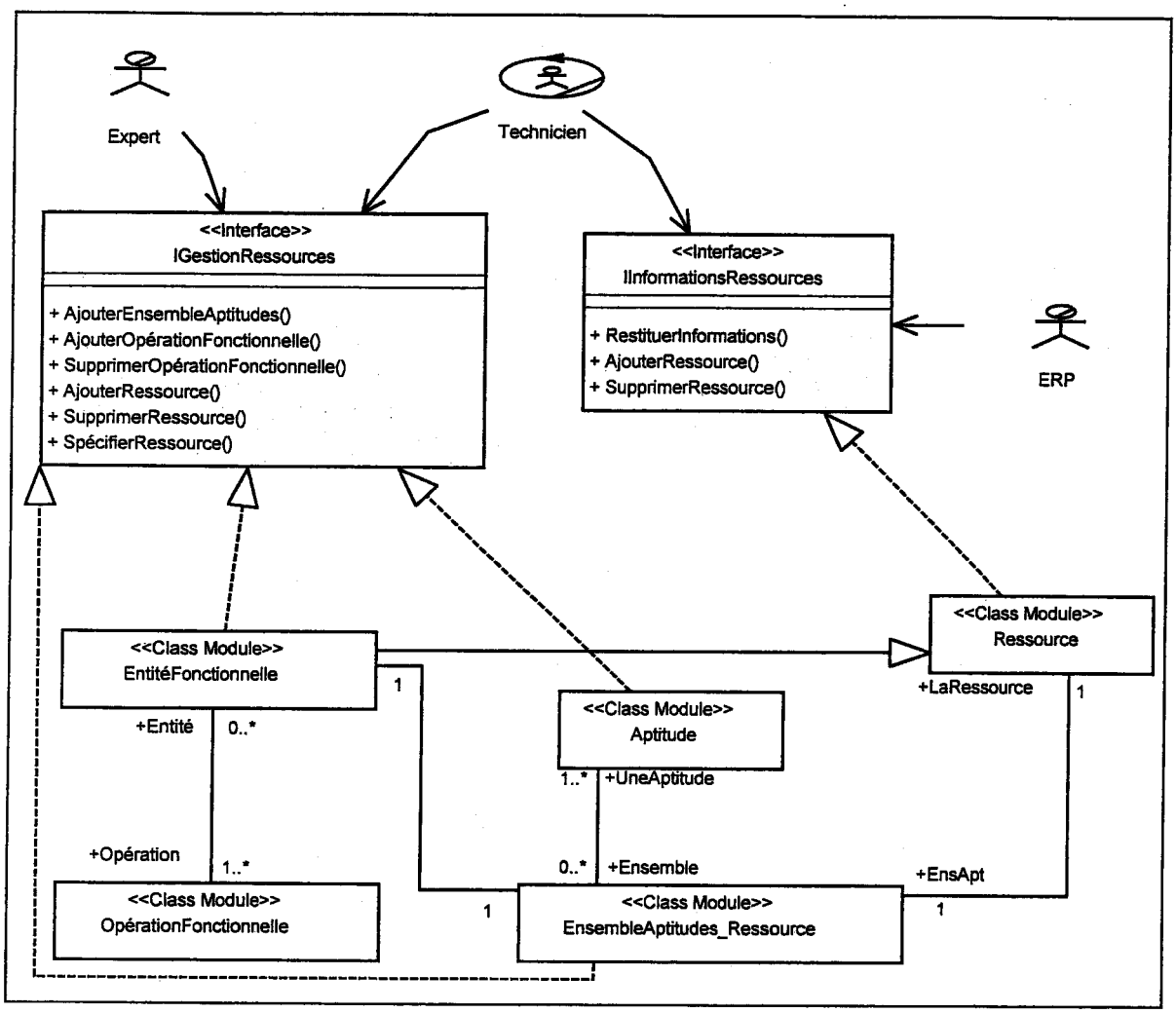

Figure IV.4-22 – Interface du composant Gestion des Ressources

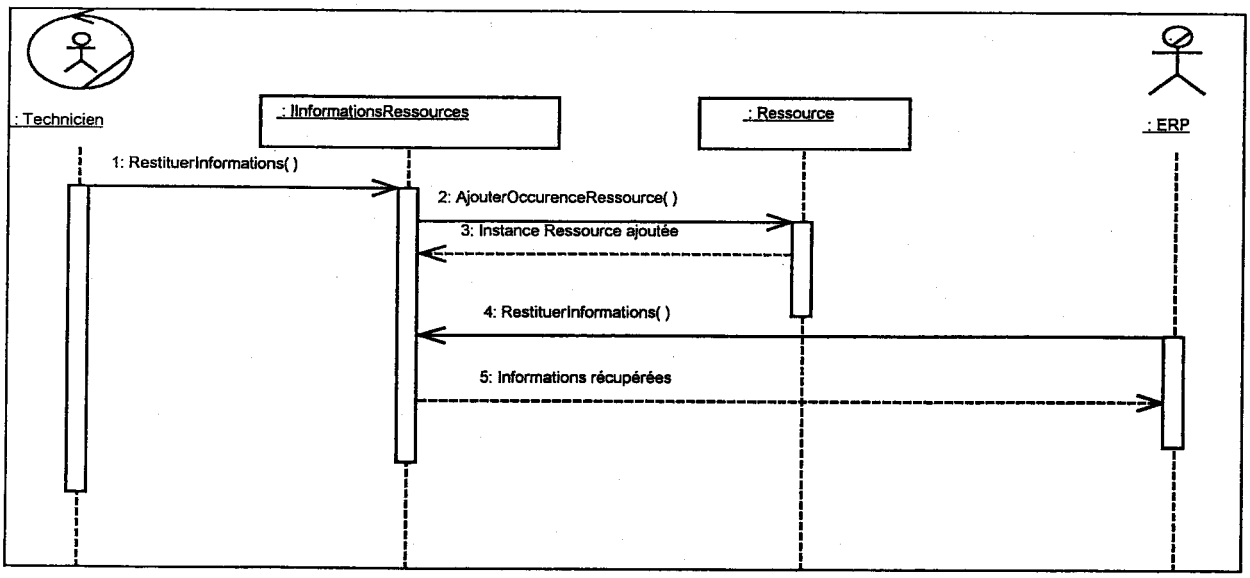

Figure IV.4-23 – Utilisation de la classe interface IInformationRessources

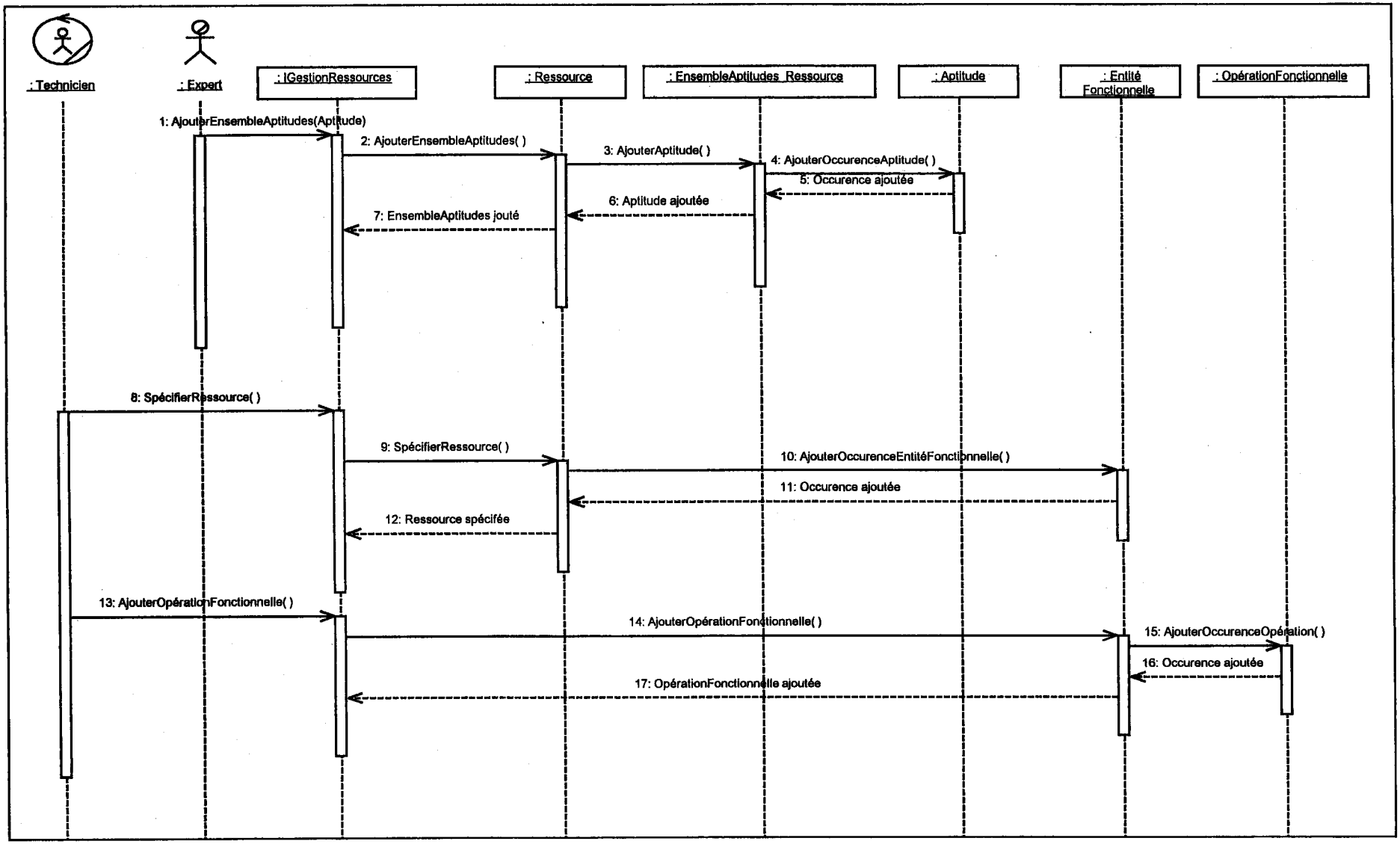

Figure IV.4-24 – Utilisation de la classe interface IGestionRessources

Le composant de gestion de ressources est représenté par la Figure IV.4-25.

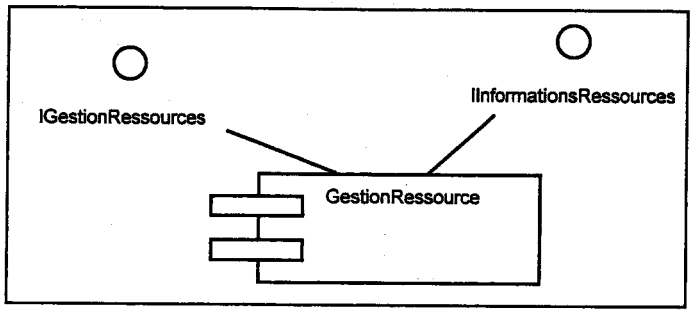

Figure IV.4-25 – Composant « Gestion de Ressources »

Les instances ressources résultant de ce composant peuvent être utilisées en tant qu'entrées de type ressource pour les activités d'entreprise.

IV.4.8. COMPOSANT DE STRUCTURE D'ORGANISATION

a) Objectifs

Avant de réaliser une organisation horizontale de l'entreprise, organisation des processus, il est obligatoire de définir la structure d'organisation du système ou de l'entreprise, ses entités d'organisation (départements, divisions, services, etc.) nommées aussi cellules d'organisation. Les acteurs et les unités d'organisation de chaque cellule sont définis par les responsables de cellules et les responsables d'organisation.

Les responsables métier, les responsables organisation et les responsables cellule définissent les responsabilités et les autorités pour chaque cellule.

Les acteurs sont définis par le terme unité d'organisation.

La Figure IV.4-26 illustre les tâches génériques réalisées par ces acteurs métier.

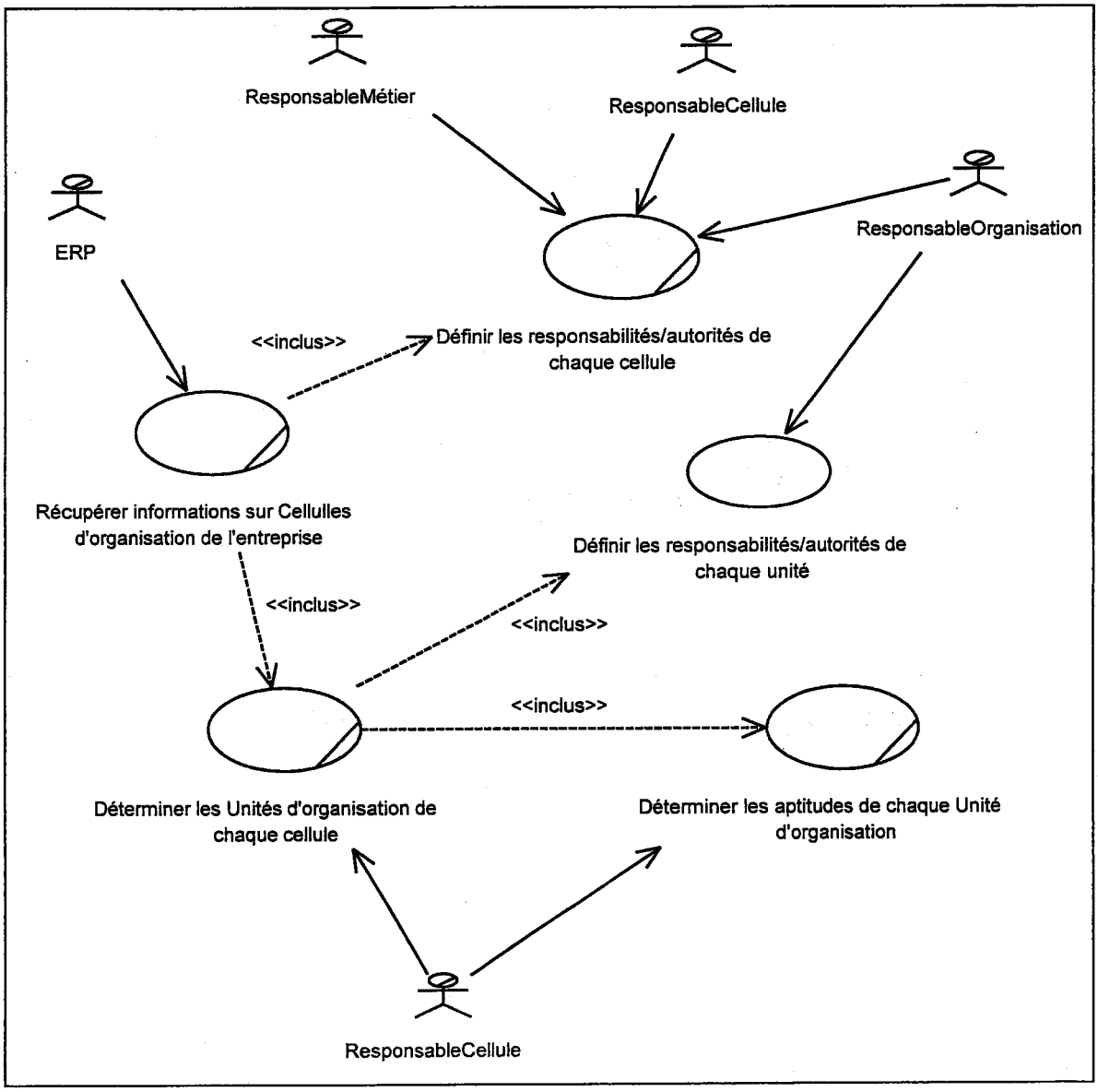

Figure IV.4-26 – Cas d'utilisation du composant Structure d'Organisation

b) Concepts

Ce composant est défini par les concepts « structure d'organisation », « cellule d'organisation », « unité d'organisation », les ensembles d'aptitudes pour chaque « unité d'organisation » et le concept « aptitude ».

Ces concepts sont représentés par des classes d'objet comme illustré dans la Figure IV.4-27.

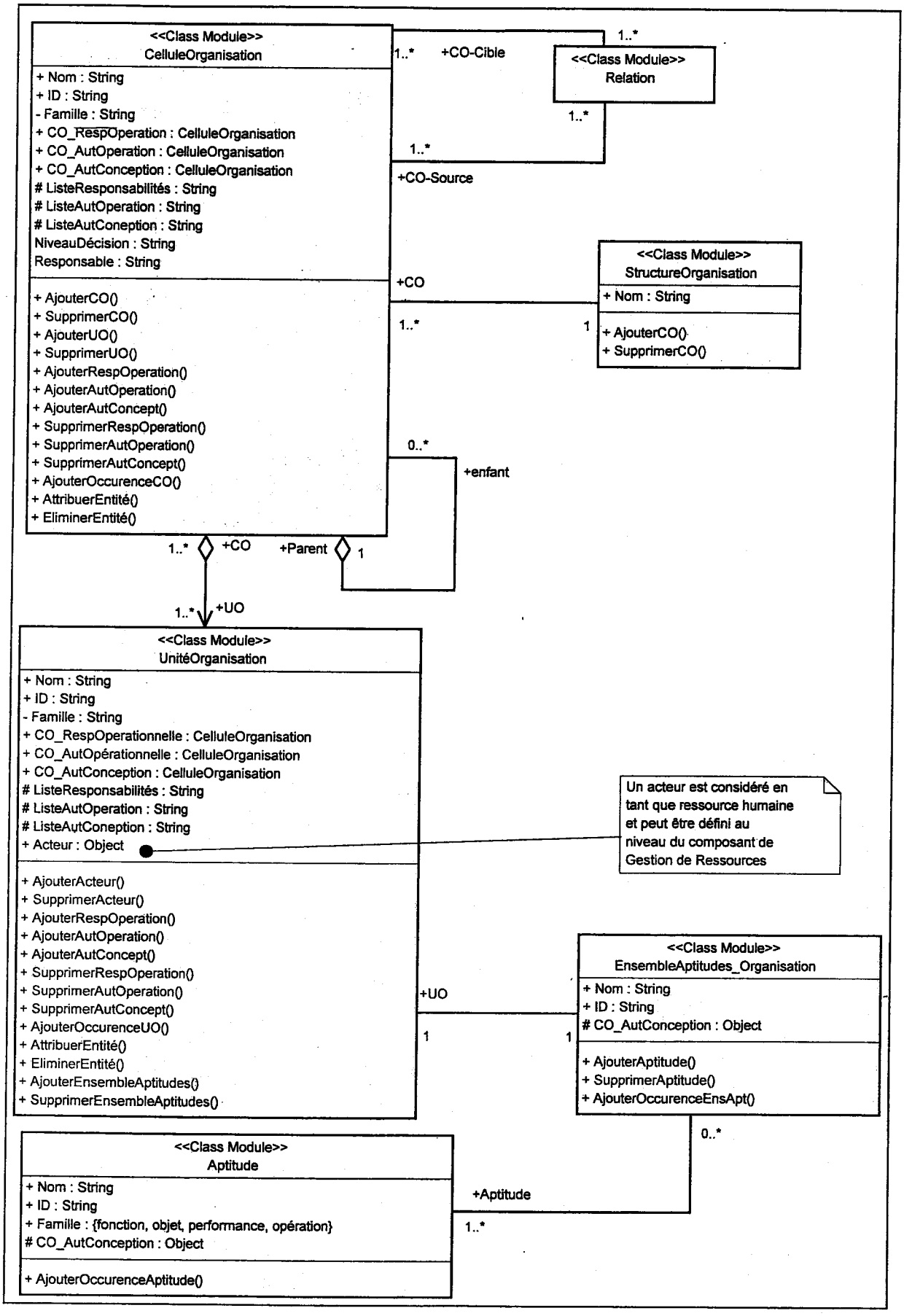

Figure IV.4-27 – Diagramme de classes du composant Structure d'Organisation

c) Collaboration entre les concepts du composant

Pour définir des cellules d'organisation, le concept 'Structure Organisation' permet l'ajout d'occurrences de cellules d'organisation.

Pour chacune de ces dernières, des unités d'organisation peuvent être identifiées et définies (responsabilité opérationnelle, autorité opérationnelle et/ou autorité de conception). Pour chaque unité, des aptitudes doivent être définies.

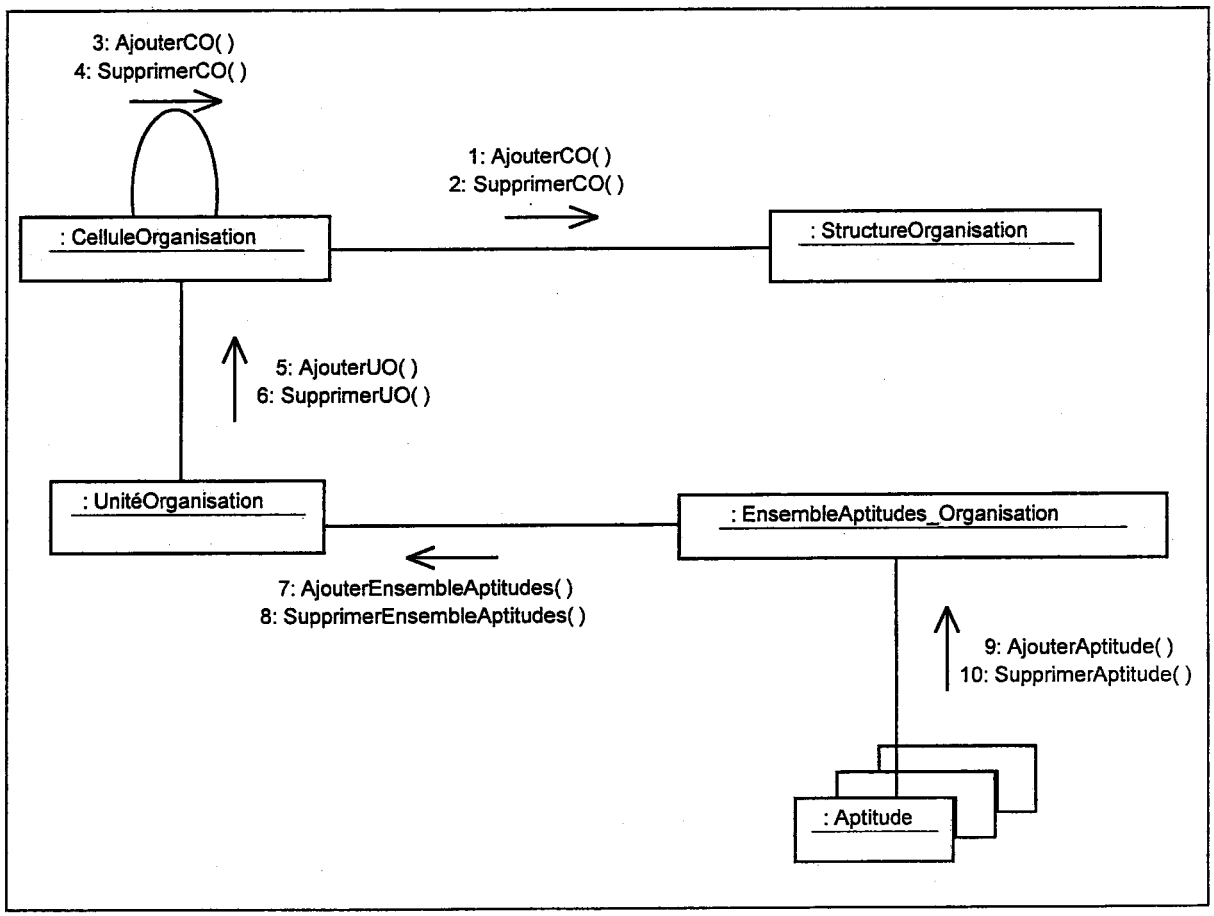

Figure IV.4-28 – Collaboration entre instance

d) Interfaces du composant

L'interface (Figure IV.4-29) de ce composant est implémentée par deux classes interfaces ; IGestionCellules, qui permet de récupérer des informations sur l'organisation de l'entreprise. Ceci est dans le but de définir des cellules d'organisation et des unités d'organisation. La deuxième classe interface est IResponsabilités_Autorités, qui permet aux responsables cellules et aux responsables d'organisation d'attribuer des responsabilités et des autorités pour les cellules d'organisation sur des entités (objets d'entreprise, vues d'objet, processus, activités, etc.)

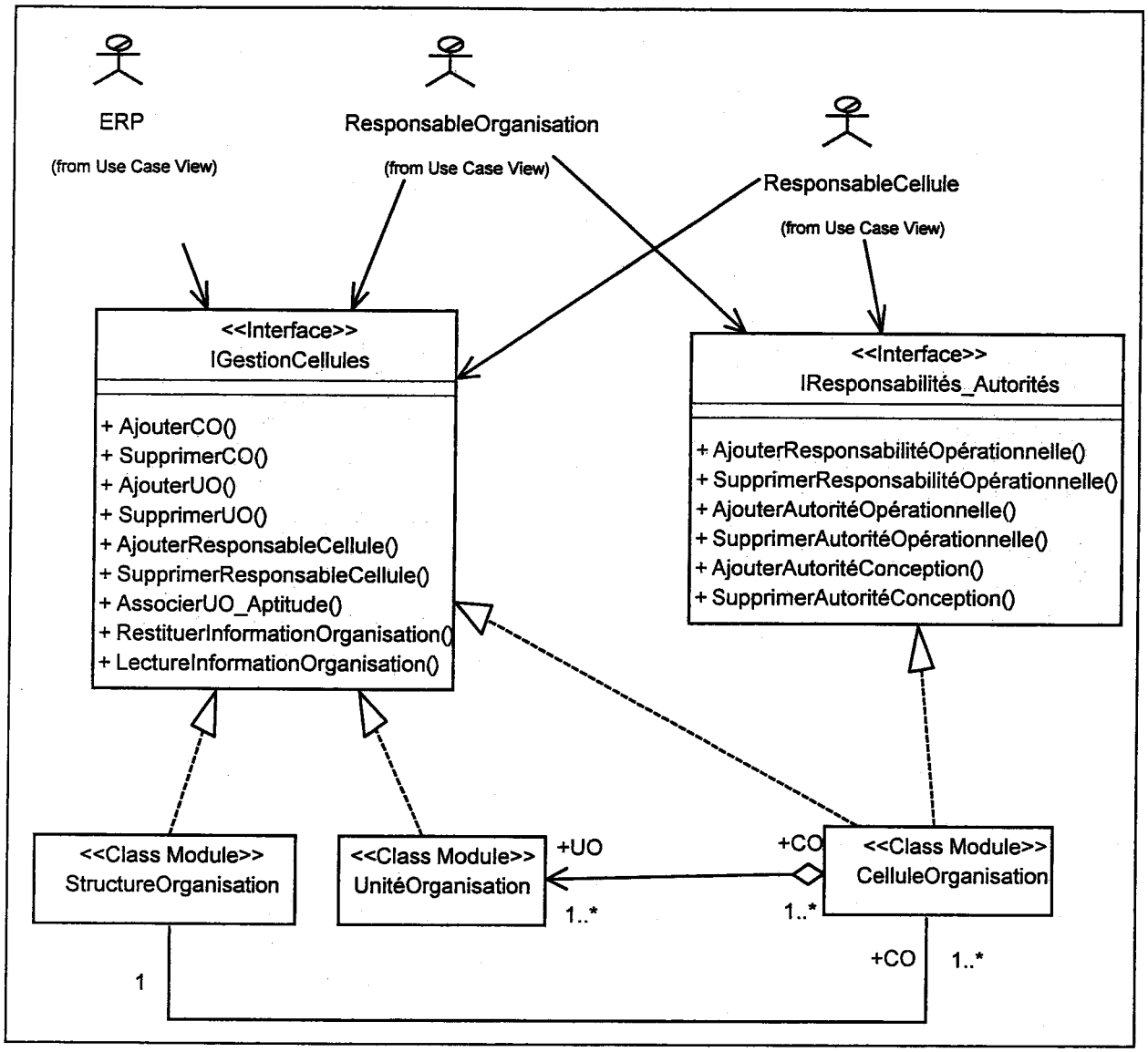

Figure IV.4-29 – Interface du composant Structure d'Organisation

Les deux diagrammes de séquences de la Figure IV.4-30 et la Figure IV.4-32 montrent les différentes opérations possibles lors de l'utilisation des deux classes interfaces définies ci-dessus.

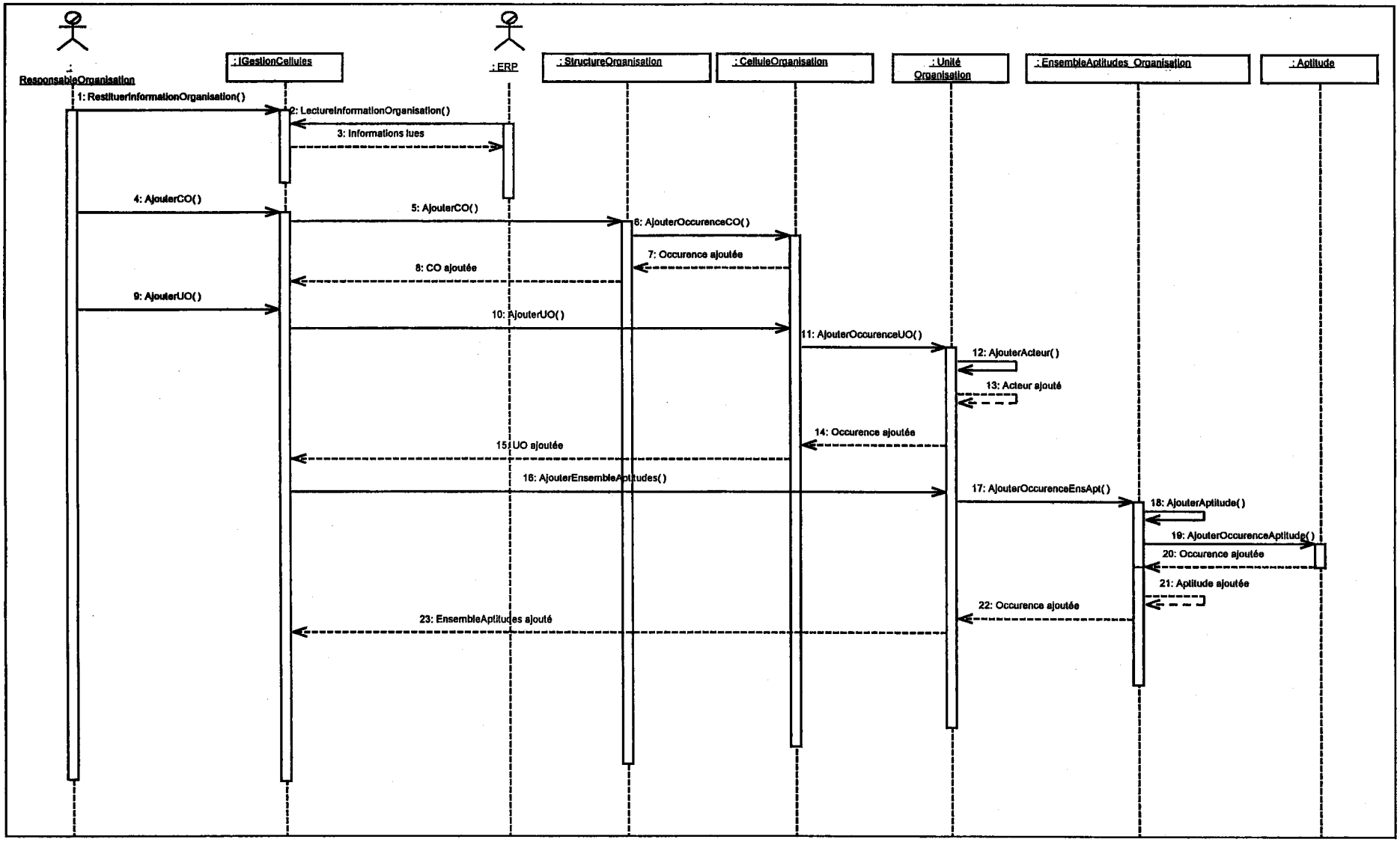

Figure IV.4-30 – Utilisation de l'interface IGestionCellules

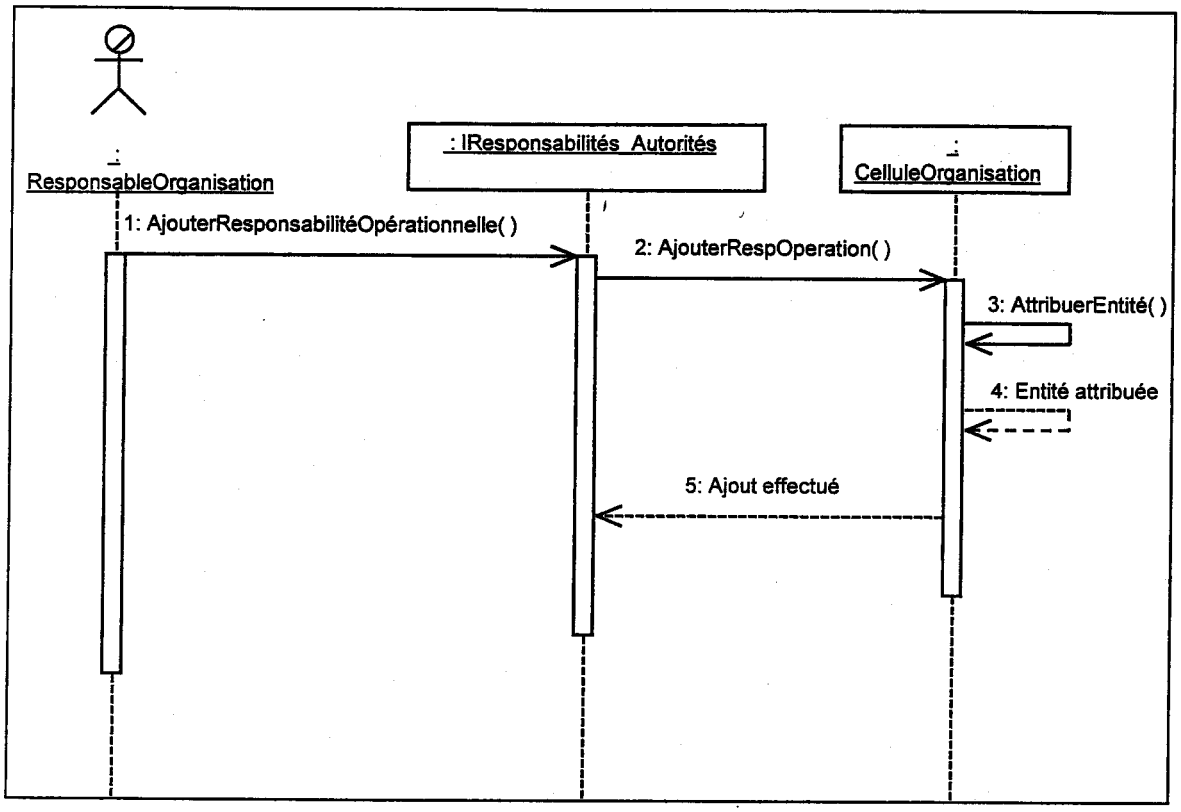

Figure IV.4-31 – Utilisation de l'interface IResponsabilités_Autorités

Les opérations d'attribution d'autorité opérationnelle et de conception aux cellules d'organisation et aux unités d'organisation se réalisent de la même façon que l'opération d'attribution de responsabilité pour les cellules et unités d'organisation figurant dans la Figure IV.4-31.

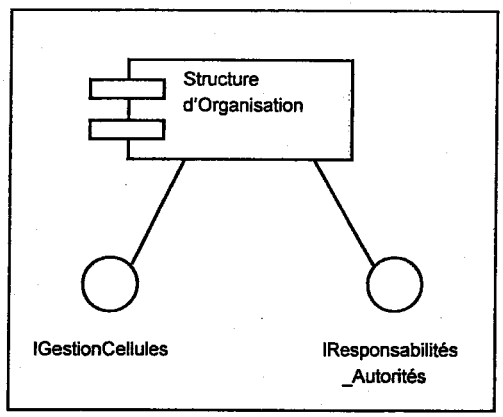

Figure IV.4-32 – Composant « Structure d'Organisation »

IV.4.9. COMPOSANT COMPORTEMENT

a) Objectifs

Ce cinquième composant permet de définir le comportement des processus définis par le composant Gestion & Conception de Processus. En effet, il met à disposition de modélisateur un éditeur de règles de comportements. Ces règles peuvent être modifiées ou supprimées selon les objectifs et les contraintes reliées au processus.

b) Principe

Le flux de contrôle des processus est décrit au moyen d'un langage formel permettant la description d'une séquence de règles de comportement (§ *Annexe A*) ayant la syntaxe [CIMOSA, 94] suivante :

WHEN (condition) DO action

La partie condition sert soit à déclencher un processus en utilisant la commande START et/ou des événements, soit à décrire les états des étapes du processus. Dans ce cas, la fonction standard ES(F) donne l'état de terminaison de l'étape F du processus.

Les règles de comportement définissent les décisions logiques requises pour déterminer les séquences des processus et des activités dans le but d'exécuter les tâches générales d'un processus métier, d'un processus maître structuré ou d'un domaine.

Exemples de déclenchement :

Déclencher un processus maître :

- WHEN (START WITH Evénement_a) DO Efy
- WHEN (START WITH Evénement_b AND Evénement_b) DO Efz
- WHEN (START WITH Evénement_b OR Evénement_c) DO Efz

Déclencher un processus métier:

- WHEN (START) DO Efx
- WHEN (ES(Efx)=valeur_1) DO Efy
- WHEN (ES(Efy)=valeur_2 AND Evénement_1) DO EFz

Efx, Efy et Efz sont des processus métier ou des activités.

START est une condition initiale, utilisée pour déclencher un processus.

c) Structure du composant

En partant de la syntaxe d'une règle de comportement, il est important de définir les concepts de « Condition », « Action » et « Opérateur » (voir Figure IV.4-33).

Avec le concept « Condition », il est possible d'identifier les événements et les états de processus (processus maître ou processus métier). Par conséquent, il est important de se référer aux événements qui sont susceptibles à déclencher le processus étudié et définir les états de ses processus/activités.

D'autre part, le concept « Action » permet de définir les processus ou les activités à déclencher. Donc, une référence aux processus/activités appartenant au processus étudié est obligatoire.

Il est aussi nécessaire de définir les opérateurs logiques au niveau condition et/ou action. Ceci permet de définir les synchronisations ou les conditions possibles d'exécution d'un processus ou d'une activité.

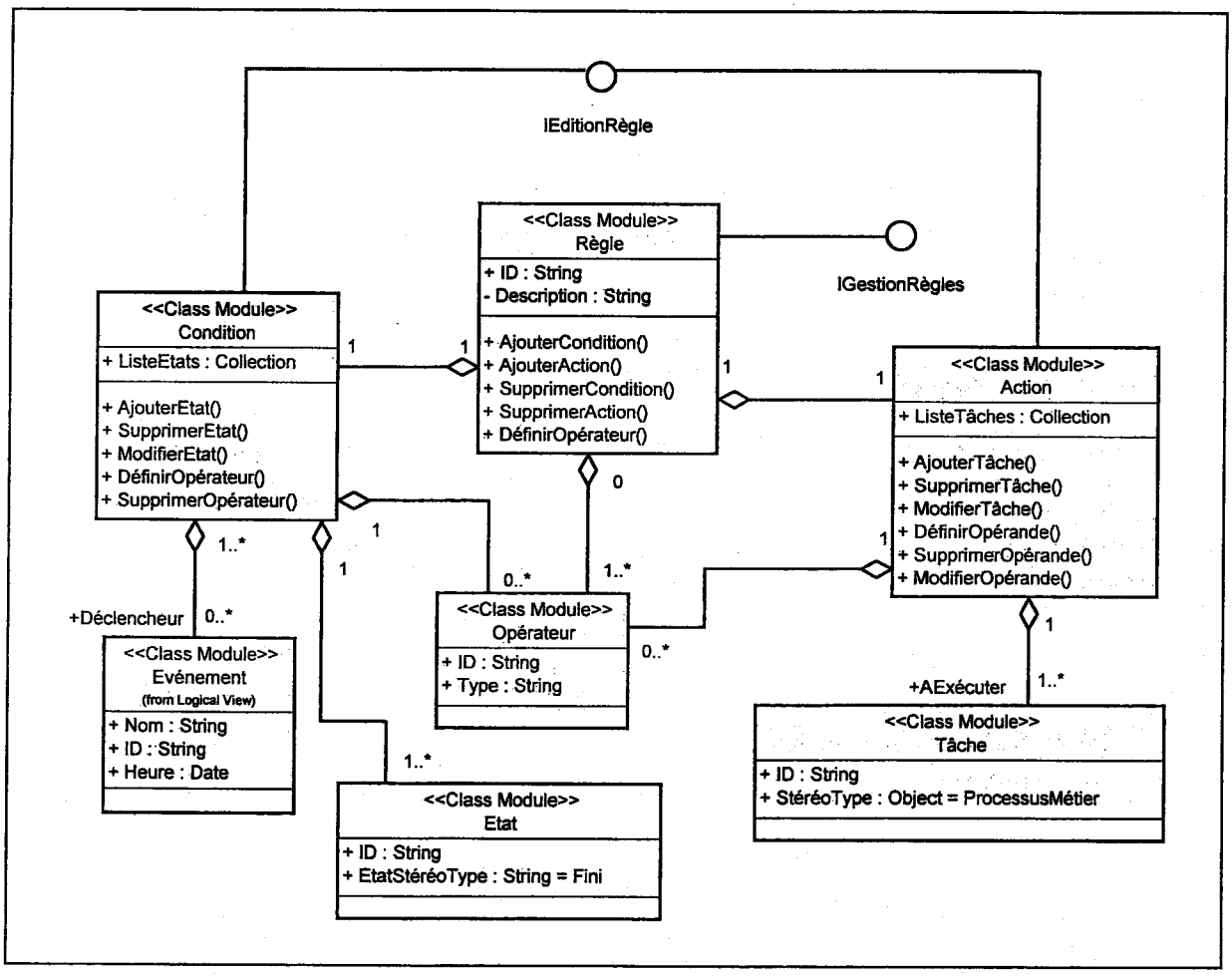

Figure IV.4-33 – Structure du composant

d) Interface du composant

Les interfaces du composant « Comportement » permettent l'édition et la gestion des règles de comportement.

En effet, nous avons défini la classe interface IEditionRègle qui se réfère aux classes « Condition » et « Action ». Cette classe interface permet au modélisateur de déterminer, pour chaque processus, les événements et les états de sous-processus en amont des sous-processus/activités à déclencher.

Au niveau « Condition », le modélisateur déterminer les sous-processus/activités à déclencher selon les conditions définies dans « Condition ». L'ensemble de ces sous-processus/activités peut être structuré ou non-structuré et synchrone ou asynchrone.

Exemple :

Soit le processus25 qui peut être déclenché par les événements « Evenement12 » et « Evenement115 ».

Ce processus contient les sous-processus « processus115 », « processus13 » et « processus111 » et, les activités « Activité5 » et « Activité201 ».

Les événements « Evenement115 » et « Evenement12 » déclenchent respectivement les sous-processus « Processus13 » et « Processus115 ».

Quand ces deux derniers processus s'achèvent en même temps (synchronisation), le sous-processus « Processus111 » se déclenche.

Ce dernier peut déclencher des activités selon son état comme suit :

- *État = fini : déclencher l'activité « Activité5 »,*
- *État = suspendu : déclencher l'activité « Activité201 »,*
- *État = abandon : achever le processus parent « Processus25 ».*

Dans le cas où l'activité « Activité201 » est lancée, et après son achèvement, le sous-processus « processus111 » est relancé.

Quand l'activité « Activité5 » est achevée, le processus parent « Processus25 » est achevé.

Ce comportement du processus « Processus25 » peut être décrit par les règles de comportements suivants et représentés par le diagramme d'activités d'UML dans la Figure IV.4-34.

« Lancement du processus « Processus25 » »

WHEN (START WITH Evenement115 AND Evenement12) DO (Processus25)

« Exécution du processus « Processus25 » »

WHEN (START WITH Evenement115) DO (Processus13)

WHEN (START WITH Evenement12) DO (Processus115)

WHEN (Processus13=fini AND Processus115=fini) DO (Processus111)

WHEN (Processus111=suspendu) DO (Activité201)

WHEN (Processus111=fini) DO (Activité5)

WHEN (Processus111=abandon) DO FINISH

WHEN (Activité5=fini) DO FINISH

« Fin d'exécution du processus « Processus25 » »

« Fin Processus25 »

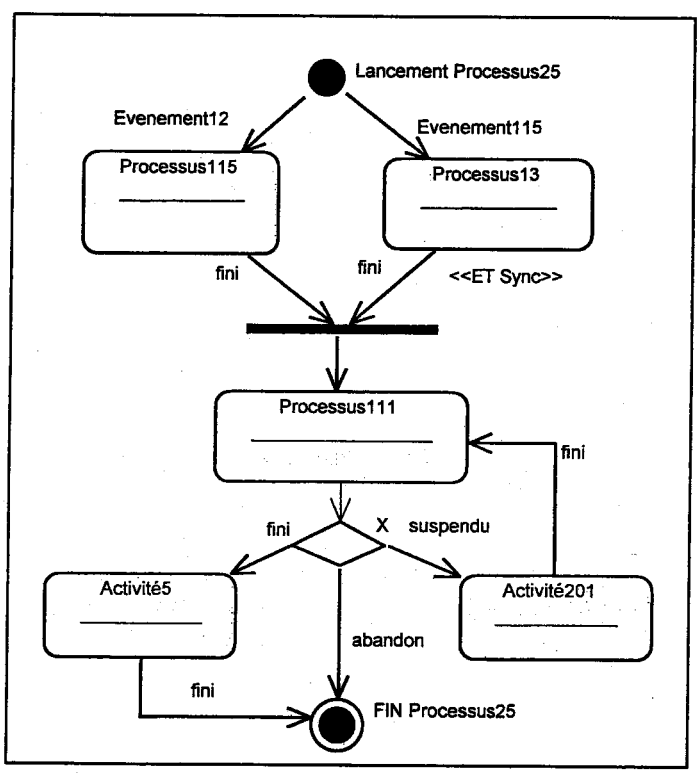

Figure IV.4-34 – Représentation de règles de comportement d'un processus

L'interface IGestionRègle permet au modélisateur de modifier, supprimer et de permuter les règles d'un processus lors d'une mise à jour de ce dernier.

Nous considérons ce composant en tant qu'extension du composant « Gestion & Conception de Processus », du fait qu'il aide à éditer et gérer les règles de comportement d'un processus défini. Cette extension est fonctionnelle lors de la conception d'un processus pour permettre la description de son exécution (Figure IV.4-35).

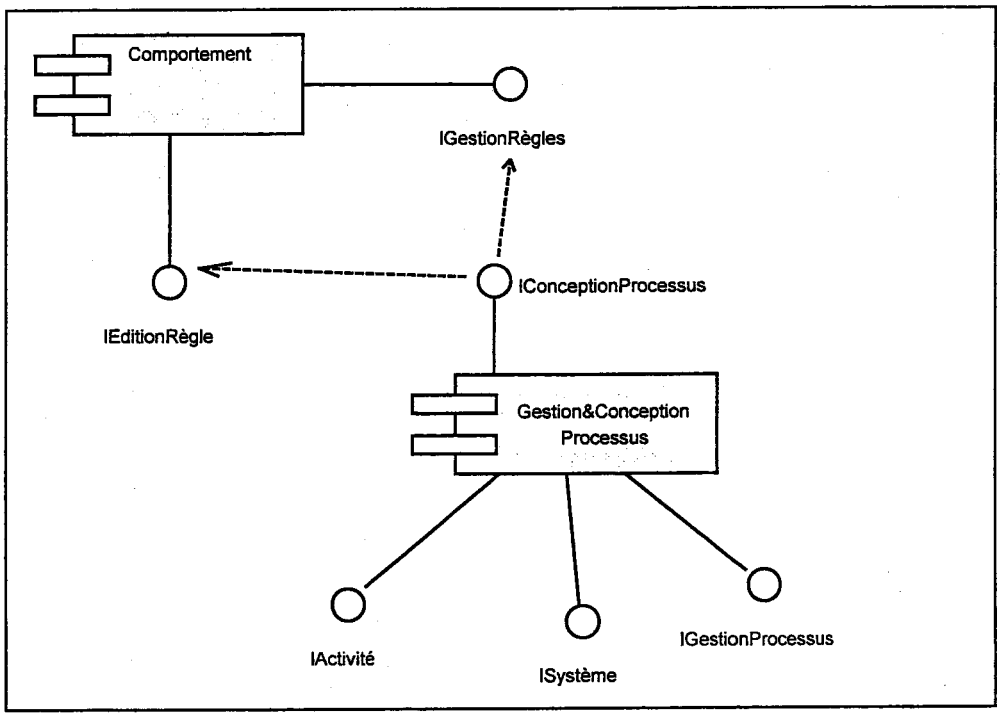

Figure IV.4-35- Collaboration du composant « Gestion & Conception de Processus » avec le composant « Comportement »

IV.4.10. PROPOSITION DE DÉPLOIEMENT DES COMPOSANTS DU RÉFÉRENTIEL

Les composants définis ci-dessus sont implémentés en tant qu'applications partagées pour permettre aux acteurs métier de développer des modèles systèmes.

Dans la Figure IV.4-36, nous proposons les bibliothèques ou *« repositories »* qui correspondent aux référentiels de la Figure IV.1-1.

Les composants de la bibliothèque EMC (*Enterprise Modeling Components*) sont partagés par différents acteurs métier et sont réutilisables pour le développement de modèles de systèmes que nous appelons ApplicationCIMOSA. Par conséquent, ils peuvent jouer le rôle de serveurs.

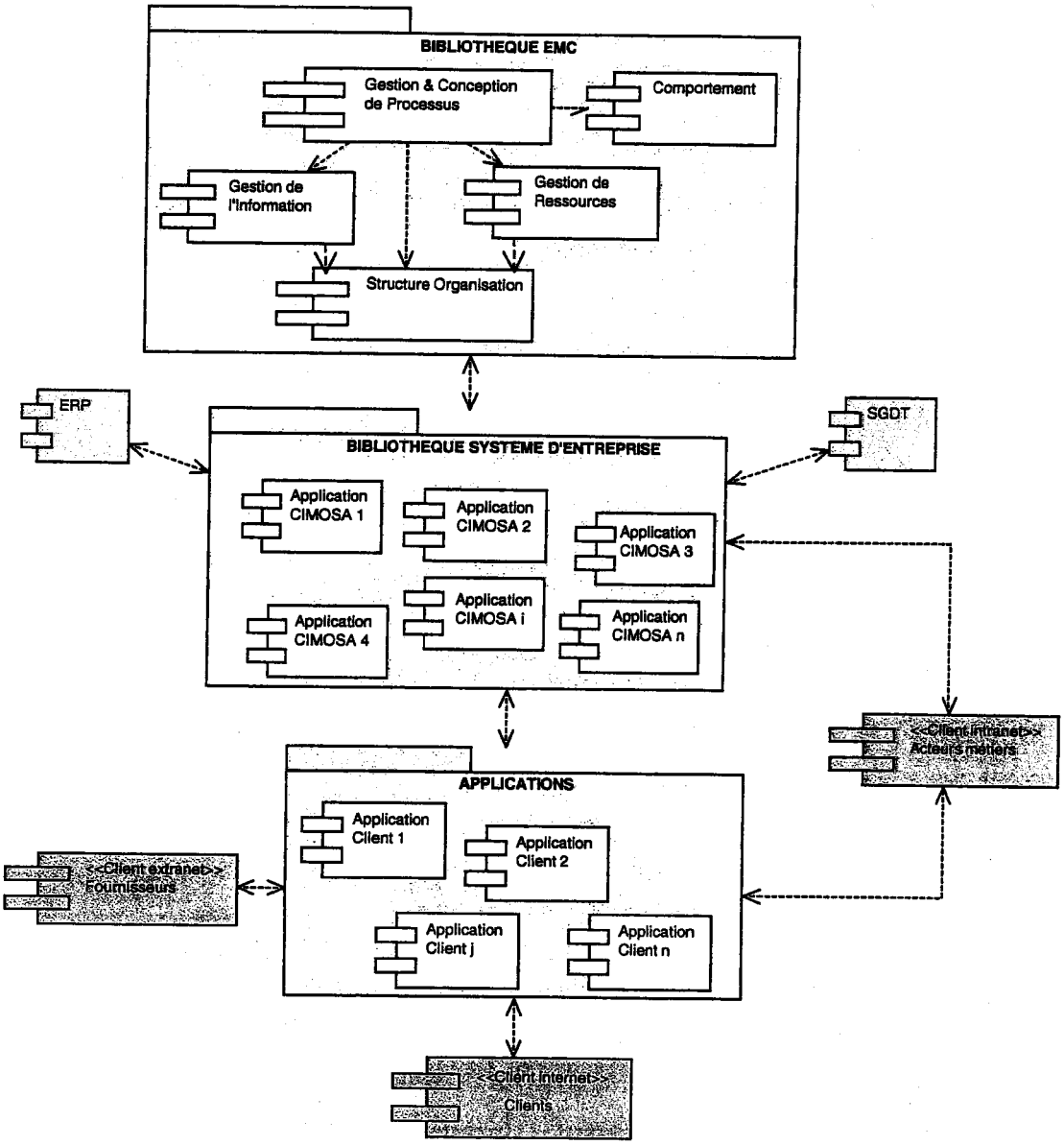

Figure IV.4-36 – Architecture proposée pour le déploiement du référentiel EMC

Les ApplicationsCIMOSA (modèles particuliers de l'entreprise) sont partagés, au sein d'une bibliothèque système dans l'entreprise, entre les acteurs métier (Figure IV.4-36).

L'utilisation de tels modèles permet le développement d'applications clients (résultats exploitables dans l'entreprise). Ces dernières sont exploitées par différents utilisateurs internes/externes ; tels que les acteurs métier, les clients, les fournisseurs et les sous-traitants ; via des clients Internet, intranet ou Extranet (Figure IV.4-36).

Les acteurs métier utilisent des bases de données, des systèmes intégrés (ex : ERP et/ou SGDT) pour récupérer des informations sur les moyens et les entités de l'entreprise. Ces informations sont utilisées dans les applications client développées.

IV.4.11. CONCLUSION

Les composants proposés correspondent à l'implémentation des quatre aspects décrivant le méta-modèle.

En effet, chaque composant représente un aspect du méta-modèle et permet aux acteurs métier de modéliser des systèmes via leurs interfaces.

Le composant S.I.R.S. est un composant qui permet de décrire le comportement d'un processus maître, les processus métier et le fonctionnement d'une activité en fonction, respectivement de ses processus métier, ses sous-processus métier/activités et des objets de ses entrées.

Chaque composant est spécifié par des fonctions par lesquelles il communique avec son environnement (bases de données, ERP, SGDT, composants, frameworks, etc.) et par des attributs. Ces fonctions et attributs sont mis en œuvre par les interfaces de chaque composant.

Ces dernières permettent aux composants de communiquer avec leur environnement. D'autre part, l'implémentation de chaque composant est encapsulée dans celui-ci par des classes objets et ne peut être consultée à partir de l'environnement, qu'à travers leurs interfaces.

Par conséquent, ceci permet de démontrer que les interfaces de chacun des composants développés sont séparées des de l'implémentation de chacun d'eux.

À part la réutilisation du savoir-faire et l'indépendance de chaque composant, la portabilité peut être assurée lors du développement d'un composant en un format standard qui lui permet d'être facilement utilisé par des clients Internet, intranet et/ou Extranet.

L'implantation des composants au sein d'une architecture distribuée de type Client/Serveur ou de type Web par les Web Services, permet l'interopérabilité entre eux ou avec des frameworks. Ceci est confirmé par Sprott dans son article à propos de l'implémentation des applications de l'entreprise par des composants [Sprott, 00].

CHAPITRE V

PROPOSITION D'UN FRAMEWORK POUR LA DISTRIBUTION DES PROCESSUS D'ENTREPRISE

CHAPITRE V : PROPOSITION D'UN FRAMEWORK POUR LA DISTRIBUTION DES PROCESSUS D'ENTREPRISE

V.1. INTRODUCTION

Dans ce chapitre, nous présentons un framework permettant la gestion de cartographies de systèmes d'entreprise qui consiste à créer, supprimer et modifier des cartographies de processus tout en spécifiant chaque processus.

Le framework est avant tout utilisé par l'administrateur CIMOSA. Il permet de développer le modèle particulier d'un système d'entreprise (S.E.). Le résultat est présenté sous la forme d'un document XML, et ainsi stocké dans une bibliothèque systèmes de l'entreprise (cf. Figure V.3-1). Ces documents XML peuvent être exploités à partir d'applications clients. C'est ainsi que chaque application est adaptée à un contexte d'utilisation.

Dans ce cadre, nous pouvons citer les travaux de Nikunj [Nikunj, 04] qui propose la réalisation d'un framework pour la modélisation des processus de l'entreprise, l'objectif étant d'enrichir les systèmes ERP de nouvelles générations. Dans ce travail, un nouveau langage graphique de modélisation de processus –Enterprise Process Modeling Language (EPML) [Chaugule, 01] – a été développé. Ce langage utilise des techniques qui ressemblent à ceux de DFD, IDEF et EPC de ARIS qui a été intégré dans SAP R/3.

V.2. NOTION DE FRAMEWORK

V.2.1. DEFINITION

Généralement, le terme framework est utilisé pour décrire les résultats de conception qui sont réutilisables dans n'importe quelle situation isomorphique[15].

Un framework décrit aussi une situation typique réutilisable au niveau du modèle, alors que le framework de composants constitue un tableau avec des places vides attendant d'être remplies par des composants. Quand cela est réalisé, le framework est dit instancié.

Parmi l'ensemble des définitions de la littérature, nous présentons trois :

- Un framework peut être défini comme un patron architectural [Gamma, 95] qui fournit un modèle extensible pour les applications, dans un domaine spécifique [Booch, 00].

[15] Situation dans un même domaine ou contexte (même besoins et objectifs).

- Un framework peut être défini comme une conception réutilisable ou un squelette d'une application qui peut être personnalisée par un développeur [Johnson, 97].
- Un framework est une micro-architecture qui fournit une définition incomplète des systèmes dans un domaine spécifique [Jacobson, 97].

Comme représenté par la Figure V.2-1, un framework peut être considéré comme un patron architectural, ayant des points d'extension. Ces points permettent d'augmenter et d'étendre la définition du comportement du framework. En outre, un framework peut être implémenté par plusieurs composants.

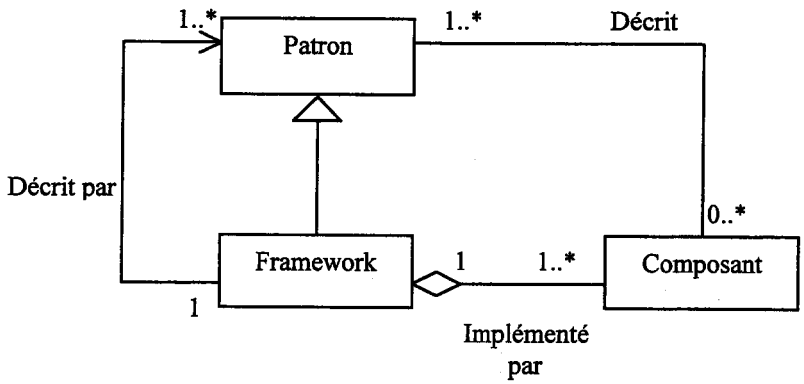

Figure V.2-1 – Méta-modèle d'un Framework (d'après [Larsen, 99])

V.2.2. TYPOLOGIE

Tkach & Puttick [Tkach, 96] distingue trois types de frameworks :

- Les Frameworks techniques, qui sont similaires aux anciennes applications.
- Les Frameworks industriels, qui sont focalisés sur le côté modèle métier.
- Les Frameworks application, qui sont focalisés sur les problèmes métier dans les domaines spécifiques.

Dans notre cas, nous nous plaçons dans le cadre d'un framework de type industriel du fait qu'il se focalise sur le méta-modèle proposé à la section IV.3. Mais, rappelons que le méta-modèle est implémenté à travers les cinq composants proposés à la section IV.4.

V.2.3. IMPLEMENTATION D'UN FRAMEWORK

Il y a plusieurs tâches à réaliser pour concevoir et implémenter un framework. Nous citons les plus essentielles [Johnson, 97] :

1. Identifier le domaine spécifique pour lequel le framework sera appliqué,
2. Déterminer le cas d'utilisation définissant le fonctionnement de base du framework,
3. Identifier l'ensemble des acteurs connus qui interagiront avec le framework,
4. Déterminer les patrons existants ou d'autres solutions validées pour aider au développement du framework,
5. Concevoir les interfaces principales et les composants du framework ; cartographier les rôles et les acteurs dans les interfaces,
6. Fournir une implémentation par défaut des interfaces dans le framework,
7. Décrire et documenter les points d'extension du framework, et
8. Créer les tests et les plans du framework.

Pour ce qui nous concerne, nous nous sommes limités aux six premières étapes.

La Figure V.2-2 peut résumer la démarche suivie pour développer notre framework de composants.

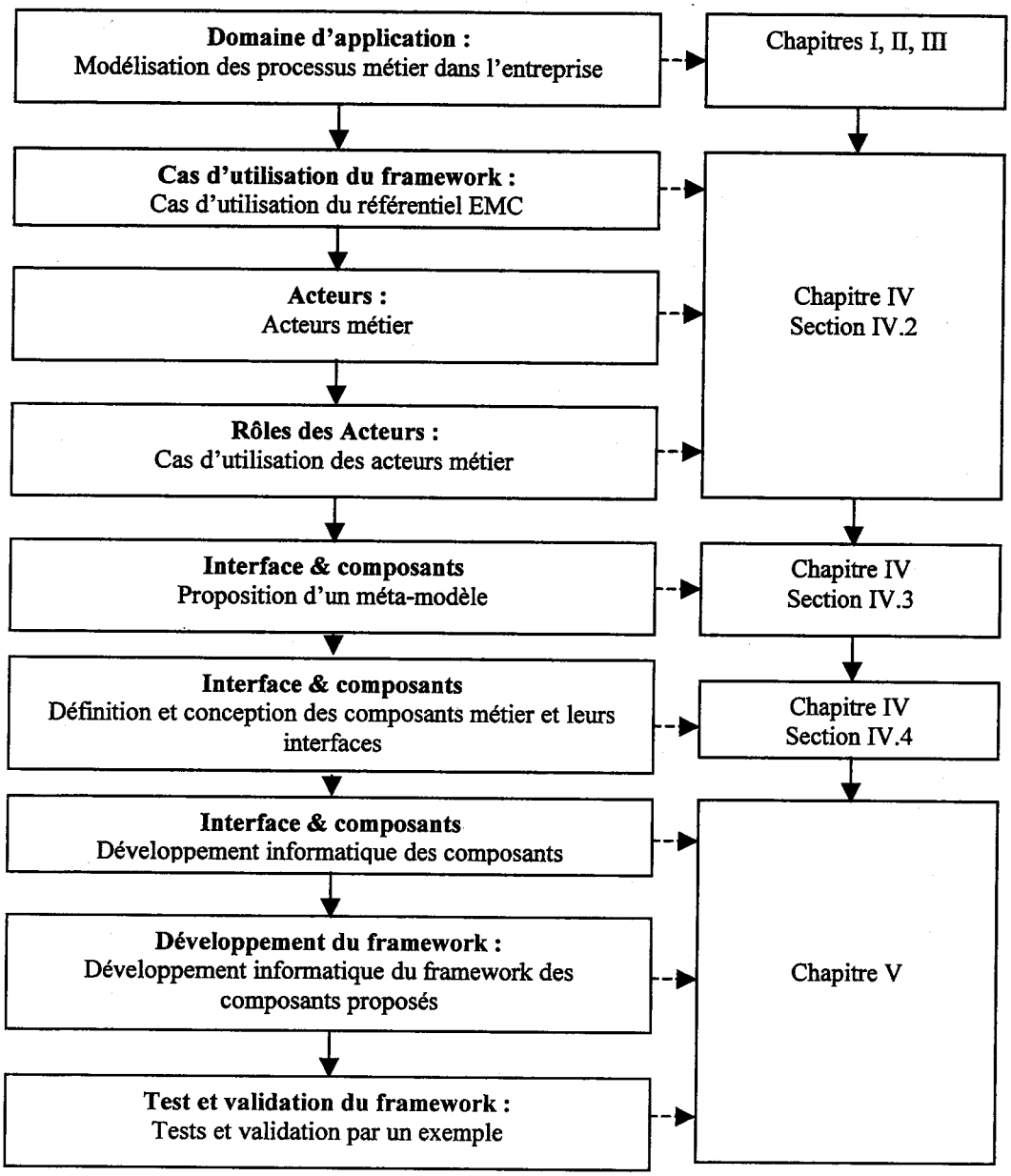

Figure V.2-2 – Démarche suivie pour la réalisation du framework

V.3. EXPLOITATION DES RESULTATS

Il est important de choisir le format des résultats donnés par ces composant via le framework. Le format de fichier résultant du framework doit permettre l'interopérabilité avec d'autres composants, applications et logiciels existants dans l'entreprise. Cette problématique d'intégration a été traitée avant par le concept CIM[16] et ensuite par les outils de EAI[17]

[16] Computer Integrating Manufacture
[17] Enterprise Application Integration

[Manouvrier, 01] [Linthicum, 00] [Lutz, 00] [Morgenthal, 00]. La dernière approche utilise le format standard XML[18] dans ses différentes formes [Boukhors, 02] et permet de :

- Intégrer les données dans un datawarehouse au format XML,
- Utiliser des serveurs applications qui intègrent des documents XML permettant le dialogue entre plusieurs application écrites dans des langages différents,
- Intégrer l'interface utilisateur en passant par des navigateurs Web. Ces interface transmettent les données d'application en format XML en les présentant par une feuille XSL[19] stockée sur le client.

Donc, le format XML permet une meilleure intégration du framework dans l'entreprise et une interopérabilité avec les autres applications existantes. A noter qu'il est aussi utilisé dans l'entreprise pour les mêmes objectifs d'échange B2B[20] [JDNet, 03] et nous ouvre la possibilité d'étendre nos travaux au futur vers les services Web [Fremantle, 02] [Workshop, 04].

En résumé, nous avons choisi le format XML pour développer des documents qui stockent la structure fonctionnelle de chaque système étudié et des informations (ressources, entrées) nécessaires à chaque élément de la structure. D'autre part, ces documents XML que nous appelons « ApplicationSysteme » peuvent être utilisés par d'autres acteurs métier pour développer des applications et/ou des logiciels qui intègrent automatiquement la structure d'un système (Figure V.3-1).

[18] eXtensible Markup Language
[19] eXtensible Style sheet Language
[20] Business to Business

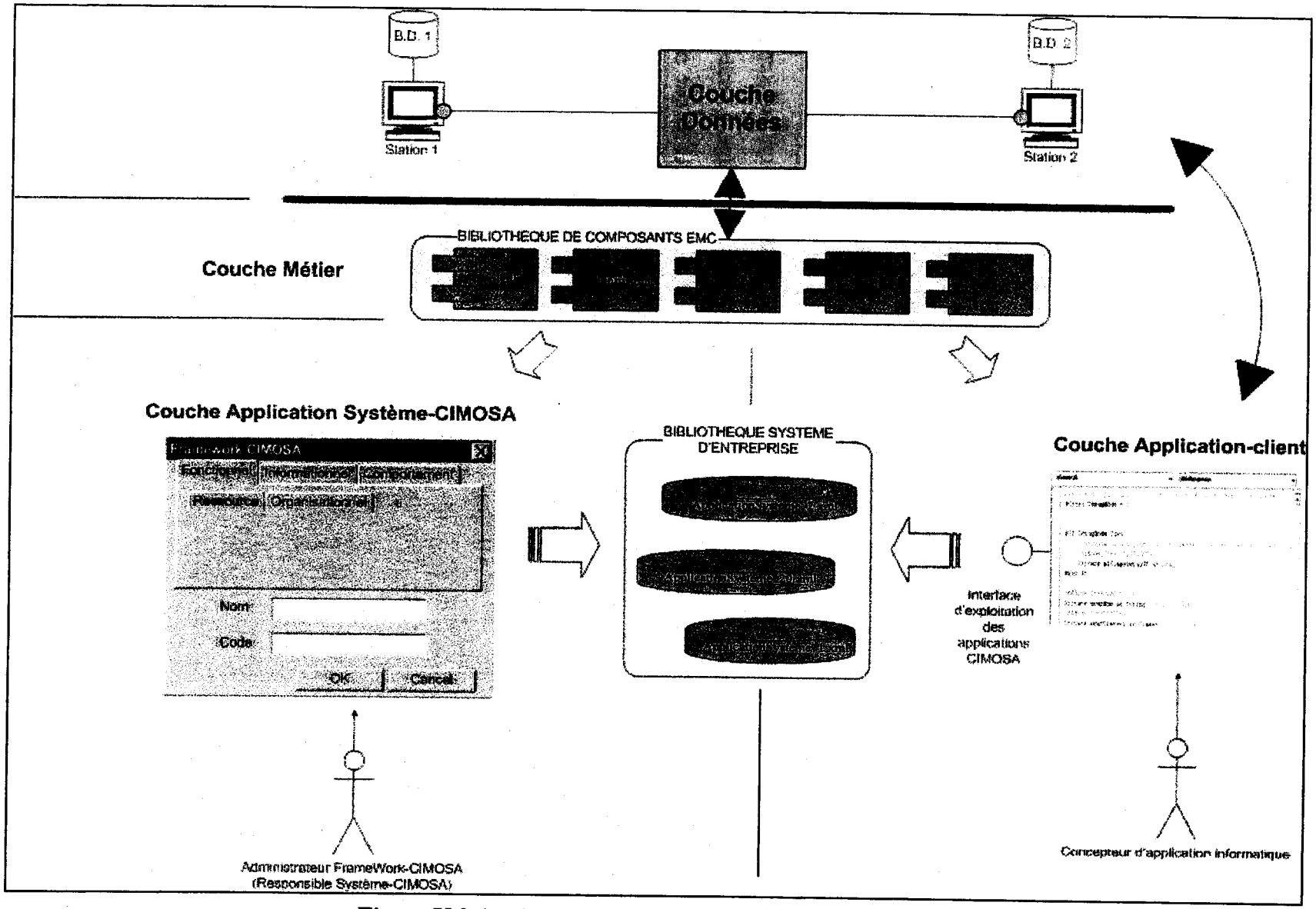

Figure V.3-1 – Architecture N-Tier pour le framework proposé

Lors du développement informatique des composants, nous avons utilisé une passerelle *« parser »* API[21] pour nous permettre de communiquer via un document XML.

Plusieurs API pour XML sont actuellement disponibles. Nous ne citons que les deux API les plus connues et les plus utilisées et qui sont DOM[22] et SAX[23]. Ces deux API correspondent à deux modèles de traitement de documents XML radicalement différents [Boukhores, 02].

- DOM est une API objet standardisé par le W3C et qui est une norme depuis 1998. Et le parser DOM compile le document XML en mémoire sous forme d'un arbre d'objets. Ce mécanisme autorise des traitements en profondeur du document XML même s'il a un coût en mémoire important (Figure V.3-2).

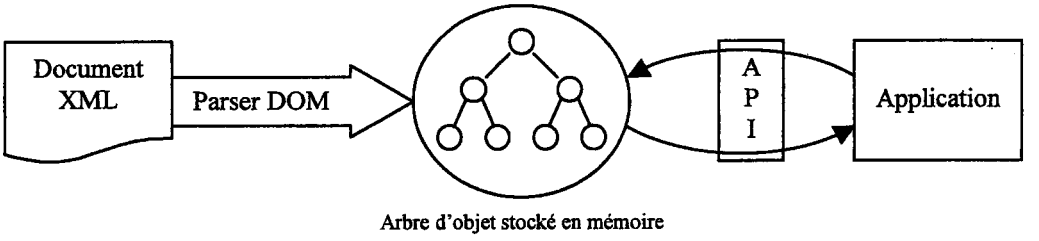

Figure V.3-2 – Le parser DOM et l'API DOM pour XML

- SAX est une API événementielle non normalisée par le W3C et a été développée sur le modèle des logiciels libres (open source software). Cette API ne charge pas l'arbre représentant le document XML, mais permet la communication par événements pour effectuer des traitements (Figure V.3-3).

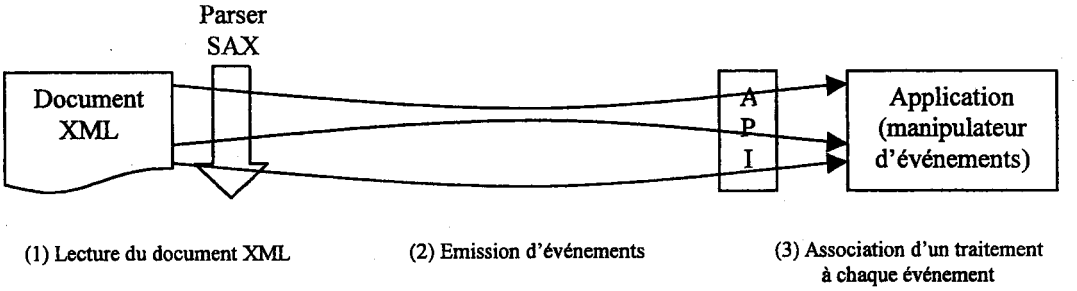

Figure V.3-3 – Le parser SAX et l'API SAX pour XML

[21] Application Programming Interface
[22] Document Object Model
[23] Simple API for XML

Notre choix est parti sur l'API et le parser DOM. Ce qui permet de charger et de traiter les objets du document XML qui correspondent aux différents éléments de la structure d'un système et des informations qui doivent être utilisées. Ce type de parser et d'API permettent aussi au client (dans notre cas un client peut être le framework CIMOSA ou une application client (Figure V.3-4) de se connecter au document XML qu'une seule fois lors de chaque utilisation du fait qu'ils chargent en mémoire l'arbre représentant ce document. Ceci est intéressant lorsque le document XML existe sur un serveur, alors le client ne sera pas obligé d'être toujours connecté au serveur, ce qui n'est pas le cas du parser SAX.

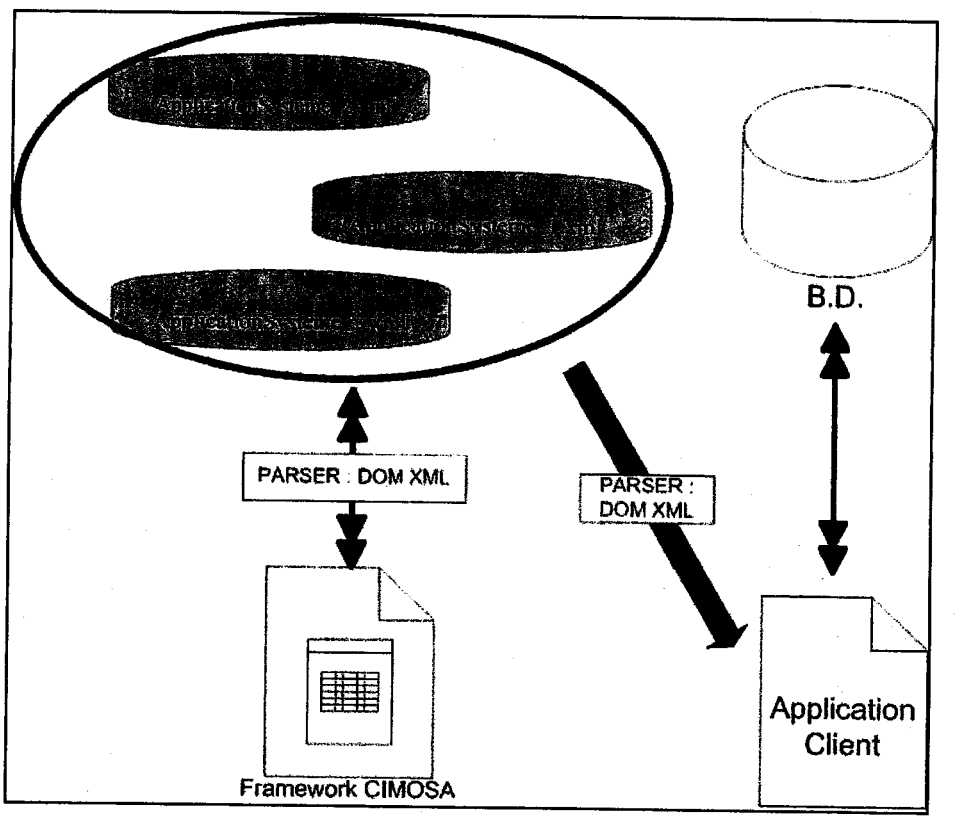

Figure V.3-4 – Représentation de communication avec les documents du référentiel du système d'entreprise par le parser DOM XML

Dans la section suivante, nous présentons un exemple académique de l'utilisation du framework.

V.4. UTILISATION DU FRAMEWORK CIMOSA : EXEMPLE ACADEMIQUE

Le framework proposé permet la manipulation des composants de modélisation existants dans la bibliothèque des composants (EMC).

Pour le moment, l'interface graphique du framework est partiellement développée et correspond surtout au traitement des aspects fonctionnels d'un système. Par conséquent, dans l'exemple ci-dessous nous étudions la structure fonctionnelle d'un système d'entreprise.

Les domaines du système à étudier sont définis en partant des objectifs de celui-ci.

Nous appelons l'ensemble des domaines du système d'entreprise une cartographie de domaines. Dans cette cartographie, les relations entre les domaines sont définies. Chaque relation correspond à un événement (ordre de fabrication) ou un échange d'informations (exemple : documents électroniques ou papier) ou de matière (exemple : un produit).

V.4.1. EXEMPLE

Nous prenons un exemple académique qui traite un système d'une entreprise. Cet exemple a été tiré en partie d'un cas réel. Ce système doit permettre d'atteindre l'objectif stratégique « améliorer le coût de fabrication d'un produit A sur la ligne de production 7 ».

Pour cela, il est nécessaire d'avoir le modèle détaillé du processus de fabrication du produit A sur la ligne 7. Ce modèle doit décrire les différentes étapes du processus, les entrées/sorties et les moyens requises pour l'évolution de celui-ci. Par conséquent, nous partons du besoin de réalisation du produit A et nous décrivons son processus de fabrication en se basant sur la définition des étapes existantes. En final, nous obtenons le modèle de processus du système étudié.

C'est ainsi que le besoin « réalisation du produit A sur la ligne de production 7 » est identifié dans notre approche en tant qu'objectif-domaine du système (étape 1). Ensuite, nous identifions les processus métier déjà existants correspondant à chaque domaine (étape 2).

Après l'identification des processus métier, nous regroupons ces derniers dans des processus maîtres par le biais des objectifs maîtres, résultants de la décomposition de chacun des objectifs-domaine (étape 3).

Plusieurs solutions de regroupements par objectifs peuvent exister.

Dans ce cas, il est difficile de gérer ces différentes solutions pour pouvoir choisir une qui convient le mieux à l'objectif stratégique donné ci-dessus.

Une solution est retenue en utilisant l'application S.I.R.S.

Ensuite, nous définissons les entrées/sorties, les ressources et le flux de contrôle de chaque processus maître et métier en utilisant (étape 4).

Finalement, le modèle obtenu du système est traduit en un document XML par le framework et stocké dans la bibliothèque de systèmes d'entreprise.

a) Définition des domaines du système d'entreprise (Étape 1)

Nous pouvons décomposer l'objectif-domaine « réalisation du produit A sur la ligne de production 7 » comme suit :

- Concevoir le produit A,
- Fabriquer le produit A,
- Intervenir pour réparer les équipements utilisés pour la fabrication,
- Maintenir les équipements périodiquement (maintenance préventive et corrective),
- Fourniture de ressources nécessaires pour la fabrication et la maintenance.

Nous définissons les domaines qui peuvent correspondre à ces objectifs comme suit (Figure V.4-1) :

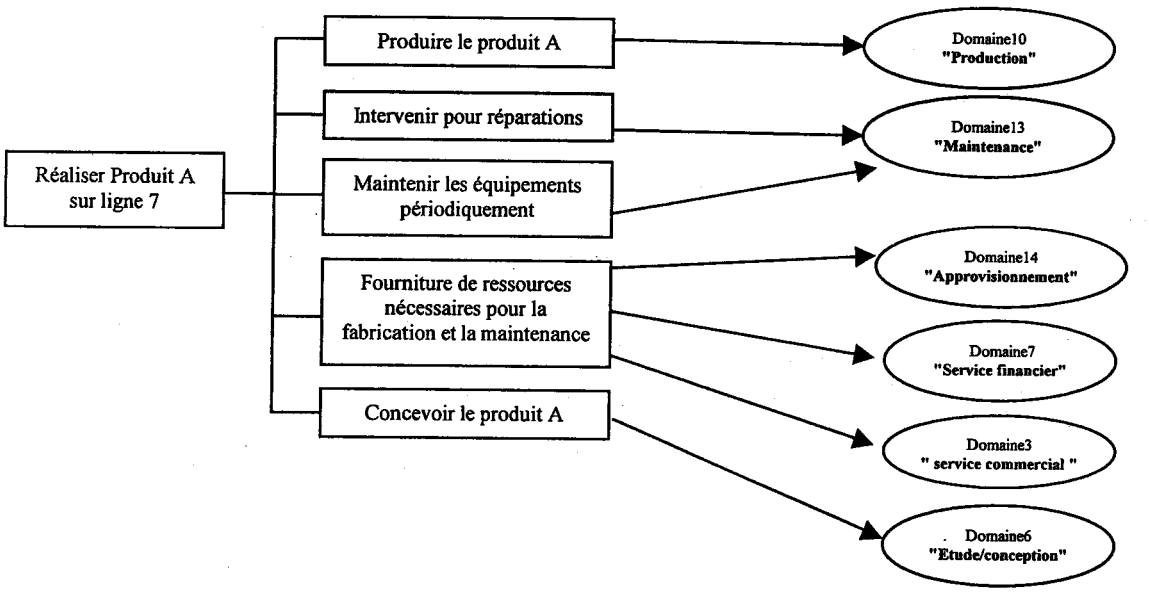

Figure V.4-1 – Définition des domaines du système

Ensuite, nous représentons la cartographie des domaines du système en définissant les relations entre les domaines.

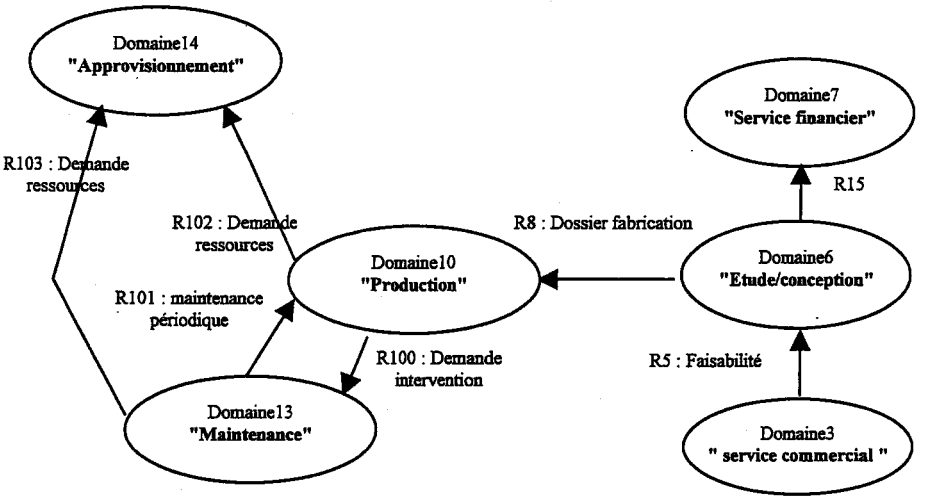

Figure V.4-2 – Cartographie des domaines d'entreprise

Après la définition des domaines du système d'entreprise, nous passons à l'étape 2 où nous identifions les processus métier de chaque domaine.

b) Identification des processus métier (Étape 2)

Prenons le cas du domaine 10 qui contient les processus de production.

Pour définir les processus de ce domaine, nous allons décomposer chacun de ses objectifs et nous définissons des processus pour atteindre chaque objectif décomposé.

En effet, l'objectif « Produire le produit A sur la ligne 7 » peut être décomposé aux sous-objectifs (nous les appelons objectifs métier) suivants :

- Préparer régler les équipements permettant la réalisation du produit avec une qualité optimale,
- Usiner les pièces sur les différentes machines définies dans la gamme de fabrication,
- Contrôler et valider les pièces usinées.

Le premier objectif métier peut être atteint en définissant des processus/activités de préparation (ex : placer outillage sur machine, régler et positionner l'outillage, régler les machines (PREF), etc.) d'outillage et de machines nécessaires pour la réalisation du produit.

Le deuxième objectif métier correspond à l'usinage des pièces sur les différentes machines nécessaires pour obtenir les produits.

Et finalement, contrôler les pièces usinées visuellement et mesurer les côtes fonctionnelles dans le but de le valider.

Lors d'une panne d'une machine (événement) par exemple, une demande d'intervention (vue d'objet) est envoyée à la maintenance (domaine).

Nous représentons graphiquement cette modélisation dans la Figure V.4-3:

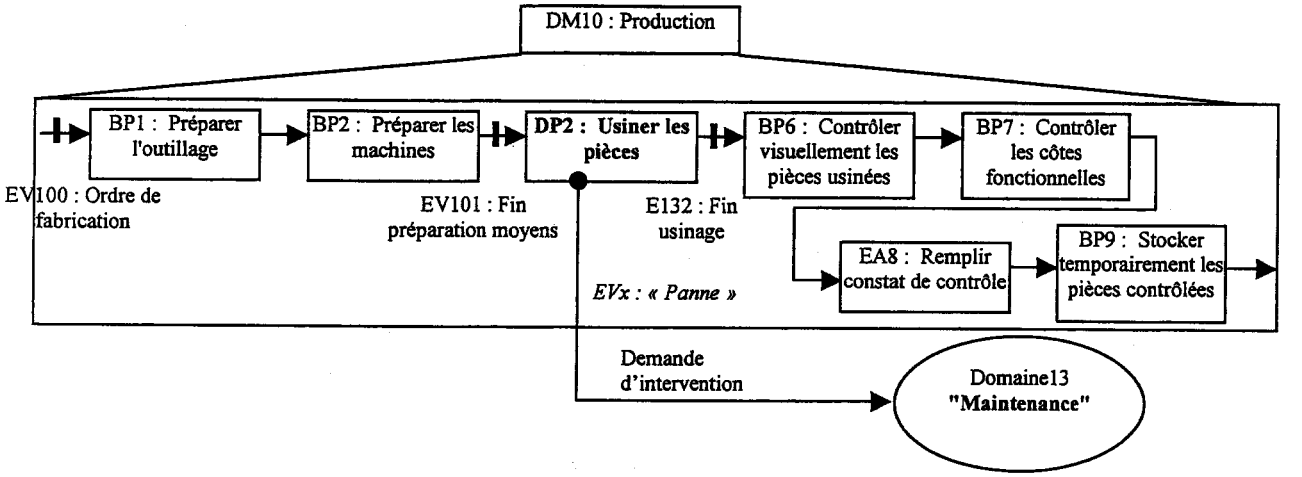

Figure V.4-3 – Identification des processus du domaine « Production »

c) Regroupement des processus métier en processus maîtres (Étape 3)

Après la définition des processus métier qui correspondent aux objectifs métier, nous allons regrouper ces processus dans des processus maîtres. Ces derniers doivent être déclenchés que par des événements et regrouper des processus métier factorisés par un objectif. Dans notre exemple, nous pouvons proposer les regroupements comme représentés dans la Figure V.4-4.

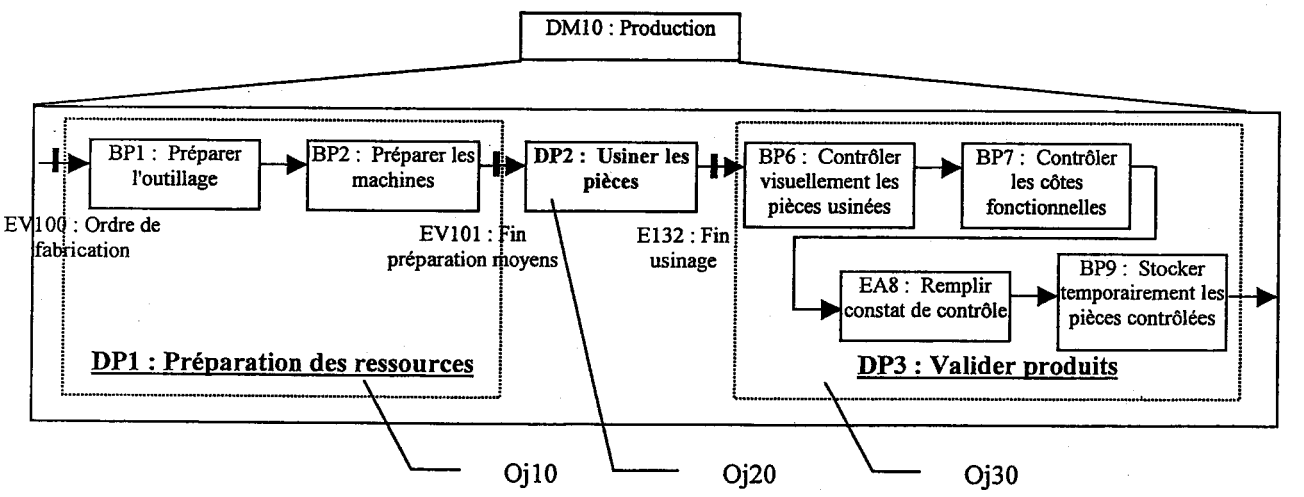

Figure V.4-4 – Proposition de regroupement des processus métier du domaine « Production »

Nous obtenons donc les processus suivants (solution 1):

Processus	Objectif	Déclencheurs
DP1 : Préparation de ressources	Oj10 : Préparation des moyens	EV100 : Ordre de fabrication
BP1 : Préparer l'outillage	Oj1 : Prépraration d'outillage	
BP2 : Préparer les machines	Oj2 : Régalge des machines	
DP2 : Usiner les pièces	Oj20 : Usinage de pièces	EV101 : Fin de préparation des moyens
BP3 Usiner les pièces sur les Tours	Oj3 : Dresser et charioter les pièces	
BP4 : Usiner les pièces sur les fraiseuses	Oj4 : surfacer les pièces	
BP5 : Rectifier les pièces	Oj5 : Améliorer l'état de surface des pièces	
DP3 : Valider produits	Oj30 : Contrôle et validation des produits	EV132 : Fin d'usinage
BP6 : Contrôler visuellement les pièces usinées	Oj6 : Vérifier s'il y a des défauts apparents sur les pièces	
BP7 : Contrôler les côtes fonctionnelles sur les pièces usinées	Oj7 : Mesurer les côtes fonctionnelles et vérifer si elles sont dans leurs intervalles de tolérences	
EA8 : Remplir constat de contrôle	Oj8 : Noter les remarques de contrôle	
BP9 : Stocker temporairement les pièces contrôlées	Oj9 : stocker les pièces validées et non validées séparément à la fin de la ligne de production 7	

Les sous-processus métier BP1, BP2, BP3, BP4 et BP5 peuvent être définis comme activités dans le cas où ils sont irréductibles (non décomposables).

Il est possible d'avoir plusieurs autres propositions de regroupement des processus métier.

Comme par exemple (solution 2) :

Processus	Objectif	Déclencheurs
DP1 : Fabrication du produit	Oj100 : Usiner les pièces composant le produit	EV100 : Ordre de fabrication
BP1 : Préparer l'outillage	Oj1 : Prépraration d'outillage	
BP2 : Préparer les machines	Oj2 : Régalge des machines	
BP3 Usiner les pièces sur les Tours	Oj3 : Dresser et charioter les pièces	
BP4 : Usiner les pièces sur les	Oj4 : surfacer les pièces	

fraiseuses		
BP5 : Rectifier les pièces	Oj5 : Améliorer l'état de surface des pièces	
DP3 : Valider produits	Oj30 : Contrôle et validation des produits	EV132 : Fin d'usinage
BP6 : Contrôler visuellement les pièces usinées	Oj6 : Vérifier s'il y a des défauts apparents sur les pièces	
BP7 : Contrôler les côtes fonctionnelles sur les pièces usinées	Oj7 : Mesurer les côtes fonctionnelles et vérifer si elles sont dans leurs intervalles de tolérences	
EA8 : Remplir constat de contrôle	Oj8 : Noter les remarques de contrôle	
BP9 : Stocker temporairement les pièces contrôlées	Oj9 : stocker les pièces validées et non validées séparément à la fin de la ligne de production 7	

Etc.

Ces solutions sont capitalisées par le biais de l'application S.I.R.S. (§ III.4) et peuvent être représentées par le graphe suivant (Figure V.4-5) :

Remarque : La différence entre les deux solutions est l'objectif Oj100 de la solution 2 à la place du Oj10 de la solution 1, au niveau de DP1.

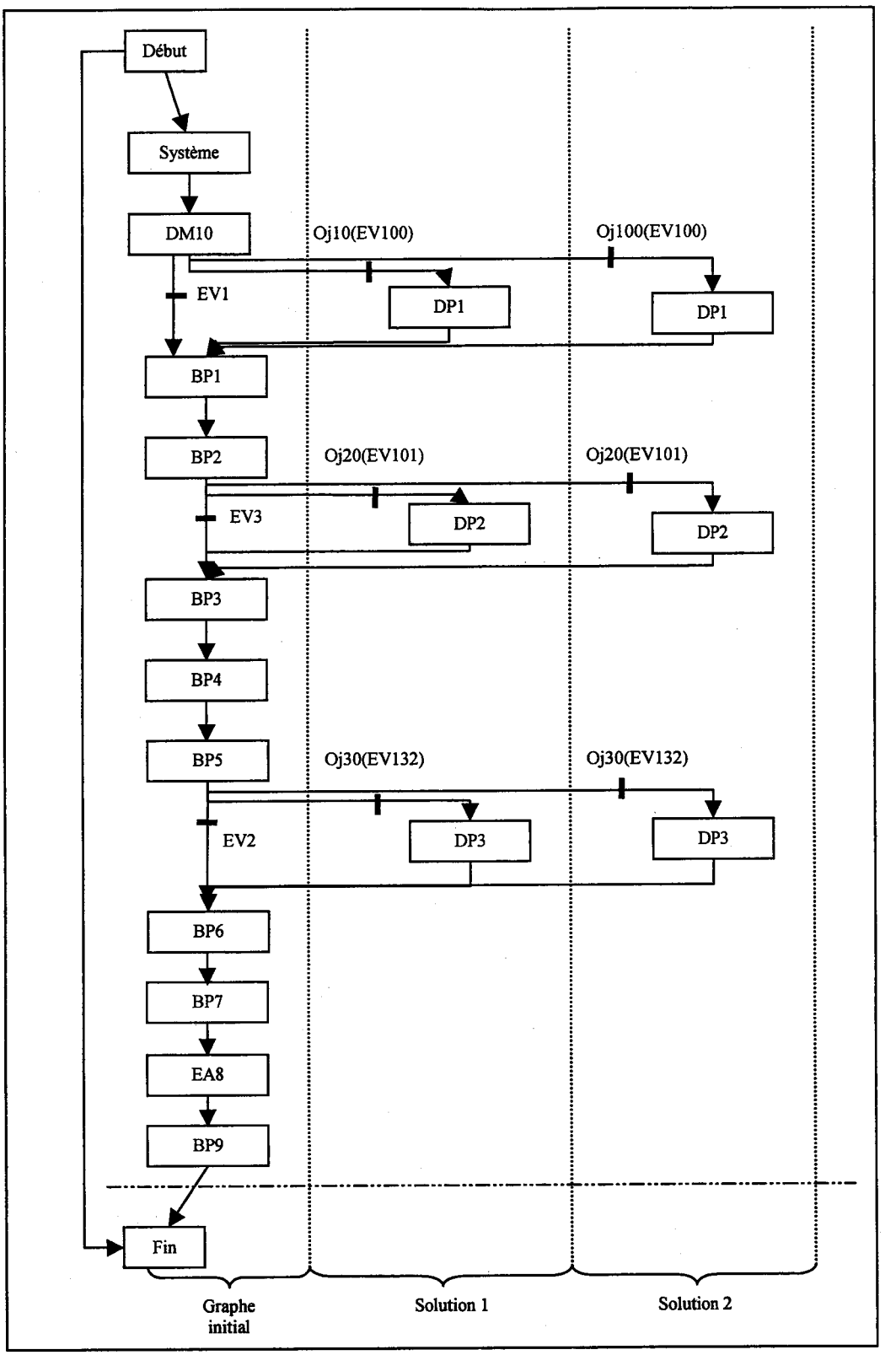

Figure V.4-5 – Famille de solutions de regroupement des processus

Ceci étant fait, l'application S.I.R.S génère les autres solutions possibles de regroupement et laisse le choix d'une solution au modélisateur.

d) Définition de la structure du système

Si nous retenons la solution 1 du regroupement par processus maîtres, nous pouvons avoir les règles de comportement suivantes :

Processus Maître
ID : DP1
Nom : Préparation_moyens
Objectif : Oj10
Déclencheurs : EV100
ListeProcessusMétier :{BP1, BP2}
Comportement :

WHEN (START WITH EV100) **DO** BP1 & BP2
WHEN (ES(BP1)=fini AND ES(BP2)=fini) **DO** FINISH

Processus Maître
ID : DP2
Nom : Usiner_pièces
Objectif : Oj20
Déclencheurs : EV101
ListeProcessusMétier :{BP3, BP4, BP5}
Comportement :

WHEN (START WITH EV101) **DO** BP3
WHEN (ES(BP3)=fini) **DO** BP4
WHEN (ES(BP4)=fini) **DO** BP5
WHEN (ES(BP5)=fini) **DO** FINISH

Processus Maître
ID : DP3
Nom : Valider_pièces_usinées
Objectif : Oj30
Déclencheurs : EV132
ListeProcessusMétier :{BP6, BP7, EA8, BP9}
Comportement :

WHEN (START WITH EV132) **DO** BP6 & BP7 & EA8
WHEN (ES(EA8)=fini) **DO** BP9
WHEN (ES(BP9)=fini) **DO** FINISH

La définition des règles de comportement est possible à travers l'interface graphique du framework correspondant au composant « Comportement ».

Ces règles sont représentées graphiquement par des jonctions entre les différents processus par la Figure V.4-6.

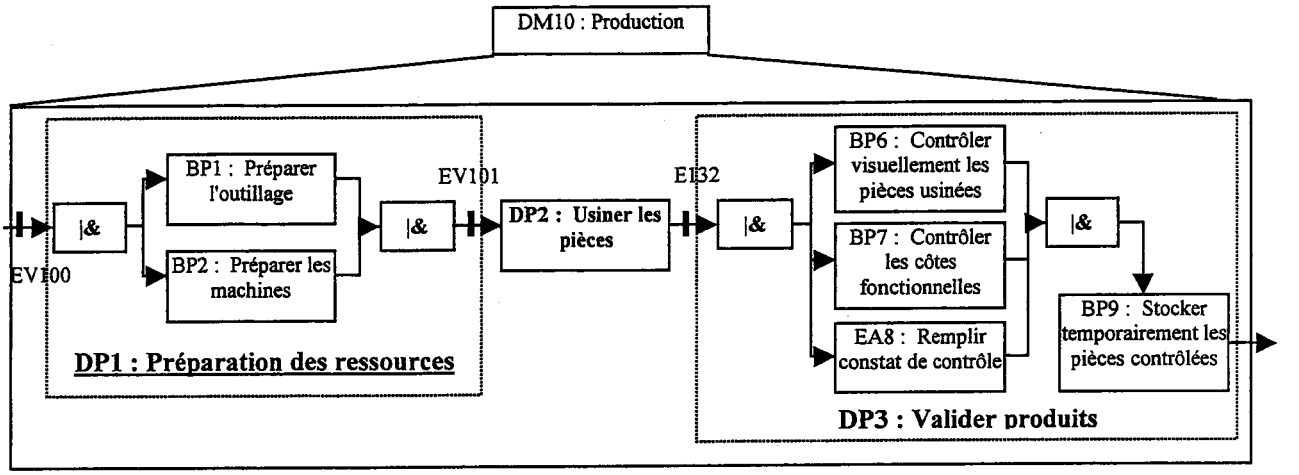

Figure V.4-6 – Solution retenue de regroupement des processus métier du domaine « Production »

La structure du système étant réalisée, le modélisateur doit définir les entrées/sorties et les ressources nécessaires pour chaque processus métier et activité. Rappelons que cette étape peut être réalisée à l'aide des deux composants « Gestion d'Information » et « Gestion de Ressources ».

D'autre part, le composant « Structure d'Organisation » permet la définition de la structure d'organisation par le biais des cellules d'organisation (l'ensemble des services, divisions, départements) et les unités d'organisation.

À chaque unité et cellule, le responsable d'organisation peut attribuer des responsabilités et des autorités au système, domaines, processus maîtres et métier et aux activités.

L'interface graphique du framework correspond aux composants « Gestion d'Information », « Gestion de Ressources » et « Structure d'organisation » n'a pas été encore développée.

Dans ce qui suit, nous représentons l'utilisation du framework de CIMOSA que nous avons développé pour obtenir les résultats de l'exemple.

V.4.2. UTILISATION DU FRAMEWORK

Appliquons l'exemple traité ci-dessus, en utilisant le Framework de CIMOSA.

L'interface utilisateur du framework (Figure V.4-7) permet de réaliser de nouveaux modèles de systèmes d'entreprise et les enregistrer sous format « ssy » ou « xml ». Le format « ssy » est un format texte que nous avons défini et qui signifie Structure SYstème. Ce dernier format, qui est un ancien standard, permet la compatibilité avec d'anciennes applications que ne supportent par le format XML.

Ces modèles peuvent être restituées pour être modifiées ou simplement consultées.

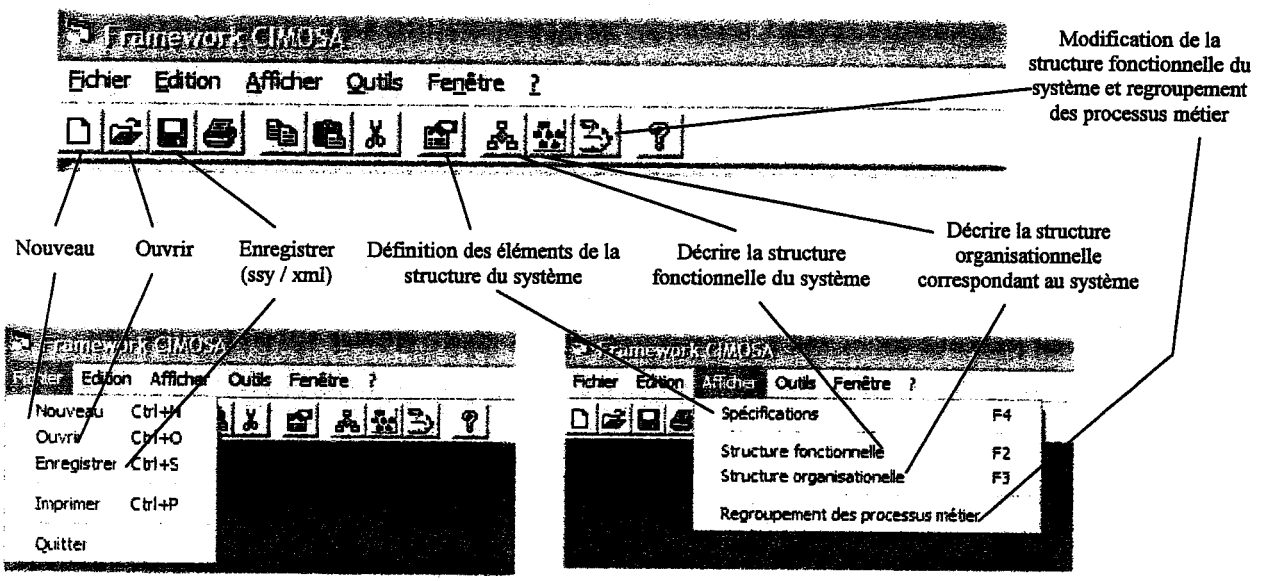

Figure V.4-7 – Commandes du framework

Nous choisissons la fonction « nouveau » pour traiter le nouveau modèle du système de l'exemple.

La première étape consiste à lancer le module S.I.R.S. (Figure V.4-8) pour donner les différentes solutions possibles pour structurer le système en domaines et en processus maîtres et métier.

Les solutions proposées sont enregistrées sous le nom de « Appl1.dat » (Figure V.4-9).

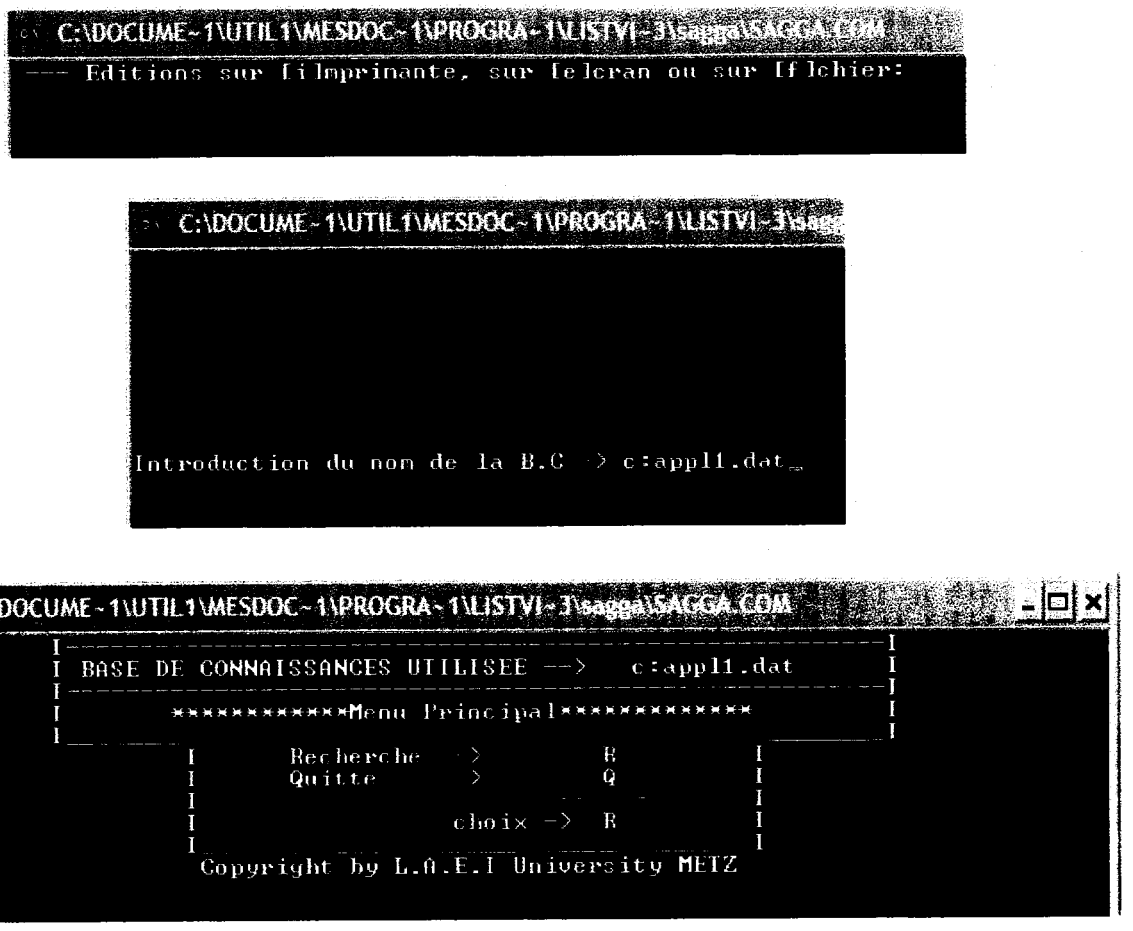

Figure V.4-8 – Version DOS du module S.I.R.S.

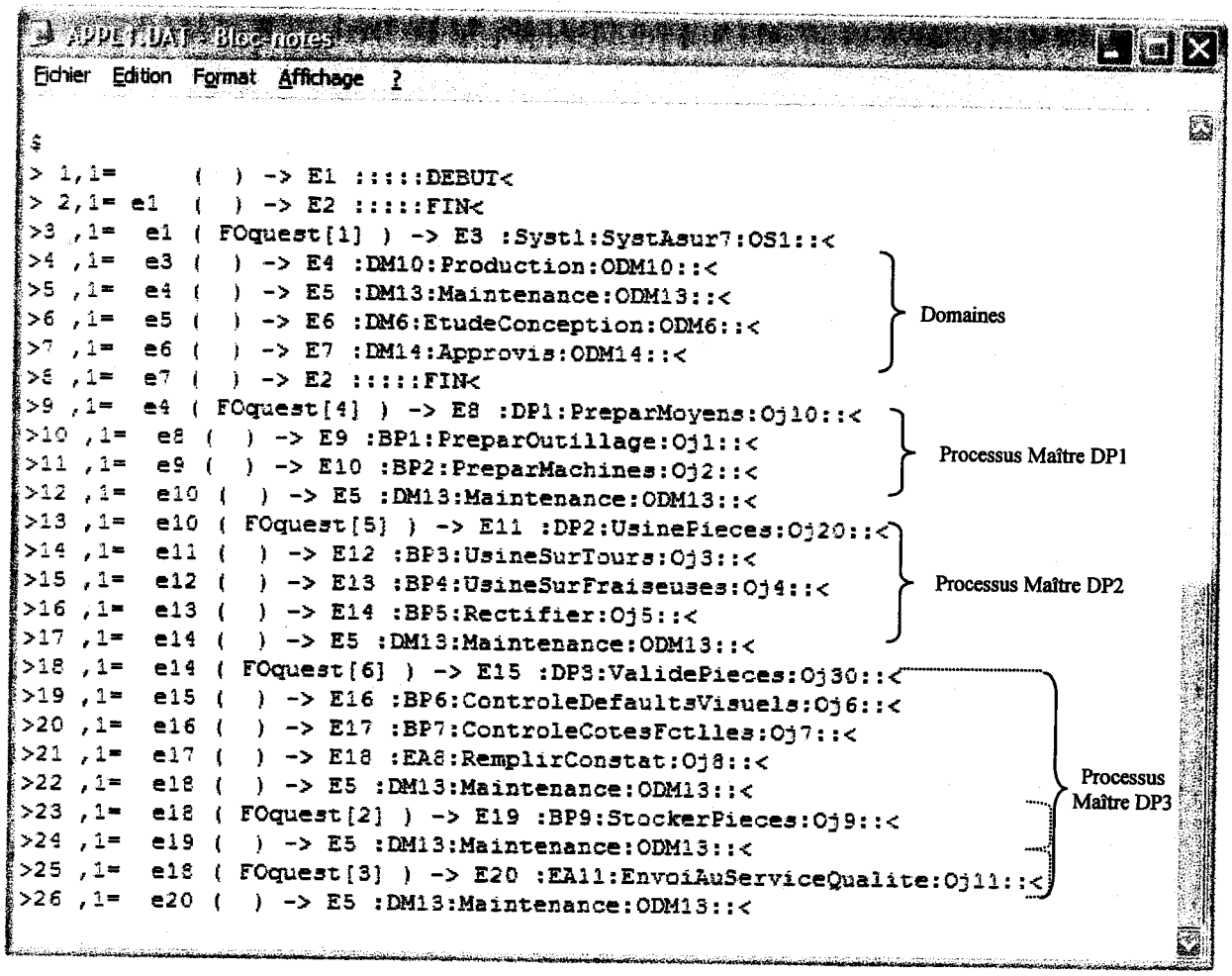

Figure V.4-9 – Fichier contenant les solutions de la structure fonctionnelle du système étudié

Cependant, nous pouvons choisir la solution qui conviennent à la stratégie de l'entreprise en répondant aux questions (Figure V.4-10 et Figure V.4-11) correspondantes aux objectifs des processus maîtres/métier (§ V.4.1.c).

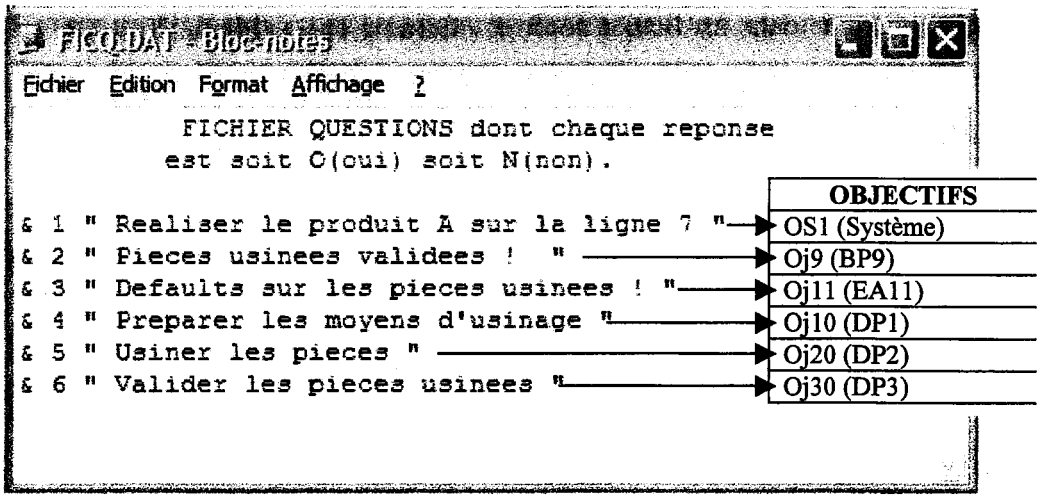

Figure V.4-10 – Fichier des questions correspondantes aux différents objectifs dans le système

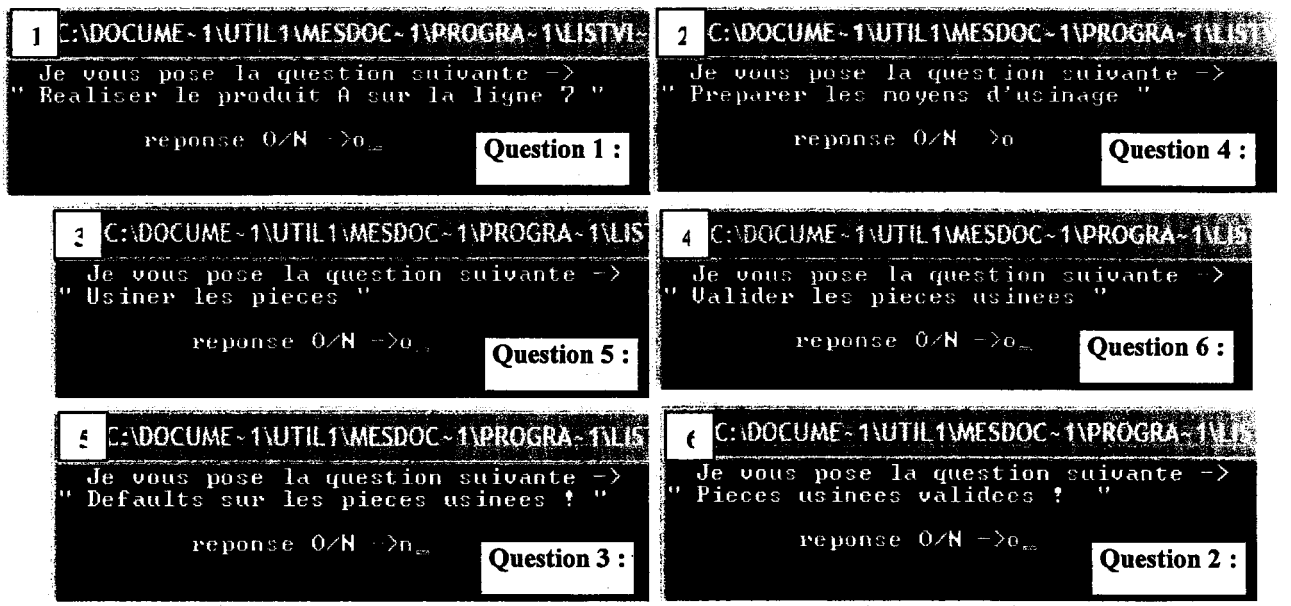

Figure V.4-11 – Questions/Réponses spécifiques au système étudié

Après ce jeu de question/réponse (Figure V.4-11), une solution est retenue (Figure V.4-12). Celle-ci est sauvegardée dans un fichier nommé « Gamme.ope » (Figure V.4-13). Ce dernier est nécessaire pour définir les différents éléments (Objectifs, domaines, processus maîtres, processus métier, événements, informations, etc.) de la structure du système.

Figure V.4-12 – Solution retenue après d'après les objectifs

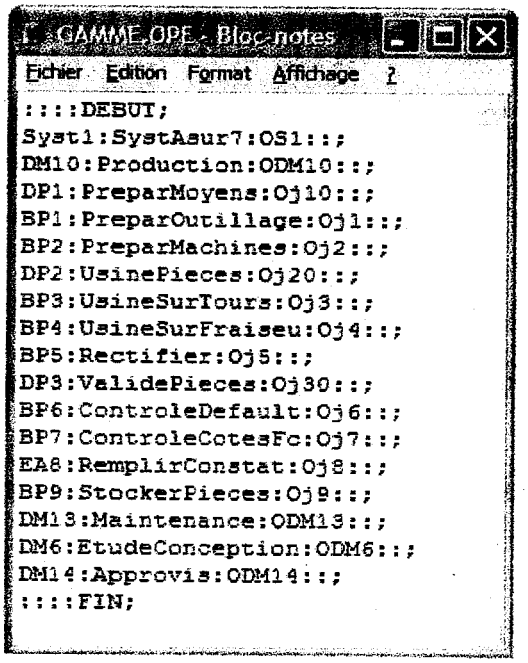

Figure V.4-13 – Fichier « Gamme.ope » contenant la solution retenue

En effet, la solution retenue est représentée graphiquement comme dans les Figures V.4-14, 15, 16, 17, 18 et 19. Cette interface graphique permet une interactivité et une manipulation rapide et précise tout en navigant dans la structure du système. Cette interface correspond au composant « Gestion et Conception des Processus » (§ IV.4.5).

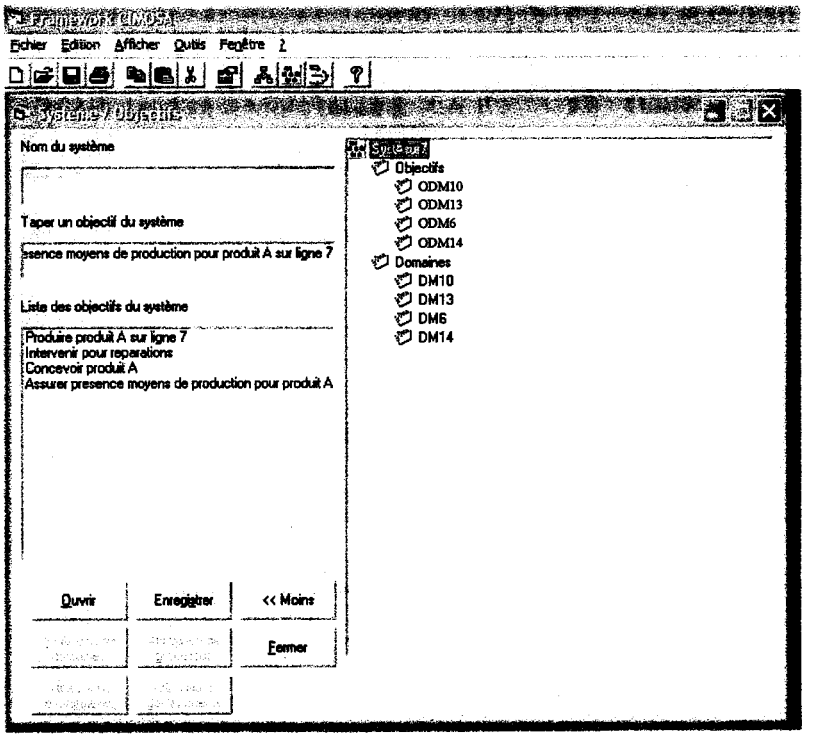

Figure V.4-14 – Représentation interactive du fichier « Gamme.ope »

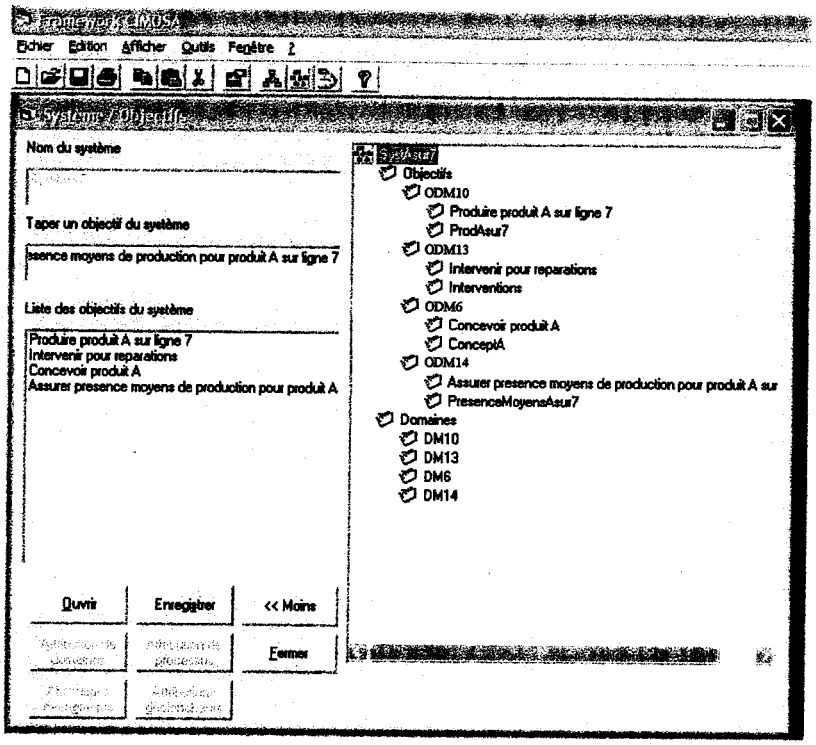

Figure V.4-15 – Définition des objectifs

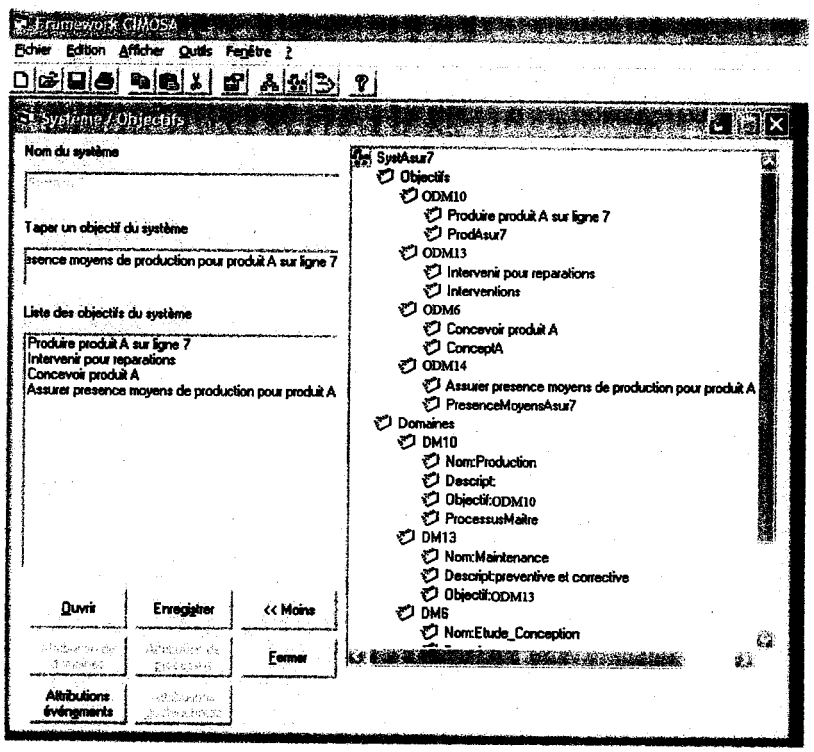

Figure V.4-16 – Développement des domaines

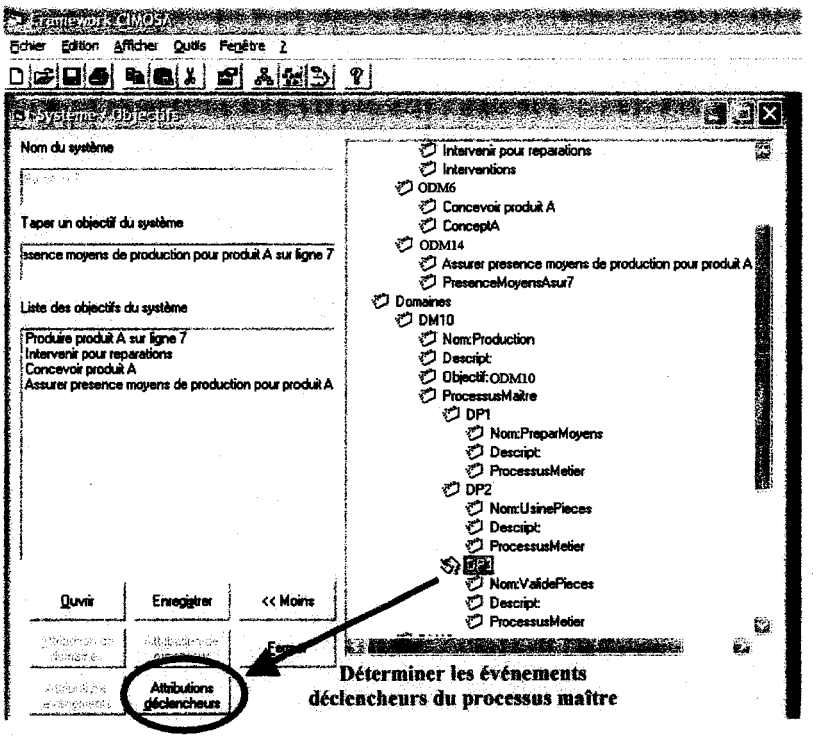

Figure V.4-17- Développement des processus maîtres

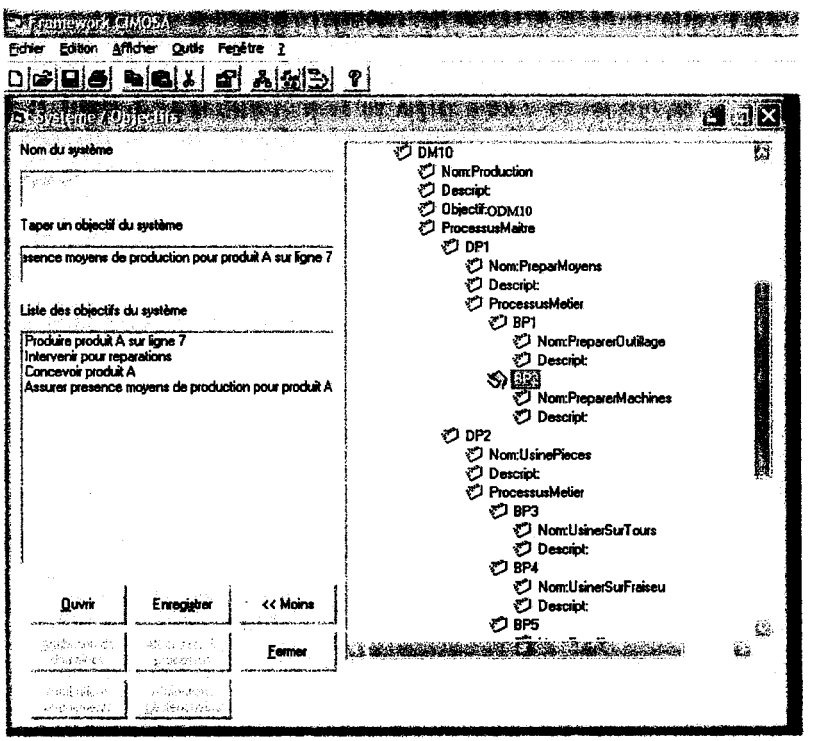

Figure V.4-18 – Développement des processus métier

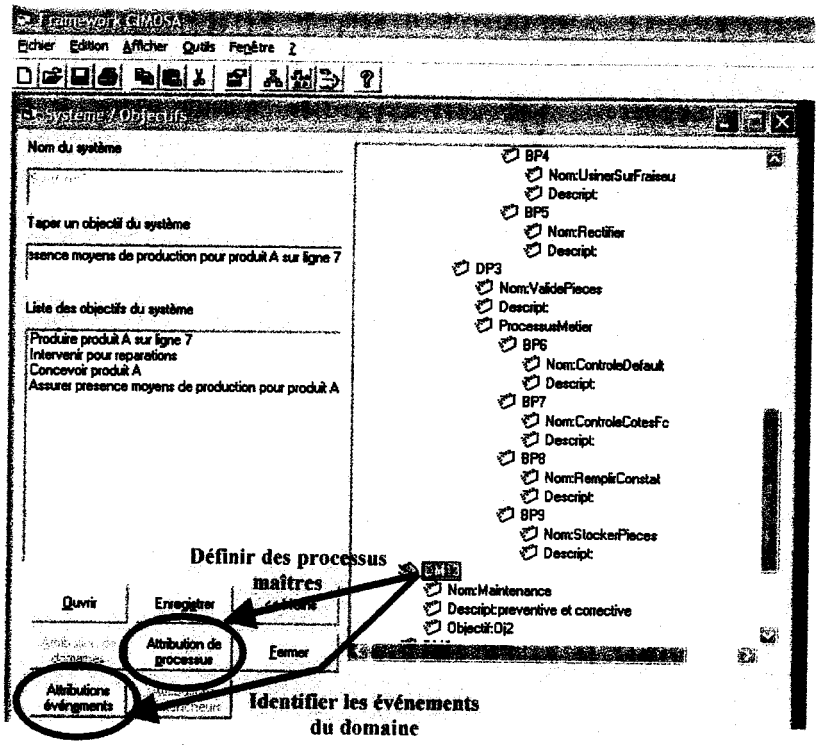

Figure V.4-19 – Commandes pour rajouter des processus maîtres et identifier les événements dans les domaines

Ensuite, nous pouvons définir les objectifs, les processus maîtres, les événements et les processus métier (Figure V.4-20). Cette définition se base sur la spécification des attributs de chacun de ces éléments qui ont été définis lors de la conception des composants (§ IV.4).

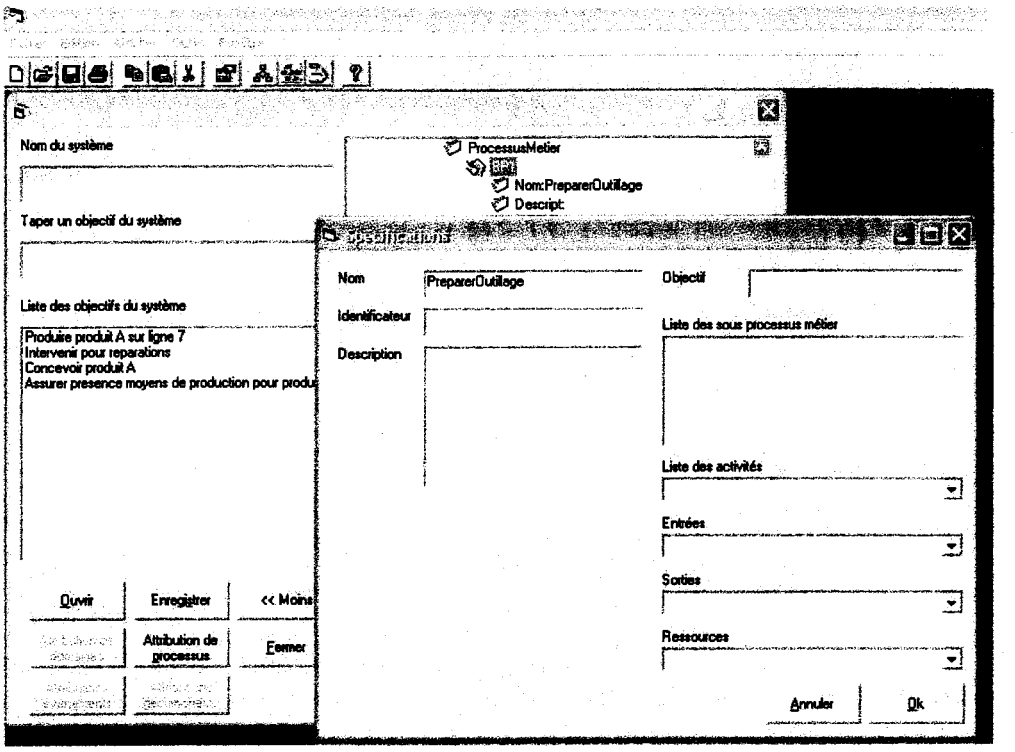

Figure V.4-20 – Définition d'un processus métier

Par exemple, dans le cas d'un processus métier, nous pouvons lui donner une description, modifier son nom, définir son objectif, consulter la liste de ses sous-processus et de ses activités si elles étaient déterminées, identifier les objets en entrées/sorties et préciser le type des ressources requises pour la réalisation d'un tel processus.

Les entrées/sorties et les ressources sont définis, respectivement, par le biais des composants « Gestion de l'Information » (§ IV.4.6) et « Gestion de Ressources » (§ IV.4.7).

Dans le cas d'un domaine, nous pouvons définir les différents événements externes et internes connus. Ceci permet d'identifier les événements déclencheurs de chacun des processus maîtres du domaine (Figure V.4-17 et Figure V.4-19).

Après la définition de chacun des éléments, le modèle peut être sauvegardé pour une consultation ou une modification ultérieure (Figure V.4-21).

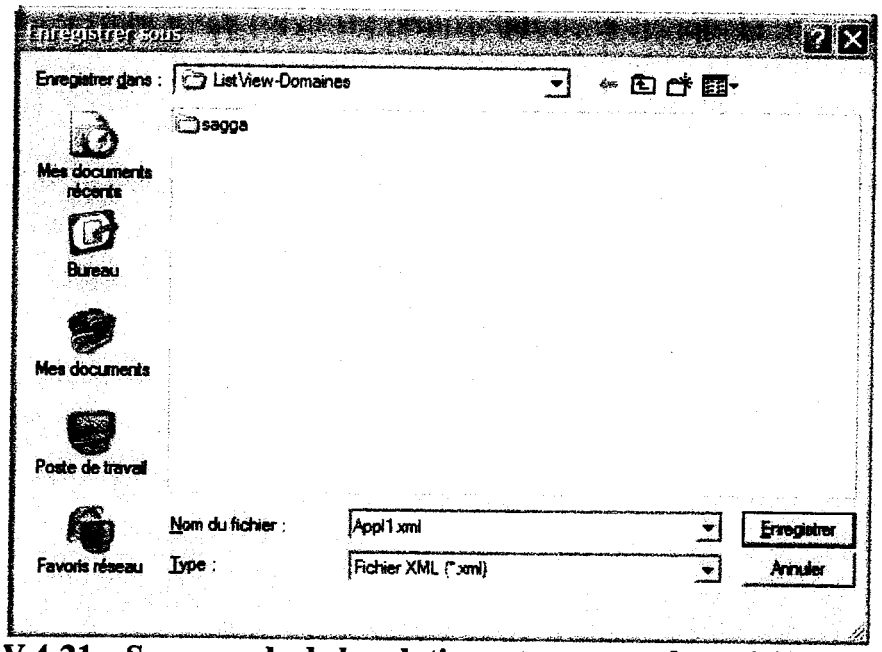

Figure V.4-21 – Sauvegarde de la solution retenue avec les spécifications correspondantes

La Figure V.4-22 représente le format « ssy » de la structure (partielle) du système étudié.

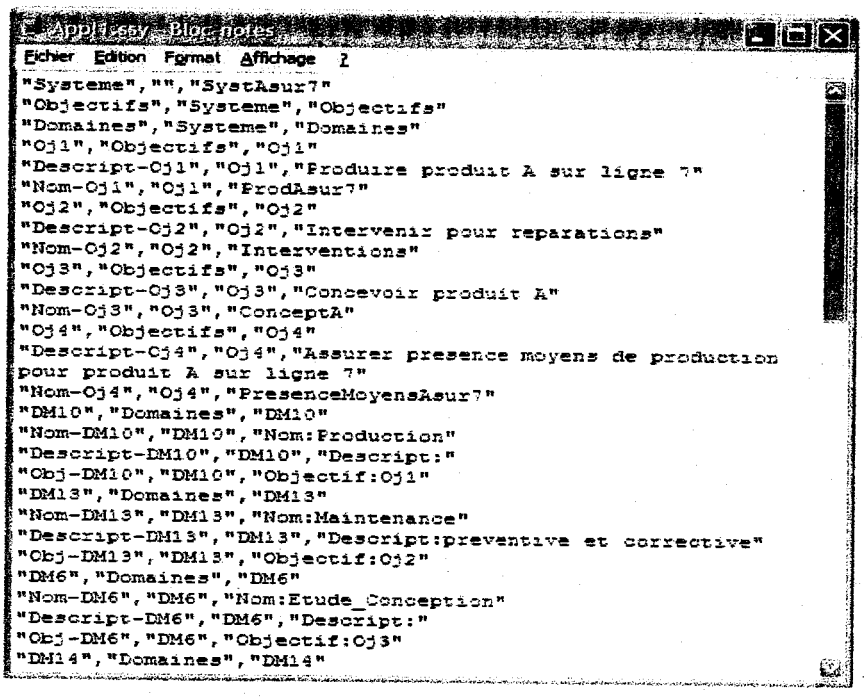

Figure V.4-22 – Fichier « Appl1.ssy » contenant la structure fonctionnelle du système

Le fichier XML reprend la structure représentée dans les Figures V.4-14, 15, 16, 17, 18 et 19. Toutefois, la sauvegarde de ce fichier est cours de correction du fait qu'il y a quelques erreurs de programmation en utilisation l'API DOM XML (§ V.3).

En effet, le fichier XML résultant est celui présenté dans les Figure V.4-23.

```
<?xml version="1.0" encoding="iso-8859-1" ?>
- <Syst1sur7>
    <Objectif ID="ODM10" Nom="ProdAsur7">Produire produit A sur ligne 7</Objectif>
    <Objectif ID="ODM13" Nom="Interventions">Intervenir pour reparations</Objectif>
    <Objectif ID="ODM6" Nom="ConceptA">Concevoir produit A</Objectif>
    <Objectif ID="ODM14" Nom="PresenceMoyensAsur7">Assurer presence moyens de production pour produit A sur ligne
      7</Objectif>
    <Objectif ID="Oj10" Nom="PreparerRessources" />
    <Objectif ID="Oj20" Nom="UsinerPiecesSurMachines" />
    <Objectif ID="Oj30" Nom="ControlerEtValiderPiecesUsinees" />
    <Objectif ID="Oj1" Nom="PlacerOutilsSurMachines" />
    <Objectif ID="Oj2" Nom="ReglerMachines" />
    <Objectif ID="Oj3" Nom="DresserCharioterPieces" />
    <Objectif ID="Oj4" Nom="SurfacerPieces" />
    <Objectif ID="Oj5" Nom="AmeliorerEtatSurfacePieces" />
    <Objectif ID="Oj6" Nom="ControlVisuelDefauts" />
    <Objectif ID="Oj7" Nom="MesureControlCotesFc" />
    <Objectif ID="Oj8" Nom="NoterDefautsDetectes" />
    <Objectif ID="Oj9" Nom="StockageTemp" />
  + <Domaine ID="DM10" Nom="Production" Objectif="ODM10">
    <Domaine ID="DM13" Nom="Maintenance" Objectif="ODM13">Maintenance preventive et corrective des equipements
      d'atelier</Domaine>
    <Domaine ID="DM6" Nom="EtudeConception" Objectif="ODM6">Conception du produit A</Domaine>
    <Domaine ID="DM14" Nom="Approvis" Objectif="ODM14">Approvisionnement des moyens de production et de la matiere
      premiere</Domaine>
    <RelationDomaine ID="R100" Nom="DemandeIntervention" DMS="DM10" DMC="DM13" />
    <RelationDomaine ID="R101" Nom="MaintenancePreventive" DMS="DM13" DMC="DM10" />
    <RelationDomaine ID="R102" Nom="DemandeRessource" DMS="DM10" DMC="DM14" />
    <RelationDomaine ID="R103" Nom="DemandeRessources" DMS="DM13" DMC="DM14" />
    <RelationDomaine ID="R8" Nom="DossierFabrication" DMS="DM6" DMC="DM10" />
  </Syst1sur7>
```

Objectifs et sous-objectifs dans le système

Domaines du système

Relations entre les domaines du système

Figure V.4-23 – Fichier XML « Appl1.xml » décrivant la strcucture fonctionnelle du système étudié

```
- <Domaine ID="DM10" Nom="Production" Objectif="ODM10">
    Preparation de gammes de fabrication, planification et fabrication
    <Evenement ID="EV100" Nom="OrdreFabrication" Heure="" Type="" />
    <Evenement ID="EV101" Nom="FinMiseEnPlaceMoyens" Heure="" Type="" />
    <Evenement ID="EV132" Nom="FinUsinage" Heure="" Type="" />
    <Evenement ID="EVx" Nom="Panne" Heure="" Type="alea" />
    <Evenement ID="EV203" Nom="PiecesUsineesValidees" Heure="" Type="" />
  - <ProcessusMaitre ID="DP1" Nom="PreparMoyens" Objectif="Oj10">
      <Declencheur Evenement="EV100" />
      <ProcessusMetier ID="BP1" Nom="PreparOutillage" Objectif="Oj1" />
      <ProcessusMetier ID="BP2" Nom="PreparMachines" Objectif="Oj2" />
    </ProcessusMaitre>
  - <ProcessusMaitre ID="DP2" Nom="UsinePieces" Objectif="Oj20">
      <Declencheur Evenement="EV101" />
      <ProcessusMetier ID="BP3" Nom="UsineSurTours" Objectif="Oj3" />
      <ProcessusMetier ID="BP4" Nom="UsineSurFraiseu" Objectif="Oj4" />
      <ProcessusMetier ID="BP5" Nom="Rectifier" Objectif="Oj5" />
    </ProcessusMaitre>
  - <ProcessusMaitre ID="DP3" Nom="ValidePieces" Objectif="Oj30">
      <Declencheur Evenement="EV132" />
      <ProcessusMetier ID="BP6" Nom="ControleDefault" Objectif="Oj6" />
      <ProcessusMetier ID="BP7" Nom="ControleCotesFc" Objectif="Oj7" />
      <ProcessusMetier ID="BP8" Nom="RemplirConstat" Objectif="Oj8" />
      <ProcessusMetier ID="BP9" Nom="StockerPieces" Objectif="Oj9" />
    </ProcessusMaitre>
  </Domaine>
```

Evénements du domaine DM10

Déclencheurs du processus maître DP1

Les processus métier (BP1, BP2) du processus maître DP1

Les processus maîtres du domaine DM10

Figure V.4-24 – Domaine « DM10 » développé partiellement dans le fichier « Appl1.xml »

L'interface interactive permet aussi de rajouter de nouveaux éléments à la structure retenue, de modifier ou de supprimer des éléments existants. Ces manipulations mettent à jour automatiquement les fichiers générés par le module S.I.R.S.

Toutefois, l'interface interactive ne permet pas de proposer de nouvelles solutions. Pour le moment, ceci n'est possible qu'avec la version DOS du module S.I.R.S.

Après sauvegarde de la structure du système au format XML, ce dernier est stocké dans la bibliothèque de systèmes d'entreprise.

Dans le cas de cet exemple et après la définition des règles de comportement qui décrivent le flux de contrôle des différents processus, le fichier XML correspondant peut être, en outre, utilisé pour simuler les processus du système étudié au sein d'une application de simulation existante ou à développer.

V.5. CONCLUSION

Dans ce chapitre, nous avons proposé de réaliser un framework correspondant au méta-modèle proposé à la section IV.3, telles que les vues de ce dernier sont implémentées par les composants métier de la section IV.4.

Nous utilisons les composants EMC et l'interface graphique du framework pour réaliser des modèles de système d'entreprise et les stocker en documents XML. Le format XML a été choisi pour permettre à notre framework et nos composants de s'intégrer facilement dans l'entreprise et d'interopérer avec les autres applications existantes.

D'autre part, l'utilisation de XML nous permet d'envisager et d'étendre nos travaux dans le futur vers les services Web ; ce point est l'un des objectifs du projet CAS (§ Chapitre III).

Le framework de CIMOSA permet de décrire un système existant dans l'entreprise dans le but de capitaliser son modèle pour des manipulations (modifications) ultérieures ou de modéliser un nouveau système. Ces modèles peuvent être utilisés pour réaliser des simulations ou pour contrôler l'évolution d'un système.

Le point fort de ce framework c'est qu'il permet de capitaliser le modèle d'un système avec les différentes solutions possibles et de naviguer et manipuler sa structure d'une façon interactive et précise ; ce qui réduit le temps de modélisation d'un système et permet une capitalisation sans perte d'information.

Sur le plan informatique, nous avons développé trente six classe d'objet et dix classes interfaces au niveau des composants (§IV.4.). A part les bugs de compilation, nous avons développé aussi d'autres classes et composants informatiques (ActivX) spécifiques au développement de l'interface graphique du prototype du framework.

CONCLUSION GENERALE

CONCLUSION GENERALE

L'objectif principal de cette thèse est de « distribuer » les modèles de processus d'entreprise dans le but de les partager et les réutiliser par les modélisateurs. L'accès à ces modèles peut être effectué via le Web ou par des frameworks (exemple : langage de programmation, application de simulation, etc.) dans le but de les consulter et/ou les utiliser. Dans ce contexte, nous nous sommes basés sur une architecture de méta-modélisation des processus. L'architecture choisie est CIMOSA. Ce choix n'est pas capital. Cependant, cette architecture permet la réalisation de modèles de processus par instanciation.

En partant du modèle de cette architecture, nous avons proposé un méta-modèle conceptuel et nous l'avons formalisé à l'aide des diagrammes de classes d'UML. Ce diagramme permet de représenter les concepts nécessaires à la modélisation des processus par des classes d'objet. Ce méta-modèle traite les aspects fonctionnels, informationnels, ressources et organisationnels cités dans CIMOSA.

Les aspects fonctionnels sont au cœur de nos préoccupations du fait qu'ils traitent directement les processus d'entreprise. Nous avons proposé une méthodologie pour réaliser une cartographie des processus d'un système d'entreprise. Cette méthodologie est représentée par des diagrammes de séquence mis en place lors de la définition de l'implémentation du méta-modèle (§ IV.4.5.). Les diagrammes de séquence définissent les scénarios de réalisation de la cartographie.

Les définitions des entités manipulées par les processus et des ressources techniques et humaines utilisées pour leur exécution sont traitées respectivement dans les aspects informationnels et de ressources. Alors que l'identification et l'attribution des responsabilités sur le processus peuvent être traitées dans les aspects organisationnels.

Dans ce contexte, nous avons divisé en quatre parties le méta-modèle que nous avons proposé à la section IV.3.3. Chaque partie correspond à chacun des aspects traités par le méta-modèle et est représentée par un paquetage. Ce dernier rassemble les classes d'objet représentant les concepts nécessaires pour traiter les aspects correspondants à ce paquetage.

Ces paquetages peuvent servir de référence pour l'entreprise lors de réalisation de modèles orientés processus. Par conséquent, nous proposons un référentiel qui rassemble ces paquetages.

Nous avons appelé ce référentiel « Référentiel de Composants de modélisation pour l'Entreprise (EMC : *Enterprise Modeling Components*) » du fait que nous proposons d'implémenter chaque paquetage par un composant métier.

Le choix de l'approche par composants permet une meilleure distribution et une maintenance moins complexe que les applications de granularité importante.

Les composants proposés dans cette thèse sont :

- Gestion et conception de processus,
- Gestion d'information,
- Gestion de ressources,
- Structure d'organisation

Une extension du composant « Gestion et conception de processus » permet la description du comportement des processus. Cette extension est aussi un composant que nous avons appelé « Comportement ». Ce dernier implémente les règles de comportement définies dans CIMOSA pour la description de l'exécution d'un processus.

L'implémentation de chaque composant proposé est définie par des classes d'objet qui représentent les concepts de chacun des aspects du méta-modèle.

Chaque composant est défini par son implémentation et son interface.

L'utilisateur ne peut utiliser un composant que via son interface, laquelle a accès à l'implémentation du composant.

Les fonctions offertes par chaque composant sont l'ensemble des méthodes de ses classes d'objet.

Les composants métier proposés sont stockés dans une bibliothèque EMC en tant que modèles partiels. Cette bibliothèque est la concrétisation physique du référentiel EMC proposé.

Chaque composant proposé permet aux modélisateurs de développer des applications client qui implémentent des modèles particuliers de systèmes d'entreprise. Ce développement est réalisé en utilisant le framework CIMOSA qui permet, grâce à son interface graphique, de

guider les modélisateurs pour développer des modèèles de processus d'un système d'entreprise. Ces modèles sont transormés en documents en un format permettant l'interopérabilité avec d'autres applications puis stockés dans une bibliothèque de systèmes d'entreprise.

Le format neutre retenu pour les modèles est XML (eXtensible Markup Language), qui est utilisé dans le domaine des services Web et surtout en e-commerce.

La structure des documents XML d'un modèle de système d'entreprise respecte la structure du méta-modèle proposé (§ IV.3.).

Ces modèles peuvent être exploités pour développer des applications informatiques de simulation de processus, de suivi d'évolution de processus d'un système d'entreprise ou d'un traitement de commandes, etc.

Les composants peuvent être utilisés hors du framework pour subvenir aux besoins de développer des applications clients.

Dans cette thèse, nous avons développé une interface graphique du framework CIMOSA. L'interface en cours traite surtout les apsects fonctionnels d'un système d'entreprise.

La distribution des modèles de systèmes d'entreprise permet aussi aux différents acteurs métier de valider un modèle en fonction de leurs rôles et leurs savoir-faire. Dans ce cas, les développeurs (modélisateurs) des modèles reçoivent les différentes suggestions et les propositions pour faire les modifications nécessaires en utilisant le framework de CIMOSA.

Chaque fois que le contenu du modèle de système d'entreprise change, une mise à jour de celui est déclenchée automatiquement au sein du framework, en particulier de la famille de solutions des regroupements de processus métier. Ceci démontre bien que la maintenance (mise à jour, modification, suppression, ajout) de modèles d'entreprise est l'un des points fort du framework de CIMOSA.

D'autre part, la maintenance du framework est moins compliquée que les applications classiques. En effet, quand il y a des modifications au niveau d'un composant selon les besoins d'entreprise ou selon d'autres paramètres (exemple : ISO), l'interface graphique du framework s'adapte aux modifications réalisées dans les composants. Par conséquent, la maintenance du composant reste facile et rapide relativement aux applications classiques (qui se basent sur un développement procédural ou structurel statique) qui doivent être récrites lors de modifications.

Perspectives

Le framework développé doit être évolué et amélioré pour permettre une modélisation plus approfondie.

Dans ce contexte, deux projets ont été lancés au cadre de DEA pour l'année prochaine :

- Le premier consiste à améliorer l'interface utilisateur du module S.I.R.S. dans le but d'avoir une interactivité semblable au reste du framework.
- Alors que le deuxième sera focalisé sur la continuation de la définition de la structure fonctionnelle d'un système jusqu'au niveau des activité et des opérations fonctionnelles et le développement l'interface graphique correspondant au composant « Structure d'Organisation » pour déterminer les entités organisationnelles correspondantes au système à modéliser.

L'utilisation des documents XML correspondants aux modèles de systèmes d'entreprise nous permet d'étendre nos travaux futurs vers les services Web ; ce point est l'un des objectifs du projet CAS (§ Chapitre III).

Les services Web dans le domaine de modélisation des processus en entreprise n'ont pas encore vu le jour. Cependant, il faut noter que des travaux qui utilisent le format XML ont déjà été menés pour permettre l'interopérabilité entre les applications informatiques de modélisation d'entreprise. Nous pouvons citer entre autres, le projet européen UEML (Unified Enterprise Modeling Language) [UEML, 03] qui traite cette problématique d'interopérabilité.

REFERENCES BIBLIOGRAPHIQUES

A

[Abdmouleh, 00] Abdmouleh A., Les méthodes orientées objets. Rapport de DEA, Université Henri Poincaré, Nancy1, Nancy, France, septembre 2000.

[Abdmouleh, 02] Abdmouleh A, Spadoni M., F. Vernadat., Proposition of a Repository for Enterprise Modeling: (EMC) Enterprise Modeling Components. CARS&FOF2002, Porto, Portugal, 3-5 juillet 2002, pp.631-638.

[Abdmouleh, 03] Abdmouleh A, Spadoni M., F. Vernadat., Distributed Client/Server Architecture For Implementing A meta-model CIMOSA-Based. Computational Engineering in Systems Applications (CESA), Lille, France, 9-11 juillet 2003, CD-ROM .

[ACNOS, 96] ACNOS, Intégration des ACtivités NOn Structurées dans la modélisation des systèmes de production. Action incitative du D.S.P.T.8 en productique, Rapport final, ENSAIS, Strasbourg, 1996.

[ACNOS, 97] ACNOS, Intégration des ACtivités NOn Structurées dans la modélisation des systèmes de production. Rapport final, ENSAIS, Strasbourg, 1997.

[AFNOR, 92] AFNOR, *Gérer la qualité : concepts et terminologie*, tome 1, AFNOR, Paris, 1992.

[AMICE, 93] AMICE, *CIMOSA : Open Systems Architecture for CIM.* Springer-Verlag, Berlin, 1993.

B

[Bachman, 00] Bachman F., Bass, L., Buhman, S., Comella-Dorda, S., Long, F., Seacord, R.c., Wallnau, K.C., Technical Concepts of Component-Based Software Engineering. Technical Report CMU/SEI-2000-TR-008, Software Engineering Institute, Carnegie Mellon University, 2000.

[Berchet, 00] Berchet C., Modélisation pour la simulation d'un système d'aide au pilotage industriel. Thèse de doctorat, Institut National de Polytechnique de Grenoble, Décembre 2000.

[Berio, 99] Berio G., Vernadat F., New developments in enterprise modelling using CIMOSA. *Computers in Industry*, Vol. 40, No. 2-3, 1999, pp. 99-114.

[Berio, 01] Berio G., Vernadat F., Enterprise Modelling with CIMOSA: functional and organization aspects. *Production Planning & Control*, Vol. 12, No. 2, 2001, pp. 128-136.

[Berrah, 01] Berrah L., Clivillé V., Harzallah M., Haurat A., Vernadat F., PETRA : un guide méthodologique pour une démarche de réorganisation industrielle. LGIPM & LLP, décembre 2001.

[Booch, 91] Booch G., *Object-Oriented Design with Applications*. The Benjamin/Cummings Publishing Company, Inc.1991.

[Booch, 00] Booch G., Rumbaugh J. *Le guide de l'utilisateur UML*. Eyrolles, Paris, 2000 (Traduction de : The Unified Modeling Language User Guide.)

[Boukhores, 02] Boukhores A., Kaszycki A., Munerot S., Poublan L., Laplace J., *XML la synthèse : intégrez XML dans vos architectures*. DUNOD Informatique, Paris, 2002.

[Brereton, 00] Brereton P., Budgen D., Component-base systems: A classification of issues. *IEEE Computer*, November, 2000, pp. 54-62.

[Brown, 97] Brown A.W., & Short K., On Components and Objects: The Foundations of Components Based Development. Proc.5th International Symp. Assessment of Software Tools and Technologies, *IEEE Computer Science Press*, Los Alamitos, Californie, 1997, pp. 112-121.

C

[Cattan, 02] Cattan M., Idrissi N. & Knockaert P., *Maîtriser les processus de l'entreprise, guide opérationnel.* 3ème édition, Les Editions d'organisation. Paris 2002.

[CEN, 90] CEN TC 310/WG 1, ENV 12204: Computer-Integrated Manufacturing. Systems Architecture, Framework for Enterprise Modeling. CEN/CENELEC, Bruxelles, 1990.

[Chaugule, 01] Chaugule A., A User-Oriented Enterprise Process Modeling Language. Thèse de Master, Oklahoma State University, Stillwater, OK, 2001.

[CIMOSA, 94] CIMOSA Association, *CIMOSA Technical Baseline.* Böblingen, Allemagne, 1994.

[Cockburn, 01] Cockburn A., *Rédiger des cas d'utilisation efficaces.* Eyrolles, Paris, 2001.

[Crnkovic, 02] Crnkovic I., Larsson, M., *Building Reliable Component-based Systems.* Artech Publishing House, Boston, 2002.

[Curtis, 92] Curtis B., Kellner M.I., Over J., Process Modeling. *Communications of the ACM*, Vol. 35, No. 9, 1992, pp. 75-90.

D

[Davenport, 93] Davenport T.H., Process Innovation : Reegineering work through Information Technology. Harvard Business School, Boston, US, 1999.

[Deblock, 03] Deblock F. De la fusion au sein des processus métier. JDNet de Benchmark Group, http://solutions.journaldunet.com , novembre 2003.

[Dogac, 98] Dogac, A., Kalinichenko, L., Ozsu, T., Sheth, A., *Workflow Management Systems and Interoperability*. Spinger Verlag, Berlin, 1998.

[Doumeingts, 84] Doumeingts G. La méthode GRAI. Thèse d'état, Université de Bordeaux I, France, 1984.

E

[ENV, 96] ENV 12204, Constructs for Enterprise modelling. CEN/CENELEC, Bruxelles, 1996.

[Eimaraghy, 92] Eimaraghy, Integrating assembly planning and scheduling CAPP-related issues. Annals of CIRP, vol. 41/1, 1992.

[Eriksson, 00] Eriksson H.E., Penker M., *Business Modeling wih UML-Business Patterns at work*. John Wiley & Sons, NY, 2000.

[Ezran, 99] Ezran M., Moriso M., Tully C., *Réutilisation logicielle*. Eyrolles, Paris, 1999.

F

[Frecher, 03] Frecher D., Segot Jacques, Tizzolino Ph., *100 questions pour comprendre et agir - Les processus*. AFNOR, Paris, 2003.

[Fremantle, 02] Fremantle P., Weerawarana S., Khalaf R., Enterprise Services, Enterprise Components and Services. *Communications of the ACM*, Vol. 45, No.10, octobre 2002, pp. 77-82.

G

[Gamma, 95] Gamma E., Helm R., Johnson R., Vlissidies J., *Design Patterns, Elements of Reusable Object-Oriented Software*. Addison-Wesley, Reading, MA, 1995.

[Gartner, 03] Gartner GROUP, http://www4.gartner.com/Init, novembre 2003.

[Geib, 97] Geib J-M., Gransart Ch., Merle Ph., *Corba, des concepts à la pratique*. Masson, Collection Informatique. Paris 1997.

H

[Hachette, 01] © 2001 Hachette Multimédia / Hachette Livre.

[Hammer, 93] Hammer M., Champy J., *Reengineering the Corporation. Manifests for Business Revolution*, Harper New-York, NY, 1993.

[Harrington, 91] Harrington H.J., Business Process Improvement, The Breakthrough Strategy for Total Quality, Productivity and Competitiveness, McGraw-Hill, NY, 1991.

[Harzallah, 00] Harzallah M., Modélisation des aspects organisationnels et des compétences pour la réorganisation d'entreprises Industrielles. Thèse de doctorat, Université de Metz, Metz, France, mai 2000.

[Harzallah, 02] Harzallah M., Vernadat F., IT-based competency modeling and management: From theory to practice in enterprise engineering and operations. *Computers in Industry*, Vol. 48, No. 2, 2002, pp. 157-179.

I

[IFAC, 97] IFAC-IFIP Task Force. GERAM: Generalised Enterprise Reference Architecture and Methodology. Version 1.4, ISO TC 184/SC5/WG1, N398, août 1997.

J

[Jacob, 94] Jacob G., *Le reengineering de l'entreprise, l'entreprise reconfigurée.* HERMES, Paris, 1994.

[Jacobson, 92] Jacobson I., Christerson M., Jonson P., Övergaard G., *Object-Oriented Software Engineering: A Use Case Driven Approach.* Addison-Wesley, Reading, MA, 1992.

[Jacobson, 97] Jacobson I., Griss, M.L., Johnsson, P., *Software Reuse, Architecture, Process and Organization for Business Success.* Addison Wesley and ACM Press, 1997.

[Jagou, 93] Jagou P., *Concurrent Engineering – La maîtrise des coûts, des délais et de la qualité.* Hermes, Paris, 1993.

[JDNet, 02] JDNet, Les entreprises françaises privilégient les processus transversaux. http://solutions.journaldunet.com/0212/021206_idc.shtml, Benchmark Group, décembre 2002.

[JDNet, 03] JDNet, Enquête: XML plutôt associé aux échanges B2B qu'à l'intégration applicative. Benchmark Group, février 2003.

[JDNet, 03a] JDNet, Sondage: Gérer des processus : modéliser l'existant d'abord. http://solutions.journaldunet.com/0304/030409_sondage.shtml, Benchmark Group, avril 2003.

[Johansson, 93] Johansson H.J., McHugh P., Pendlebury A.J., Wheeler III W.A., *Business Process Reengineering, Breakpoint Strategies for Market Dominance*. John Wiley & Sons Ltd, England, 1993.

[Johnson, 97] Johnson R.E., Frameworks = (Components + patterns). *Communications of the ACM*, Vol. 40, No. 10, 1997, pp. 39-42.

K

[Kobryn, 00] Kobryn C., Modeling components and frameworks with UML. *Communications of the ACM*, Vol. 43, No. 10, 2000, pp. 31-38.

[Kruth, 92] Kruth J.P., Detand J., A CAPP system for nonlinear process plans. Annals of CIRP, vol. 41/1, 1992.

L

[Larsen, 99] Larsen G., Designing component-based frameworks: using patterns in the UML. *Communications of the ACM*, Vol. 42, No. 10, 1999, pp. 38-45.

[Linthicum, 00] Linthicul D.S., *Enterprise Application Integration*. Addison Wesley, Reading, MA, 2000.

[Lutz, 00] Lutz J.C., EAI Architecture Pattern. EAI Journal, mars 2000.

M

[Manouvrier, 01] Manouvrier B., *EAI : Intégration des Application d'Entreprise*. Hermes, Paris, 2001.

[Martin, 03] Martin M., Intervenir à l'interface des processus pour améliorer leur performance. BPMS.info, JDNet de Benchmark Group, http://solutions.journaldunet.com, janvier 2003.

[Mathieu, 00] Mathieu S., *Comprendre les normes ISO 9000 version 2000.* AFNOR, Paris, 2000.

[Mintzberg, 82] Mintzberg H., *Structure et dynamique des organisations.* Les Editions d'Organisation, Paris, 1982.

[Morgenthal, 00] Morgenthal J.P., *Enterprise Application Integration with XML and JAVA.* Prentice Hall, Englewood Cliffs, NJ, 2000.

[Mougin, 02] Mougin Y., Peyrat O., *La cartographie des processus – Maîtriser les interfaces.* Les Editions d'Organisation, Janvier 2002.

N

[Nikunj, 04] Nikunj P.D., Manjunath K., William J.K., Eswar S., Toward an Integrated Framework for Modeling Enterprise Processes. *Communications of the ACM*, Vol. 47, No. 3, 2004, pp. 83-87.

O

P

[Proth, 95] Proth J.M., Xie X., *Les réseaux de Petri pour la conception et la gestion des systèmes de production.* Masson, Paris, 1995.

Q

R

[Roboam, 93] Roboam M., La méthode GRAI, principes, outils, démarche et pratique. Teknéa, Toulouse, 1993.

[Roques, 00] Roques P., Vallee F., *UML en Action*. Eyrolles, Paris, 2000.

[Ross, 77] Ross D.T., Structured Analysis (SA): A Language for communicating ideas. *IEEE Transactions on Software Engineering*, SE-3, 1977.

[Rumbaugh, 91] Rumbaugh J., Blaha M., Premerlani W, Eddy F., *Object Oriented Modeling and Design*. Prentice Hall International, 1991.

S

[Schneider, 98] Schneider G., Winters J.P., *Applying Use Cases*. Addison-Wesley, Reading, MA, 1998.

[Scheer, 99] Scheer A.W., *ARIS-Business Process Modeling. Springer-Verlag*. Berlin, 1999.

[Shorter, 94] Shorter D., An evaluation of CIM modelling constructs; Evaluation report of constructs for views according to ENV 40003. *Computers in Industry*, Vol. 24, No. 2-3, 1994, pp.159-236.

[Shorter, 00] Shorter D., Revision of ENV 40003, Second Internal Draft. Enterprise Integration-framework for Enterprise Modeling, décembre 2000.

[Spadoni, 00] Spadoni M., Youssef A., Vernadat F., Abdmouleh A., SYPA : Adaptive Objective-based Production Control System. International Conference on

Engineering Design and Automation (EDA'2000), Orlando/Floride, 3-5 août 2000, pp.593-598.

[Spadoni, 01] Spadoni M., Gardoni M., Abdmouleh A., Nsingi-Menamo T., Formalisme Générique pour la Représentation d'Informations Structurées. *Journal Européen des Systèmes Automatisés*, Vol. 35, No. 6, 2001, pp.747-781.

[Sprott, 00] Sprott D., Componentizing the enterprise application Packages. *Communications of the ACM*, Vol. 43, No. 4, 2000, pp. 63-69.

[Szyperski, 98] Szyperski C., *Component Software: Beyond Object-Oriented Programming*. Addison-Wesley, Reading, MA, 1998.

T

[Tkach, 96] Tkach D. and Putticj R., *Object Technology in Application Development*. Addison Wesley, Reading, MA, 1996.

[Tomas, 99] Tomas J.L., *ERP et progiciels intégrés*. InterEdition, 1999.

U

[UEML, 03] UEML: Unified Enterprise Modeling Language, http://www.ueml.org, 2003.

V

[Vernadat, 96] Vernadat F., *Enterprise Modeling and Integration: Principles and Applications*. Chapman&Hall, London, 1996.

[Vernadat, 99] Vernadat F., *Techniques de Modélisation en Entreprise : Applications aux Processus Opérationnels*. ECONOMICA, Paris, 1999.

W

[Wang, 94] Wang S., OO Modeling of business processes, Object-oriented Systems Analysis. *Information Systems Management*, Spring 1994, pp.36-43.

[Williams, 92] Williams T.J., The Perdue Enterprise Reference Architecture. Instrument Society of America, Research triangle Park, NC, 1992.

[Workflow, 96] Workflow Management Coalition, Workflow Standard-Interoperability abstract specification. Doc. No. WFMC-TC-1012, Version1.0, http://www.aiai.ac.uk/wfmc, October 1996.

[Workshop, 04] Workshop sur la modélisation en entreprise et les services Web, organisé par le groupe GDR ECI avec le laboratoire PRISMA, Université de Lyon, Lyon, 15-16 janvier 2004.

X

Y

[Young, 00] Young M.J., Formation en XML. Microsoft Press, 2000.

Z

[Zachman, 87] Zachman J.A., A framework for information systems architecture. *IBM Systems Journal*, Vol. 26, No. 3, 1987, pp. 276-292.

ANNEXES

TABLES DES MATIERES

ANNEXE A

REGLES DE COMPORTEMENT DANS CIMOSA

I. INTRODUCTION

Définir un ensemble de règles de comportement qui sont utilisées pour représenter le flux de contrôle associé aux processus maîtres et métier [Vernadat, 99].

Les règles de comportement sont utilisées pour définir le comportement de l'entreprise.

Elles sont utilisées pour définir les décisions logiques requises pour déterminer les séquences des processus et des activités pour exécuter les tâches générales d'un processus maître ou d'un domaine.

La syntaxe de cette règle est comme suit :

- WHEN 'état final de processus métier/activité ET condition de déclenchement validée',
- DO 'déclencher un ou plusieurs processus métier/activités'

Si l'exécution d'un processus métier/activité donne un état final qui n'a pas été défini dans l'ensemble des règles de comportement en cours, CIMOSA doit arrêter le déroulement de ces règles et se reporte à l'autorité responsable (définie dans la vue organisation de CIMOSA [CIMOSA, 94]) pour corriger l'erreur.

Une règle de comportement est représentée comme suit :

WHEN (condition de déclenchement) DO action

II. DECLENCHEMENT DE PROCESSUS

Les règles de comportement décrivent le déclenchement des différents processus maîtres, processus métier et activités par des événements ou par synchronisation de plusieurs actions en amont. Dans ce qui suit, nous présentons les différents types de déclenchements où les règles de comportement sont exploitées.

Notons bien que les cas suivants sont utilisés dans le cas de *processus structurés* que nous avons utilisés pour la conception du composant « Comportement » (§ IV.4.9.).

II.1. DECLENCHER UN PROCESSUS MAITRE

- WHEN (START WITH Evénement_a) DO Efy
- WHEN (START WITH Evénement_b AND Evénement_b) DO Efz
- WHEN (START WITH Evénement_b OR Evénement_b) DO Efz

→ WHEN (START WITH Evénement_b) DO Efz
WHEN (START WITH Evénement_c) DO Efz

II.2. DECLENCHER UN PROCESSUS METIER

- WHEN (START) DO Efx
- WHEN (ES(Efx)=valeur_1) DO Efy
- WHEN (ES(Efy)=valeur_2 AND Evénement_1) DO EFz

Efx, Efy et Efz sont des processus métiers ou des activités.

START est une condition initiale, utilisé pour déclencher un processus.

II.3. DECLENCHEMENT FORCE

Dans ce cas, une règle active le processus métier/activité suivant indépendamment des états finaux :

- WHEN (ES(Efx)= any) DO Efy

'any' est un mot réservé signifiant: pour n'importe quelle valeur de Efx.

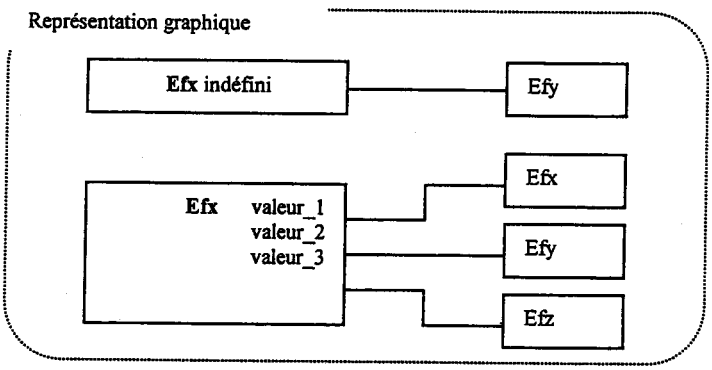

Figure 1 – Déclenchement forcé

II.4. DECLENCHEMENT CONDITIONNEL

Déclencher des actions en tenant compte de plusieurs conditions qui se suivent, comme suit :

- WHEN (ES(Efx)= valeur_1) DO Ef1
- WHEN (ES(Efx)= valeur_2) DO Ef2
- WHEN (ES(Efx)= valeur_3) DO Ef3

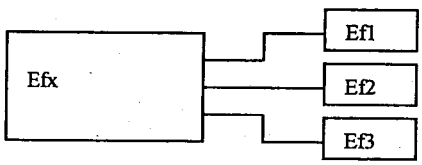

Figure 2 – Déclenchement conditionnel

II.5. DECLENCHEMENT PARALLELE

Déclencher en parallèle un ou plusieurs processus métier/activités, qui peuvent être activités dépendamment.

WHEN (ES(EFx)=valeur_1) DO EF1 & EF2 & EF3

& : signifie le parallélisme

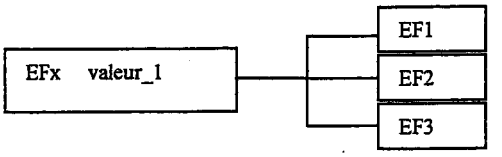

Figure 3 – Décelenchement parallèle

II.6. DECLENCHEMENT EN RENDEZ-VOUS

Déclencher des processus métier/activités lors de conditions de plusieurs validations d'états finaux.

- WHEN (ES(EF1)=valeur_1 AND ES(EF2)=valeur_2 AND ES(EF3)=valeur_3) DO EF4

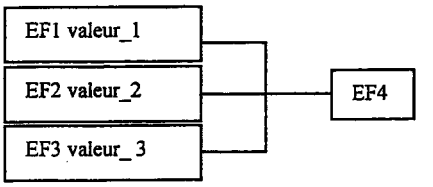

Figure 4 – Déclenchement en Rendez-vous unique

- WHEN (ES(EF10)=valeur_10 AND ES(EF20)=valeur_20) DO EF30 & EF40

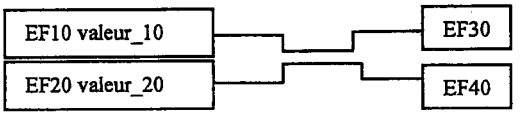

Figure 5 – Déclenchement en Rendez-vous multiple

II.7. DEFINITION DE BOUCLE

Exécuter une fonction en boucle pour une valeur définie, jusqu'à atteindre une autre valeur.

- WHEN (ES(EFx)=valeur_a) DO EFx
- WHEN (ES(EFx)=valeur_b) DO Efy

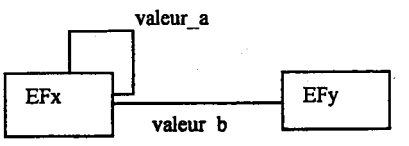

Figure 6 – Définition de boucle

II.8. REGLES DE TERMINAISON

Ces règles déterminent la fin de l'ensemble des règles de comportement et la fin de la réalisation d'un processus maître ou métier.

- WHEN (ES(EFx)=any) DO FINISH
- WHEN (ES(Efy)=valeur_1 AND (ES(Efz)=valeur_2) DO FINISH

ANNEXE B

LES PATRONS METIER

I. INTRODUCTION

Il y a plusieurs types de patrons. Nous pouvons citer les patrons métier, les patrons architecturaux, et les patrons de conception. Ces différents types de patrons ont plusieurs objectifs différents, et ils sont utilisés dans des contextes bien spécifiques.

Les patrons métier sont utilisés dans le domaine métier, pour analyser des situations telles que la modélisation et la structuration des ressources (facture, organisation, information, etc.). Ces patrons sont aussi utilisés pour organiser et relier les processus, les règles métier et les objectifs.

Les patrons architecturaux sont utilisés pour restructurer l'architecture de conception des systèmes d'information, tels que l'organisation des sous systèmes et leur implémentation au plus haut niveau d'abstraction.

Les patrons de conception sont utilisés dans les situations où l'analyse est déjà décrite et définie. Ces patrons aident à produire des solutions techniques flexibles et adaptables. Plusieurs de ses patrons ont été documentés pour traduire les bases de données objets en bases de données relationnelles, créer des hiérarchies de classes flexibles, et pour changer ou étendre les structures de codes.

II. UTILISATION DES PATRONS METIER

La première phase est de résoudre tous les types de problèmes et de comprendre le contexte et le domaine concerné. Cela peut être achevé en utilisant des méthodes d'analyse, telles que la réduction de la déduction dans la théorie des sciences, la résolution par les méthodes mathématiques, etc. Les patrons métier peuvent être utilisés comme des outils d'aide; premièrement, pour comprendre la situation du problème dans le contexte de la modélisation des métier. Deuxièmement, pour définir comment traiter les problèmes dans cette situation.

Comprendre une situation d'un problème est dépendante de la méthode d'analyse utilisée ; c'est aussi dépendant de la connaissance du modélisateur et de son expérience, des modèles existants et des frameworks, de la distinction des processus et des patrons appris. Ainsi, les patrons sont un complément aux méthodes d'analyse et ils sont utilisables pour l'analyse des situations métier (le niveau de modélisation métier). L'analyse peut être focalisée pour comprendre une situation métier existante à travers la modélisation ; elle peut aussi entreprendre de modéliser un nouveau métier ou un métier validé pour l'implémenter. Il est important de savoir que les patrons métier ne sont pas directement développables en code. Ils sont utilisés pour créer des modèles métier flexibles et compréhensibles qui décrivent la structure et le comportement métier. Ces modèles peuvent être réutilisés comme support pour la création des systèmes d'information pour les métiers correspondants, ainsi le niveau suivant est le développement du code. La conception du système d'information implémente les patrons de conception et architecturaux. Les patrons métier restent toujours manipulables pour re-modéliser un modèle métier existant. De même, des modèles à structurer peuvent être occasionnellement re-modélisés et validés par l'utilisation des patrons. En effet, les patrons métier ne sont pas uniquement un outil de production des modèles, mais aussi une excellente façon d'enseigner les bonnes techniques de modélisation. Les patrons métier sont parfois

combinés entre eux ou adaptés à la situation en cours. D'autre part, il est important de connaître le but principal du patron pour pouvoir faire des combinaisons ou des adaptations. Quand les combinaisons et les adaptations sont incorrectes les patrons ne seront plus utiles pour résoudre les problèmes. Changer ou modifier un patron peut éliminer les avantages de celui-ci ; en effet, ce changement transforme le patron totalement ou partiellement en un patron non testé et non validé.

Il est possible de créer son propre patron, surtout quand le travail dans un domaine particulier ramène à résoudre les mêmes problèmes plusieurs fois. Les patrons se développent des structures et des interactions qui se produisent dans un domaine. Ils peuvent être plus ou moins génériques au domaine.

III. CLASSIFICATION DES PATRONS METIER

Les patrons métier peuvent être classifiés, en tenant compte de la modélisation métier, comme suit :

- les patrons des ressources et des règles,
- les patrons des objectifs,
- les patrons des processus.

III.1. LES PATRONS DES RESSOURCES ET DES REGLES

Ces patrons fournissent une aide pour la modélisation des règles et des ressources à travers un domaine métier. Tous les métiers sont traités par des produits et des documents. Les patrons les plus importants sont adressés à ces produits et documents. D'autres patrons sont adressés pour trouver et séparer les concepts du métier de leurs représentations et leur types, objets et valeurs de modélisation.

Voici deux patrons de ressources et règles qui ont été utilisé pour la conception du méta-modèle du référentiel EMC (§ IV.3).

III.1.1. ACTEUR-ROLE

a) Utilisation

Il définit les relations entre un acteur et un rôle et leur séparation lors de la modélisation métier.

b) Structure

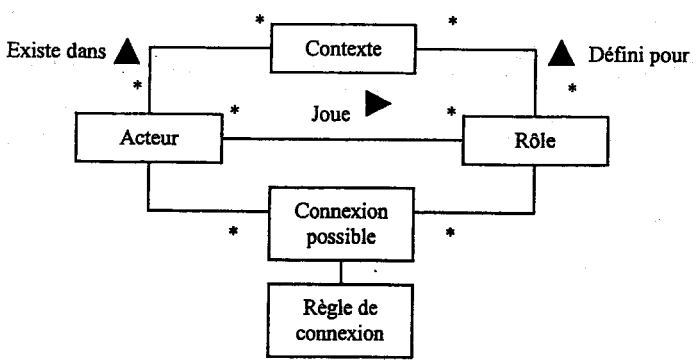

Figure 7 - Structure du patron Acteur-Rôle

- Contexte : situation dans laquelle les acteurs existent et les rôles sont définis,
- Acteur : classe décrivant les acteurs,
- Rôle : description du ou des rôles que le(s) acteur(s) doi(en)t jouer dans un contexte particulier,
- Connexion possible : exprime ou permet les connexions entre les acteurs et les rôles,

Règle de connexion : base pour la classe 'Connexion Possible' :

- XOR : une des plusieurs connexions est autorisée à un instant t,
- AND : tous les rôles peuvent être joués simultanément.

c) Exemple

Un système qui manipule des données doit être protégé et sécurisé ; dans ce cas les rôles des opérateurs système et administrateurs système doivent être séparés. Parce qu'un administrateur peut rajouter ou supprimer des comptes aux utilisateurs du système, avec lesquels un opérateur peut accéder aux données du système. Dans ce cas, un acteur qui a le rôle administrateur ne doit jamais avoir le rôle d'un opérateur, pour éviter le risque qu'un administrateur crée un compte pour lui-même, ce qui lui permettra d'accéder à des données sensibles dont il n'a pas l'autorité et les aptitudes à manipuler. Dans cet exemple, la séparation des rôles élimine le risque d'insécurité et de perte de données.

Par exemple, dans une Entreprise, Jim est un administrateur système et Michel, Jacques et François des concepteurs utilisant le système. Ces trois derniers n'ont pas le droit d'avoir le rôle d'administrateur, alors que Jim n'a le droit qu'à ce rôle. François peut utiliser le système comme technicien ou concepteur. Or, quand il se connecte comme technicien, il n'a pas le droit de changer des informations et des données développées et intégrées par un concepteur.

Alors, François ne peut pas jouer le rôle de technicien et de concepteur en même temps, par rapport au système.

Jim, Michel, Jacques et François sont des acteurs. 'Administrateur', 'Concepteur' et 'Technicien' sont des rôles.

Cela peut être représenté par un diagramme d'objets (Figure 7) qui est une instance du diagramme de classes représentant la structure du patron (Figure 8).

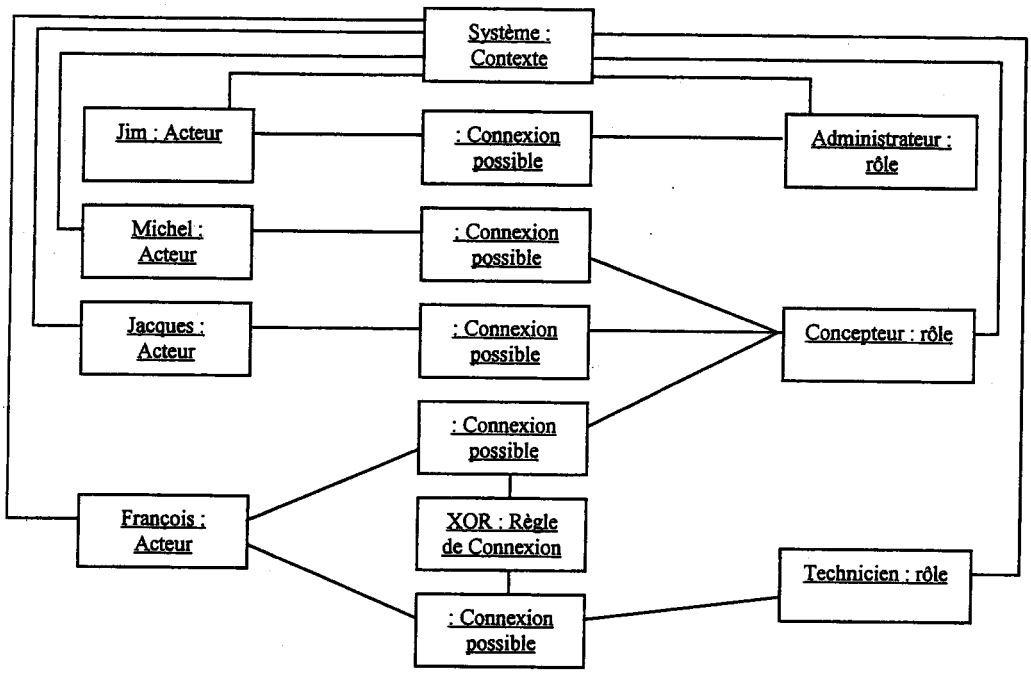

Figure 8 - Exemple de connexions acteurs-rôles dans un système

III.1.2. ORGANISATION

a) Utilisation

Créer des diagrammes d'organisation flexibles et qualitatifs dans des modèles orientés objets.

b) Structure

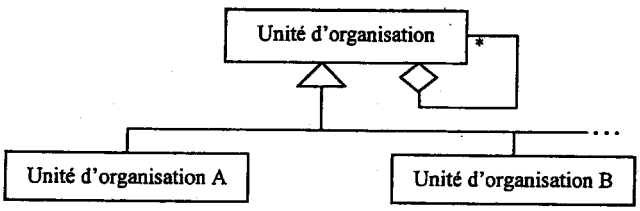

Figure 9 - Structure du patron Organisation

Unité d'organisation : les instances (A) et (B) de cette classe définissent des départements, des directions, des services, etc.

c) Exemple

Soit l'organisation de la C&A. Elle est représentée par un diagramme d'objets (Figure 10) qui est une instance du diagramme de classes décrit dans la structure du patron (Figure 9).

C&A contient les unités organisationnelles suivantes :

- Département de finances : responsable des finances de l'Entreprise, des références de crédits, etc.

- Département d'achats : responsable d'acheter des produits de l'extérieur et développer des relations avec des sous-traitants.
- Département de production : il produit des articles, ou assemble ceux qu'il fabrique ou qu'il achète de chez les sous-traitants.
- Département du système : responsable de l'infrastructure technique et de la technologie d'informations de l'Entreprise. Il contient les analystes systèmes, les programmeurs, les architectes systèmes, etc.
- Département des ventes : divisé en deux sous-départements ; télé-ventes et vente par Internet.
- Département de produits : il développe les nouveaux produits.
- Département de marketing : responsable de l'implémentation du plan de marketing.

En utilisant ce patron, C&A peut facilement ajouter, changer ou supprimer les unités quand c'est nécessaire.

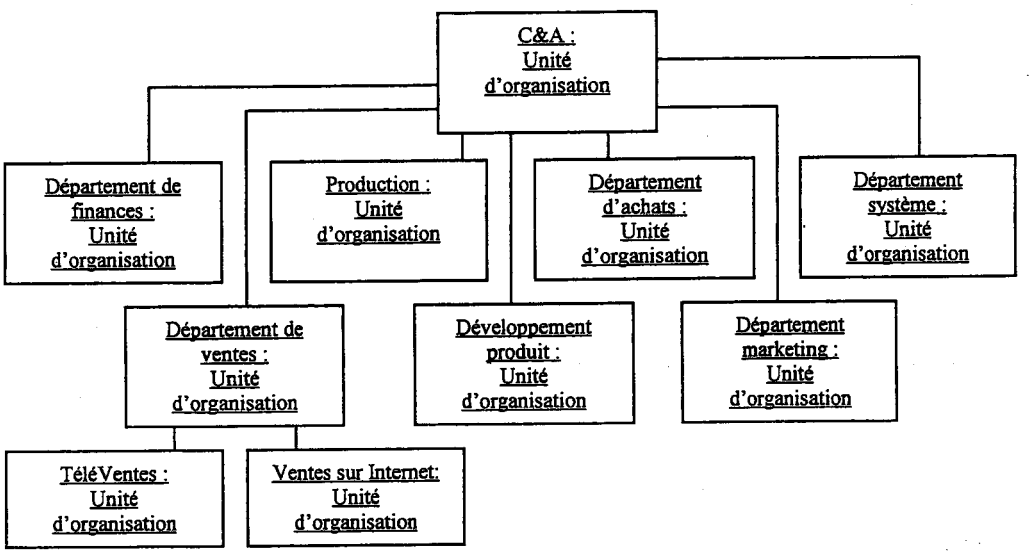

Figure 10 - Unités d'organisation de la compagnie C&A

III.2. LES PATRONS DES OBJECTIFS

Ces patrons ont été crées lors de la modélisation des objectifs. Un objectif est une issue critique ; un modèle d'objectif validé et vérifié supporte toutes les modélisations des autres travaux. Les modèles des objectifs affectent tout un processus de modélisation – comment le système est construit et comment il est utilisé.

Au cours des travaux de thèse, nous avons exploité deux processus d'objectifs qui nous ont permis de modéliser la décomposition des objectifs d'un système d'entreprise de ses domaines et de ses processus maîtres et métier (§ IV.3).

III.2.1. OBJECTIF METIER

a) Utilisation

Associer des objectifs pour spécifier les processus métier, les ressources et les règles, dans le but de faciliter la description et la validation de ceux-ci durant la modélisation métier.

b) Structure

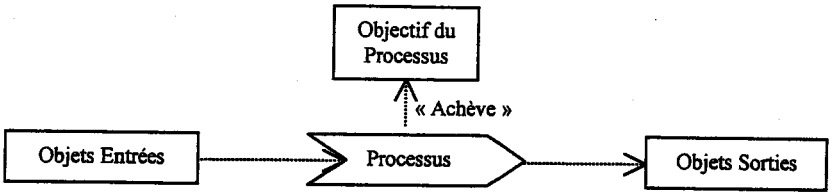

Figure 11 - Structure du patron Objectif Métier

Les Objets d'entrées sont raffinées et transformées en Objets Sorties, par le Processus, pour atteindre l'Objectif du Processus. Ce dernier décrit les résultats voulus (Objets de Sortie).

c) Exemple

Jim & Co. est une agence de divertissement, son objectif est de vendre et produire du matériel de divertissement jusqu'à 2005. Elle a plusieurs processus métier de ventes, de marketing, de gestion, etc.

Le processus de ventes reçoit des prospectus comme ordres d'entrées. L'agence gère les processus de ventes en tenant compte de ventes, définissant des directives de ventes, et en établissant des objectifs claires pour ces processus.

L'objectif financier pour l'année 1999 était d'atteindre un budget de vente de 250.000.000 $. D'autre part, il faut que l'agence améliore son chiffre d'affaire. Or, les clients non satisfaits affecteront négativement les futures ventes.

Pour atteindre l'objectif, les processus de ventes doivent donner un résultat pour satisfaire les clients et atteindre le budget de ventes voulu. Mais, ces deux derniers peuvent créer des conflits ; vendre et livrer des produits sans pouvoir satisfaire les clients ou inversement.

Dans plusieurs Entreprises les objectifs peuvent être contradictoires par nature, mais il est préférable de ne par éliminer les objectifs contradictoires et d'en tenir compte même partiellement.

La Figure 12 illustre le processus de ventes de l'agence. L'objectif du processus est l'objectif de vente quantitatif avec un budget de vente, unité monétaire, budget de l'année etc.

L'objectif de l'objet de sortie (ordre) est qualitatif (satisfaire le client).

Le processus est exécuté par les matériaux et le personnel des ventes.

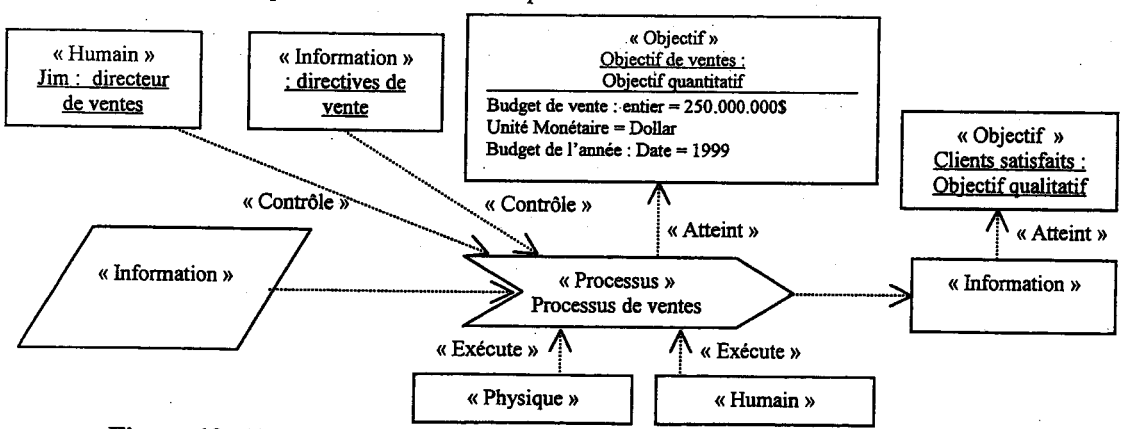

Figure 12 - Exemple d'objectifs du processus de ventes chez l'agence Jim & Co.

III.2.2. DECOMPOSITION D'UN OBJECTIF METIER

a) Utilisation

Diviser un objectif en sous-objectifs plus concrets et plus faciles à achever, dans le but d'atteindre l'objectif parent.

b) Structure

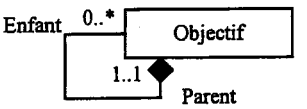

Figure 13 - Structure du patron Décomposition d'un Objectif Métier

L'objectif est de haut niveau d'abstraction pourrant être décomposé en sous-objectifs, que eux même peuvent être divisés.

c) Exemple

Attirer plus de clients est l'objectif de la compagnie de Business Internet, Inc.

Son objectif qualitatif est d'atteindre un nombre de clients équivalent à 500.000.

Cet objectif est décomposé en plusieurs sous-objectifs, nous citons uniquement deux qui décrivent :

- Plusieurs visiteurs d'Internet : les noms connus.
- Clients inscrits : noms et adresses des clients inscrits.

Ces objectifs mènent à l'objectif de 500.000 clients inscrits.

La Figure 14 montre l'objectif principal avec ses sous-objectifs qui sont eux aussi décomposés en sous-objectifs.

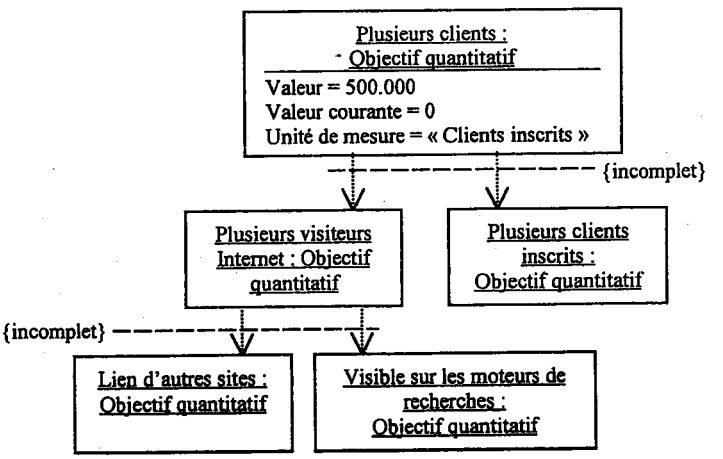

Figure 14 - Exemple de décomposition d'objectifs chez Business Internet, Inc.

III.3. LES PATRONS DES PROCESSUS

Ces patrons sont des patrons de comportement et de fonctionnement. Leurs intentions est d'améliorer la qualité dans les modèles Workflow et dans d'autres modèles orientés processus. Les modèles des processus

se réfèrent aux ressources, et sont contraints par des règles, dans le but de satisfaire les objectifs des processus. On peut dire que les processus sont décrits par la façon d'achever les objectifs spécifiques pour un ensemble de ressources et règles prédéfinies ; où les règles expriment les états possibles des ressources, et les objectifs expriment les états désirés des ressources.

III.3.1. STRUCTURE DE BASE D'UN PROCESSUS

a) Utilisation

Montre comment former le concept processus métier en termes de ressources métier, objectifs de processus, et les transformations ou raffinements des entrées objets en sorties objets.

b) Structure

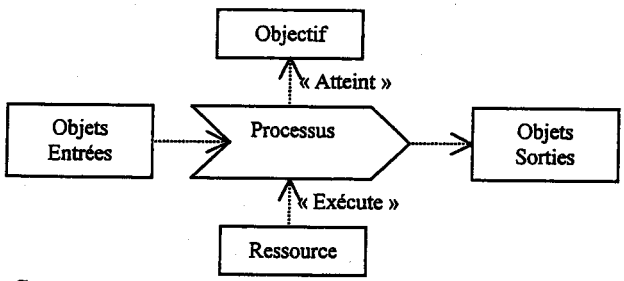

Figure 15 - Structure du patron 'Structure de Base d'un processus'

- Processus : classe identifiant les processus métier,
- Les Objets d'entrées sont raffinées et transformées en Objets Sorties par le processus métier,
- Objectif : objectif du processus métier,
- Ressource : la ou les ressources utilisées dans le processus pour exécuter une ou plusieurs étapes de celui-ci.

c) Exemple

Software Inc. développe des logiciels sur des contrats, mais la compagnie avait des problèmes : satisfaire les exigences de ses clients et des problèmes avec la planification interne des travaux.

Un planning pauvre d'un projet a mené à l'achat d'outils qui ne sont pas nécessaires au projet, à surcharger les développeurs sans arriver à satisfaire le client. Alors, la compagnie a décidé d'améliorer son projet de planification en utilisant ce patron, pour contrôler les objectifs du projet, les produits à produire, et les développeurs et les outils nécessaires. En décrivant leur processus de développement de logiciels, et son objectif (satisfaire le client), il devient plus facile de définir les séquences de travail, les activités à accomplir et leurs objectifs spécifiques.

En définissant que chaque processus doit commencer par une spécification et se termine par un logiciel et sa documentation.

Par conséquent, les membres et les chefs du projet peuvent connaître les règles de base et comment le projet commence et se termine, dans le but de coordonner les outils et les développeurs.

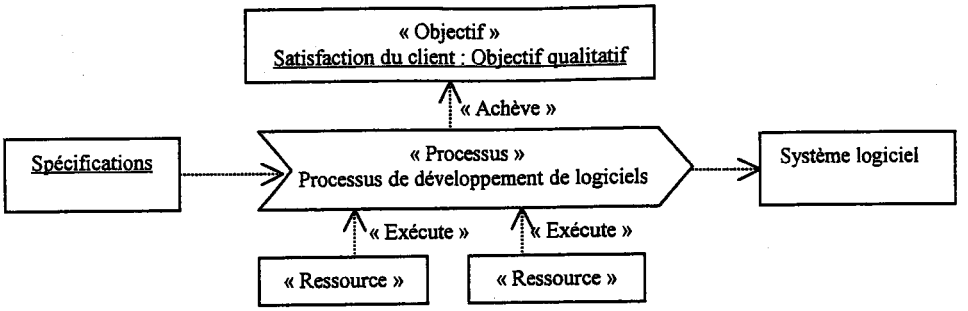

Figure 16 - Exemple d'application du patron 'Structure de base d'un patron'

La Figure 16 représente le processus de développement de logiciels en définissant l'objectif qualitatif (satisfaction du client) qu'il doit atteindre, lorsque qu'il transforme les spécifications (entrées) en un système logiciel (sorties). Pour ce faire, il a besoin de développeurs et d'outils de développement de logiciels (ressources).

III.3.2. NIVEAU DE CONTROLE DE PROCESSUS

a) Utilisation

Aide à structurer les systèmes complexes dans le but de les comprendre ou les réorganiser.

Le principe fondamental est de décomposer tous les systèmes en niveaux de processus où chaque processus contrôle celui du niveau le plus bas.

b) Structure

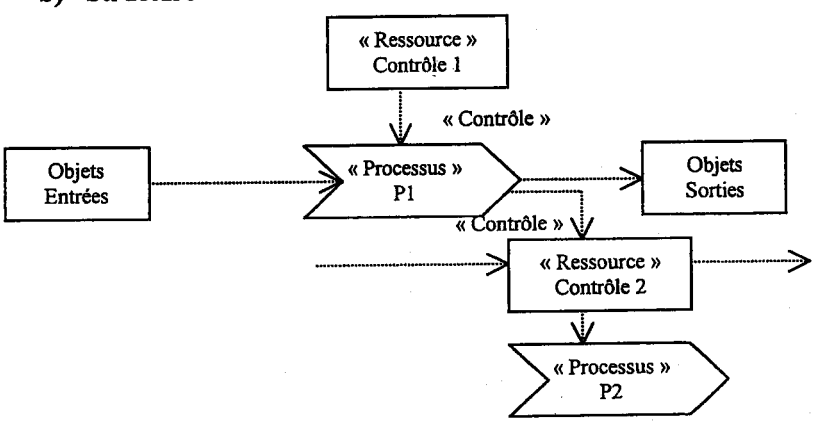

Figure 17 - Structure du patron 'Niveau de Contrôle de Processus'

- Processus P1, P2: classes identifiant les processus métier,
- Les Objets d'entrées sont raffinées et transformées en Objets Sorties par le processus P1,
- Objectif : objectif du processus métier,
- Ressource Controle1, Contrôle 2 : ce sont des ressources contrôles pour envoyer des directives et des contrôles, respectivement pour le processus P1 au processus P2.

c) Exemple

Un travail peut être contrôlé sur plusieurs niveaux. Le processus de développement du marché est le niveau le plus haut des processus qui a l'analyse du marché en entrée, et livre un plan du marché, en sortie. Ce dernier contrôle le processus de développement du travail, qui produit des directives pour les processus de gestion, de développement du produit, et de développement de la production, en terme de plan de travail.

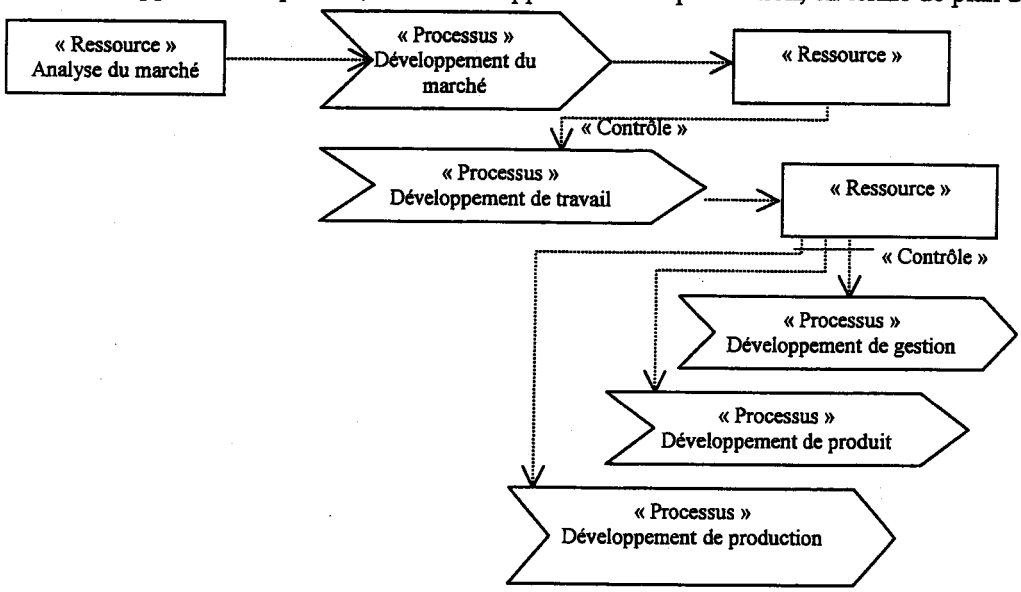

Figure 18 - Exemple de niveaux de contrôles de processus

La Figure 18 montre les processus de développement du marché, du travail, du produit et de production.

Le plan de marketing et le plan de travail sont respectivement des produits livrés pour les processus de développement du marché et de développement de travail.

III.3.3. INSTANCE DE PROCESSUS

a) Utilisation

Distinguer les processus de leurs instances.

b) Structure

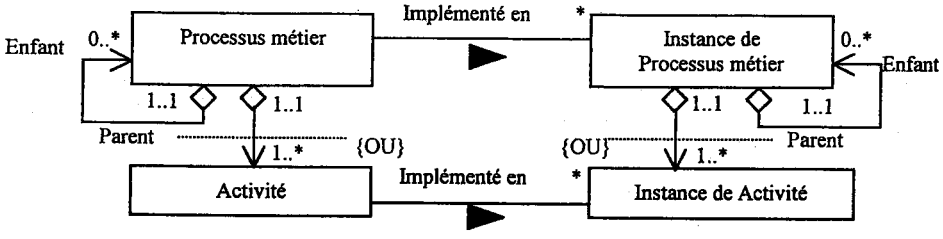

Figure 19 - Structure du patron 'Instance de processus'

c) exemple

Un processus de développement de softwares peut être documenté dans un livre -par exemple, sur Internet. Une personne peut lire une des copies décrivant le processus et le suit étape par étape pour créer une application ou un système.

Une semaine plus tard, une autre personne lit une des copies décrivant le même processus et commence une autre application du même processus.

Maintenant, il y a deux personnes qui ont suivis le même processus, mais en phases différentes de développement.

La première personne doit avoir déjà formulé ses besoins du système et a commencé les analyses, lorsque la deuxième personne a uniquement commencé la phase des besoins. Ainsi, un processus peut progresser en différentes exécutions dans de différentes phases. Cela montre qu'un processus existe en plusieurs instances.

Et puisque un processus est composé d'activités, alors les instances du processus seront composées d'instances de ces activités.

III.3.4. ETAT D'INSTANCE DE PROCESSUS

a) Utilisation

Permet de définir les états d'une instance de processus, pour mieux comprendre son comportement avant de développer des systèmes supports.

b) Structure

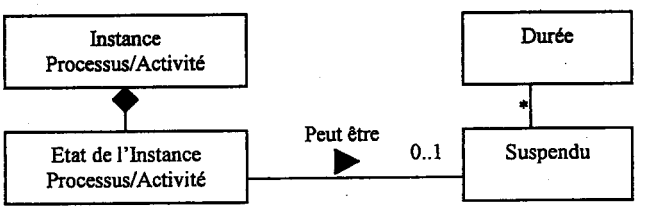

Figure 20 - Structure du patron 'Etat d'Instance d'un Processus'

- Instance de processus : représente l'exécution actuelle d'un processus et ses activités,
- Etat de l'instance de processus : représente l'état d'une instance de processus,
- Suspendu : signifie qu'une instance de processus est temporairement arrêtée, pendant une durée 'Durée'.

c) Exemple

La compagnie de consultation A&A développe des produits pour l'industrie chimique.

Basée sur l'analyse du marché, elle contacte les clients pour essayer de vendre quelques nouvelles substances ou des équipements chimiques.

Plusieurs contrats ramènent à des négociations.

Le processus de développement de produit consiste à plusieurs phases. Chacune d'elle peut être suspendue pour une certaine durée. Par exemple, le développement d'une machine d'emballage de médicaments pour la boite Pharmatica, a été suspendu à cause du prix élevé de la machine. Après, la boite revient avec un partenaire pour payer la machine. Cela redémarre le développement de la machine.

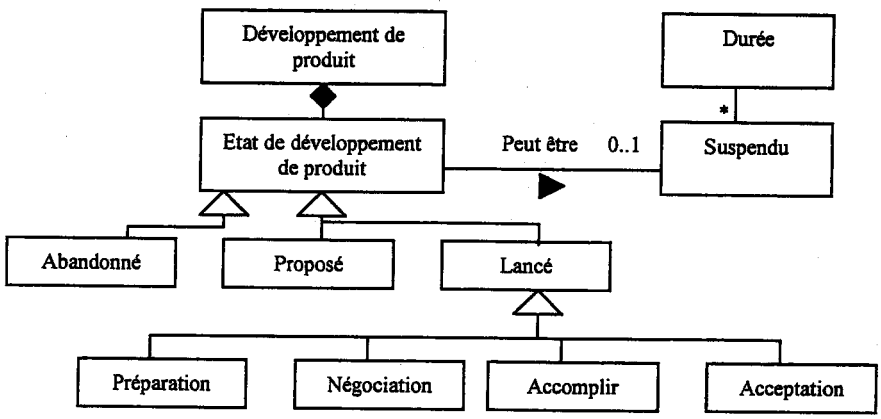

Figure 21 - Diagramme de classes

Cet exemple est représenté par le diagramme de classes (Figure 21).

Ce dernier spécifie les différents états que le processus de développement de produit peut avoir (abandonné, proposé, et lancé). Quand le processus est lancé, il peut être accompli, accepté, en cours de préparation ou de négociation.

Le processus peut être suspendu pour une certaine durée (suspension du processus de développement de la machine d'emballage).

ANNEXE C

UNIFIED MODELING LANGUAGE

« UML »

L'objectif de cette annexe consiste à permettre de lire les divers diagrammes UML qui ont été réalisés dans ce manuscrit. Notamment, nous détaillons le diagramme de cas d'utilisation, les paquetages, le diagramme de classes, le diagramme de collaboration et de séquences, le diagramme d'activités et le diagramme de composants.

I. HISTORIQUE

UML est un langage de modélisation qui permet de développer des modèles orientés objet au niveau métier ou au niveau conception des systèmes d'information.

Il est le fruit de la fusion des méthodes orientés objet dominantes comme Booch, Fusion, OMT, et OOSE.

Il a été adopté par OMG[1] et devenu une norme en 1997.

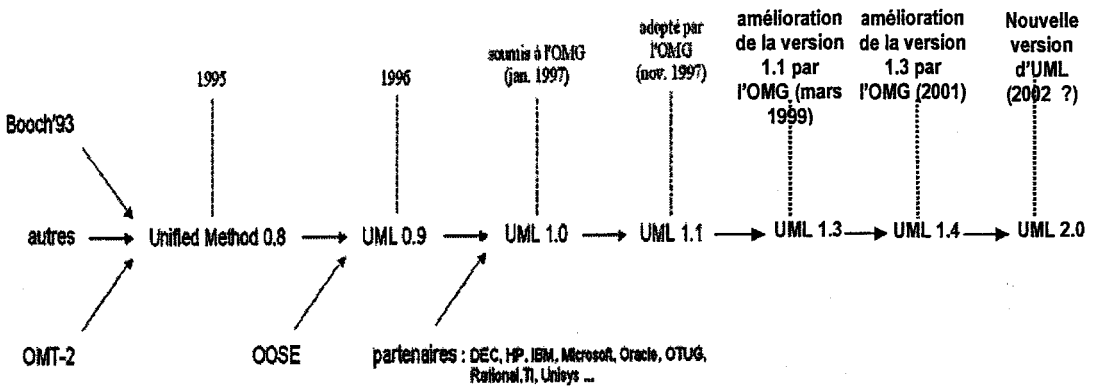

Figure 22 – Historique d'UML

Aujourd'hui, UML est un standard incontournable, il est

- Le résultat d'un large consensus (industriels, méthodologistes...).
- Le fruit d'un travail d'experts reconnus.
- Issu du terrain.
- Riche (il couvre toutes les phases d'un cycle de développement).
- Ouvert (il est indépendant du domaine d'application et des langages d'implémentation *langages de programmation*).

Après l'unification et la standardisation, et l'industrialisation d'UML : les outils qui supportent UML se multiplient (GDPro, ObjectTeam, Objecteering, OpenTool, Rational Rose, Rhapsody, STP, Visio, Visual Modeler, WithClass...).

II. MODÉLISER AVEC UML

II.1. QU'EST-CE QU'UN MODELE ?

Un modèle est une abstraction de la réalité. L'abstraction est l'un des piliers de l'approche objet.
Un modèle est une vue subjective mais pertinente de la réalité :

[1] Object Management Group (www.omg.org), organisme de normalisation qui a pour but de promouvoir les technologies objets.

- Un modèle définit une frontière entre la réalité et la perspective de l'observateur. Ce n'est pas "la réalité", mais une vue très subjective de la réalité.
- Bien qu'un modèle ne représente pas une réalité absolue, un modèle reflète des aspects importants de la réalité, il en donne donc une vue juste et pertinente.

Le caractère abstrait d'un modèle doit notamment permettre :

- De faciliter la compréhension du système étudié
- De simuler le système étudié : Un modèle représente le système étudié et reproduit ses comportements.

II.2. COMMENT MODELISER AVEC UML ?

UML est un langage qui permet de représenter des modèles, mais il ne définit pas le processus d'élaboration des modèles.

Cependant, dans le cadre de la modélisation d'une application informatique, les auteurs d'UML préconisent d'utiliser une démarche :

- Itérative et incrémentale,
- Guidée par les besoins des utilisateurs du système,
- Centrée sur l'architecture logicielle.

II.2.1. UNE DEMARCHE ITERATIVE ET INCREMENTALE ?

L'idée est simple : pour modéliser (comprendre et représenter) un système complexe, il vaut mieux s'y prendre en plusieurs fois, en affinant son analyse par étapes.
Cette démarche devrait aussi s'appliquer au cycle de développement dans son ensemble, en favorisant le prototypage.

Le but est de mieux maîtriser la part d'inconnu et d'incertitudes qui caractérisent les systèmes complexes.

II.2.2. UNE DEMARCHE PILOTEE PAR LES BESOINS DES UTILISATEURS ?

Avec UML, ce sont les utilisateurs qui guident la définition des modèles :

- Le périmètre du système à modéliser est défini par les besoins des utilisateurs (les utilisateurs définissent ce que doit être le système).
- Le but du système à modéliser est de répondre aux besoins de ses utilisateurs (les utilisateurs sont les clients du système).

Les besoins des utilisateurs servent aussi de fil rouge, tout au long du cycle de développement (itératif et incrémental) :

- A chaque itération de la phase d'analyse, on clarifie, affine et valide les besoins des utilisateurs.
- A chaque itération de la phase de conception et de réalisation, on veille à la prise en compte des besoins des utilisateurs.
- A chaque itération de la phase de test, on vérifie que les besoins des utilisateurs sont satisfaits.

II.2.3. UNE DEMARCHE CENTREE SUR L'ARCHITECTURE ?

Kruchten propose différentes perspectives, indépendantes et complémentaires, qui permettent de définir un modèle d'architecture (publication IEEE, 1995).

Cette vue ("4+1") a fortement inspiré UML :

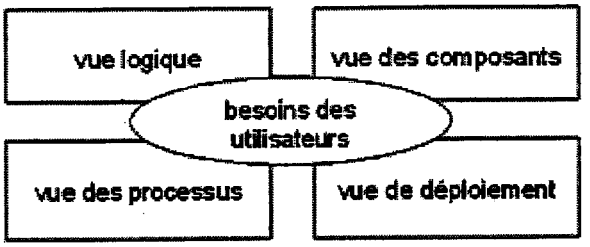

Figure 23 – Architecture d'UML

Elle se compose de cinq vues complémentaires :

a) La vue logique

- Cette vue de haut niveau se concentre sur l'abstraction et l'encapsulation, elle modélise les éléments et mécanismes principaux du système.
- Elle identifie les éléments du domaine, ainsi que les relations et interactions entre ces éléments :
- Les éléments du domaine sont liés au(x) métier(s) de l'entreprise,
  - Ils sont indispensables à la mission du système,
  - Ils gagnent à être réutilisés (ils représentent un savoir-faire).
- Cette vue organise aussi (selon des critères purement logiques), les éléments du domaine en "catégories" :
  - Pour répartir les tâches dans les équipes,
  - Regrouper ce qui peut être générique,
  - Isoler ce qui est propre à une version donnée, etc...

b) La vue des composants

Cette vue de bas niveau (aussi appelée "vue de réalisation"), montre :

- L'allocation des éléments de modélisation dans des modules (fichiers sources, bibliothèques dynamiques, bases de données, exécutables, etc...).
- En d'autres termes, cette vue identifie les modules qui réalisent (physiquement) les classes de la vue logique.
- L'organisation des composants, c'est-à-dire la distribution du code en gestion de configuration, les dépendances entre les composants...
- Les contraintes de développement (bibliothèques externes...).
- La vue des composants montre aussi l'organisation des modules en "sous-systèmes", les interfaces des sous-systèmes et leurs dépendances (avec d'autres sous-systèmes ou modules).

c) La vue des processus

Cette vue est très importante dans les environnements multitâches ; elle montre :

- La décomposition du système en terme de processus (tâches).
- Les interactions entre les processus (leur communication).
- La synchronisation et la communication des activités parallèles (threads).

d) La vue de déploiement

Cette vue très importante dans les environnements distribués, décrit les ressources matérielles et la répartition du logiciel dans ces ressources :

- La disposition et nature physique des matériels, ainsi que leurs performances.
- L'implantation des modules principaux sur les noeuds du réseau.

- Les exigences en terme de performances (temps de réponse, tolérance aux fautes et pannes...).

e) La vue des besoins des utilisateurs

Cette vue (dont le nom exact est "vue des cas d'utilisation"), guide toutes les autres.

- Dessiner le plan (l'architecture) d'un système informatique n'est pas suffisant, il faut le justifier.
- Cette vue définit les besoins des clients du système et centre la définition de l'architecture du système sur la satisfaction (la réalisation) de ces besoins.
- A l'aide de scénarios et de cas d'utilisation, cette vue conduit à la définition d'un modèle d'architecture pertinent et cohérent.
- Cette vue est la "colle" qui unifie les quatre autres vues de l'architecture.
- Elle motive les choix, permet d'identifier les interfaces critiques et force à se concentrer sur les problèmes importants.

III. MODELE CONCEPTUEL D'UML

Pour comprendre UML, il est nécessaire de s'appuyer sur son modèle conceptuel, ce qui implique l'assimilation de trois éléments essentiels [Booch, 00].

- les briques de base d'UML,
- les règles qui déterminent la manière de les assembler,
- et quelques mécanismes généraux qui s'appliquent à UML dans son ensemble.

Dans cet annexe, nous nous limitons à présenter les briques de bases d'UML.

III.1. LES BRIQUES DE BASE D'UML

La terminologie d'UML inclut trois sortes de briques:

- des éléments,
- des relations,
- des diagrammes.

Les éléments sont les abstractions essentielles à un modèle. Les relations constituent les liens entre ces éléments et les diagrammes les regroupent en ensembles dignes d'intérêt.

III.1.1. DIAGRAMMES DANS UML

Un *diagramme* est la représentation graphique d'un ensemble d'éléments qui constituent un système. La plupart du temps, il se présente sous la forme d'un graphe connexe où les sommets correspondent aux éléments et les arcs aux relations. Les diagrammes servent à visualiser un système sous différentes perspectives et sont donc des projections dans un système. Pour les systèmes complexes, un diagramme ne représente qu'une vue partielle des éléments qui composent ces systèmes. Un même élément peut apparaître sur tous les diagrammes, sur quelques diagrammes seulement (cas le plus fréquent) mais peut aussi n'apparaître sur aucun diagramme (ce qui reste très rare). En théorie, un diagramme peut contenir n'importe quelle combinaison d'éléments et de relations. Cependant, en pratique, seul un petit nombre de combinaisons fréquentes se répètent. Ces combinaisons correspondent aux cinq vues les plus utiles, qui englobent toute l'architecture d'un système à forte composante logicielle.

UML comprend deux types de diagrammes :

a) Diagrammes statiques

Les **diagrammes de cas d'utilisation** représentent un ensemble de cas d'utilisation et d'acteurs (sortes de classes particulières) et leurs relations. Ils présentent la vue statique des cas d'utilisation d'un système et sont particulièrement importants dans l'organisation et la modélisation des comportements d'un système.

La Figure 24 illustre un exemple de diagramme de cas d'utilisation où nous représentons deux acteurs qui interviennent au système de gestion des articles et les cas d'utilisation qui lui sont associés. Par exemple, le responsable des stocks peut stocker et déstocker des articles et consulter leur prix.

Les cas d'utilisation peuvent être reliés entre eux par :

- Relations de généralisation qui permettent de définir un cas d'utilisation comme la génération d'autres cas (Figure 25).
- Relations d'extension qui définissent un cas d'utilisation comme l'extension d'un autre cas. Par exemple, le stockage d'un article en rupture de stock est cas exceptionnel du stockage d'un article (Figure 25)
- Relations d'inclusion qui permettent à un cas d'utilisation de faire appel aux services d'un autre cas. Cette relation est une factorisation d'une fonctionnalité dans plusieurs cas d'utilisation. Par exemple, l'identification d'un article peut être appelée par la consultation des articles (Figure 25).

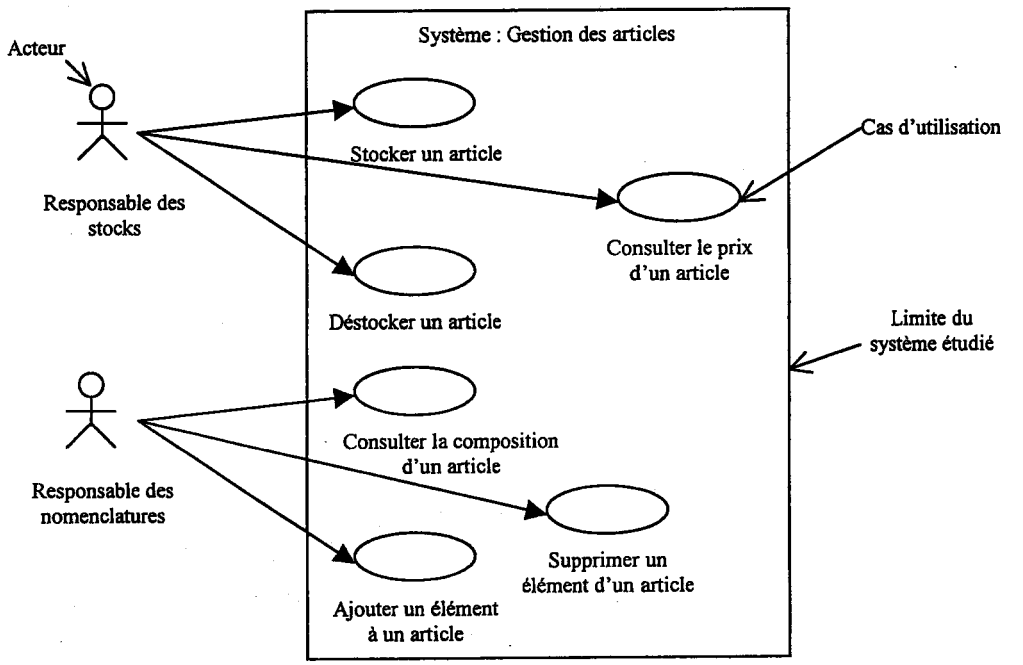

Figure 24 – Diagramme des cas d'utilisation

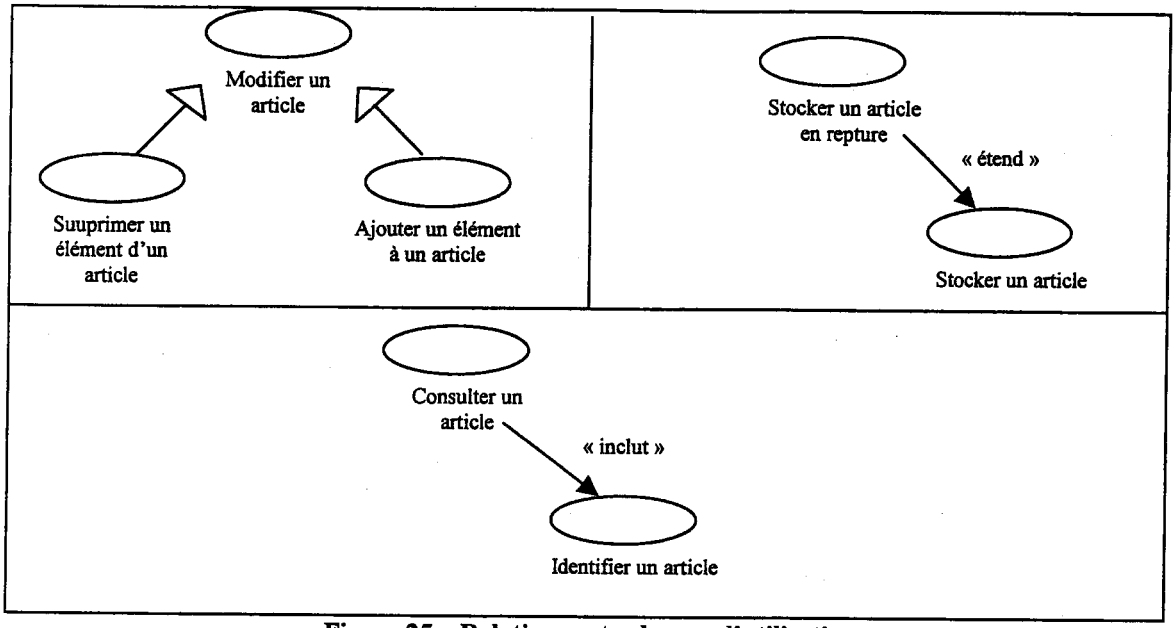

Figure 25 – Relations entre les cas d'utilisation

Le **paquetage** est une technique qui permet de mettre en œuvre le partitionnement des modèles tout en préservant la cohérence de l'ensemble. Ceci est pour découper des modèles de taille importante sans négliger la mise en évidence des liens qui existent entre ces sous-ensembles, les dépendances.

Généralement, la technique de paquetage peut être utilisée au niveau des diagrammes de cas d'utilisation ou bien dans les diagrammes de classes.

La Figure 26 montre un exemple de deux paquetages liés entre eux et qui rassembles plusieurs classes d'objets.

Ce lien est spécifié dans la Figure 27 par une association entre les deux classes « Produit » et « Fourniseeur » appartenant respectivement aux paquetages « Commandes » et « Fournisseurs ».

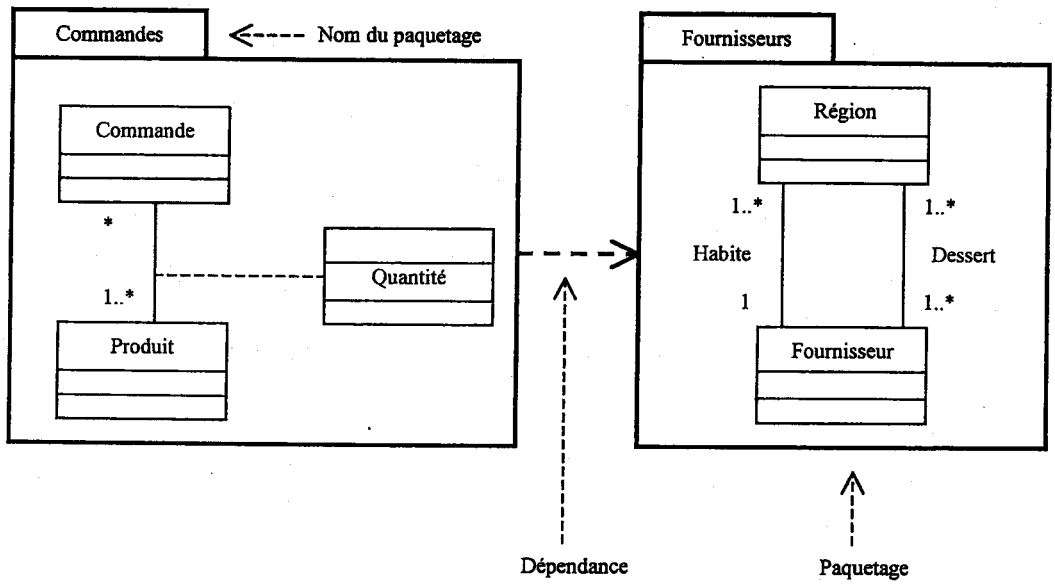

Figure 26 - Paquetages

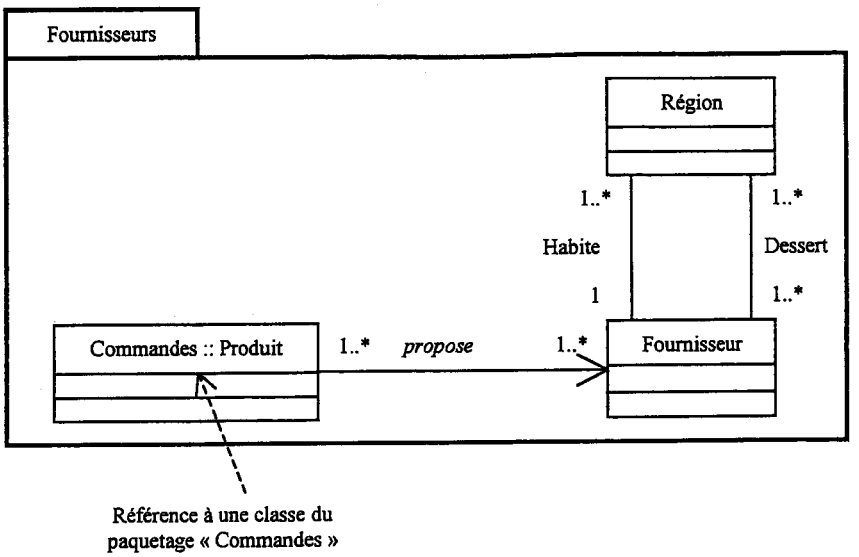

Figure 27 – Référence d'un élément d'au autre paquetage

Les **diagrammes de classes** représentent un ensemble de classes, d'interfaces et de collaborations, ainsi que leurs relations. Ce sont les diagrammes les plus fréquents dans la modélisation des systèmes orientés objet. Ils présentent la vue de conception statique d'un système.

Chaque classe admet un nom, des attributs (propriétés) et des méthodes (fonctions/opérations) qu'elle peut accomplir.

La Figure 28 montre un exemple de diagramme de classes qui contient les classes Article, Entreprise, Pièce, Piston, Soupape et Bielle. Les deux premières sont reliées entre elle par une association à double sens et qui signifie qu'un ou plusieurs (1..*) articles sont fournis par une ou plusieurs (1..*) entreprises ayant pour rôle « fournisseur ». Cette association peut admettre un attribut (propriété) qui indique le prix d'un article fournis par une entreprise. Cet attribut est représenté par une classe sans nom (Figure 28).

Un article peut être agrégé, spécifiquement composé de plusieurs composants. Cette composition est représentée par la relation « composition » où un article joue le rôle d'un composite composé d'autres articles qui sont des composant.

Un article peut être aussi spécialisé en pièce telles qu'un piston, une bielle ou une soupape. Dans ce cas, la classe « Pièce » hérite des attributs et des méthodes de la classe « Article » et de même pour les classes « Piston », « Soupape » et « Bielle » par rapport à la classe « Pièce ».

Le sens inverse de la spécialisation est la généralisation.

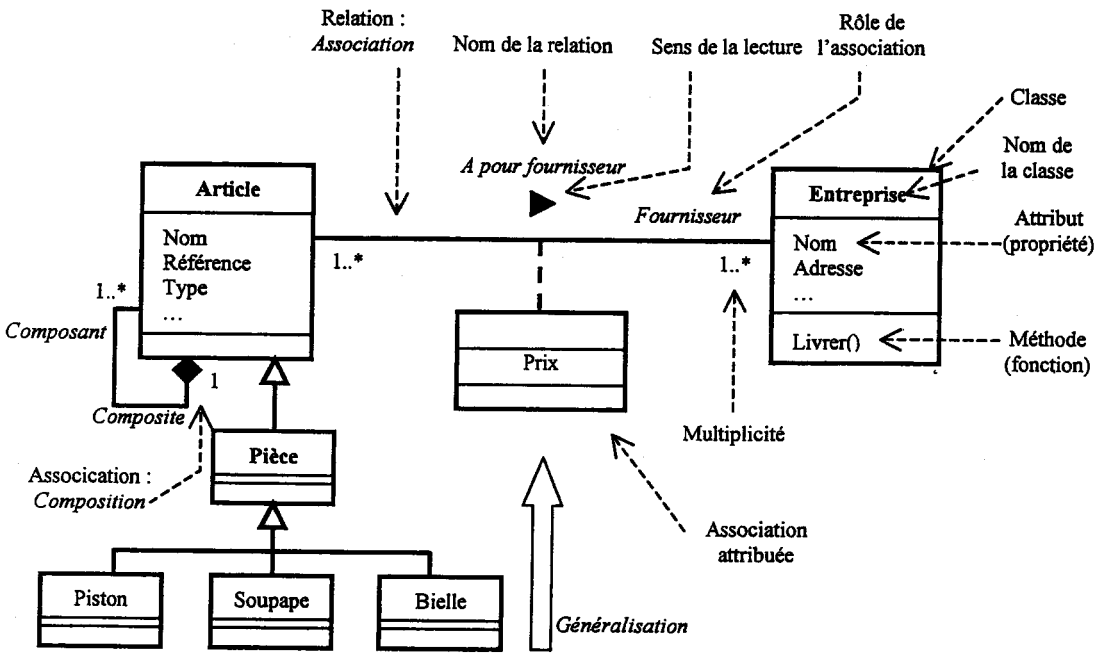

Figure 28 – Un exemple de diagramme de classes

Dans une association, chaque rôle porte une multiplicité qui indique les nombres entiers minimaux et maximaux d'occurrences de la classe jouant le rôle et pouvant être reliées aux occurrences de la classe associée.

Le tableau suivant donne les multiplicités utilisées dans UML :

Multiplicité	**Intervalle en mathématiques** $\in \aleph$
1..1	[1,1]
1	[1,1]
0..1	[0,1]
m..n	[m,n]
*	[0, +∞]
0..*	[0, +∞]
1..*	[1, +∞]

Les **diagrammes de composants** représentent l'organisation et les dépendances au sein d'un ensemble de composants. Ils présentent la vue d'implémentation statique d'un système et sont liés aux diagrammes de

classe dans le sens où un composant correspond généralement à une ou plusieurs classes, interfaces ou collaborations.

La Figure 29 montre les deux composants « Fournisseurs » et « Articles » qui implémentent chacun les classes qui lui correspondent et leurs comportements respectifs.

Chaque composant communique avec son environnement externe par ses interfaces comme c'est le cas pour les deux composants de la Figure 29 où ils communiquent ensemble via leurs interfaces « IGestionArticles » et « IGestionFournisseurs ».

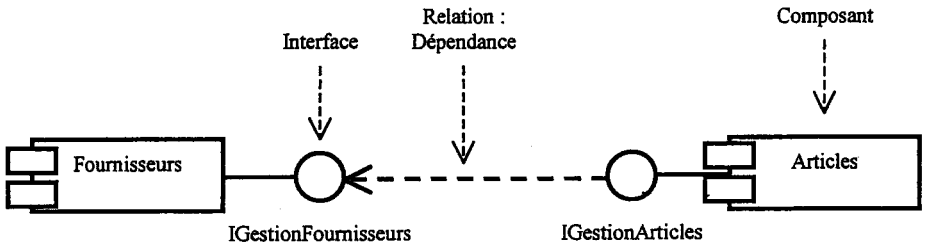

Figure 29 – Diagramme de composants

b) Diagrammes dynamiques

Les **diagrammes de séquence** et les **diagrammes de collaboration** sont deux types de diagrammes d'interaction. Les *diagrammes d'interaction* représentent une interaction, c'est-à-dire un ensemble d'objets et leurs relations, y compris les messages qu'ils peuvent s'échanger. Ils présentent la vue dynamique d'un système. Les *diagrammes de séquence* sont des diagrammes d'interaction qui mettent l'accent sur le classement chronologique des messages alors que les *diagrammes de collaboration* sont des diagrammes d'interaction qui mettent l'accent sur l'organisation structurelle des objets qui envoient et reçoivent des messages. Les diagrammes de séquence et les diagrammes de collaboration sont isomorphes, c'est-à-dire que l'un peut être transformé en l'autre.

Dans la Figure 30 un diagramme de collaboration montre les interactions entre plusieurs objets ou instances de classes. Chaque interaction représente un ou plusieurs échange de messages dans un ou les deux sens numérotés dans l'ordre d'exécution.

Par exemple, un client demande un devis, un commercial recherche le prix de base du produit et les ristournes accordées et communique le devis au client.

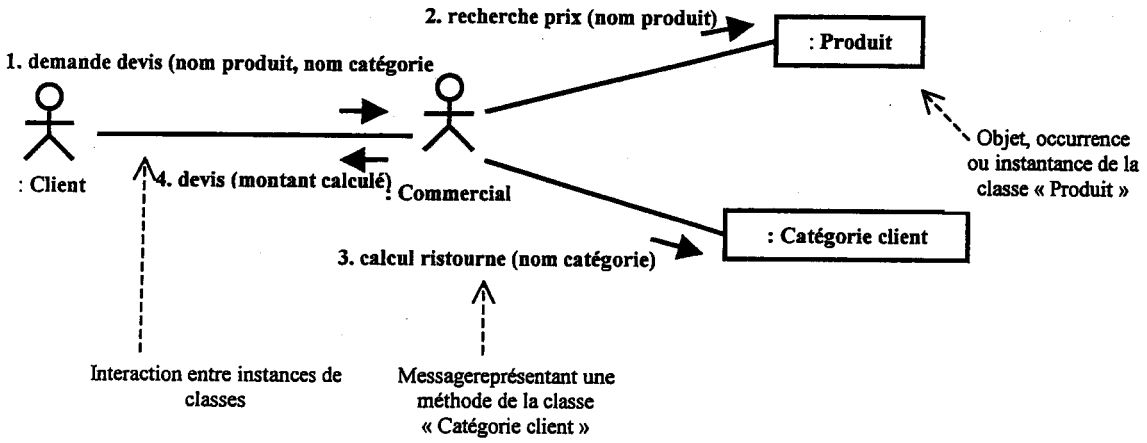

Figure 30 – Diagramme de collaboration

Un message peut être complété par un ou plusieurs *arguments*. Le message 1, Demande de devis, a pour arguments l'intitulé du produit souhaité et la catégorie du client. Le message 2, Calculer le prix, a pour argument l'intitulé du produit. Le message 3, Calculer la ristourne, a pour argument la catégorie du client.

Le diagramme de collaboration de la Figure 30 peut être représenté différemment en se basant sur un classement chronologique. En effet, la Figure 31 représente ce classement par un diagramme de séquences.

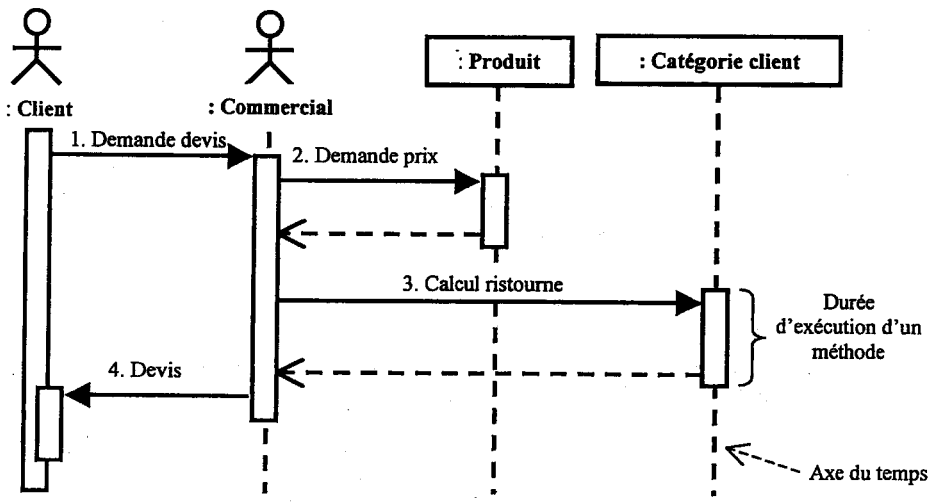

Figure 31 – Diagramme de séquences

Les **diagrammes d'activités** sont un type particulier de *diagramme d'états-transitions* qui décrit la succession des activités au sein d'un système. Ils présentent la vue dynamique d'un système; ils sont particulièrement importants dans la modélisation de la fonction d'un système et mettent l'accent sur le *flux de contrôle* entre les objets. La participation de ces objets à des traitements représente un *flux d'objet*.

L'enchaînement des activités peut être soumis à des *branchements* ou à des *synchronisations*. La visualisation de *couloirs d'activités* permet de représenter la répartition de la responsabilité des activités entre différents acteurs.

Dans la Figure 32, la transition entre les deux activités « Préparation de la commande » et « Envoi de la commande » relève du flux de contrôle. En revanche, les trois transitions, indiquées en pointillés, qui mettent en jeu un objet « Commande » dans différents états, font partie du flux d'objet. Les transitions du flux d'objet peuvent également comporter des conditions de garde.

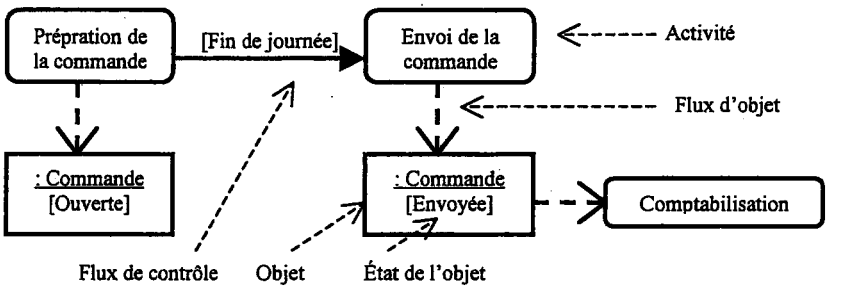

Figure 32 – Diagramme d'activités

Un flux de contrôle peut comprendre des chemins alternatifs. Chaque branche est soumise à une condition, qui est une condition de garde comme le montre l'exemple de la commande électronique sur un site de e-commerce (Figure 33).

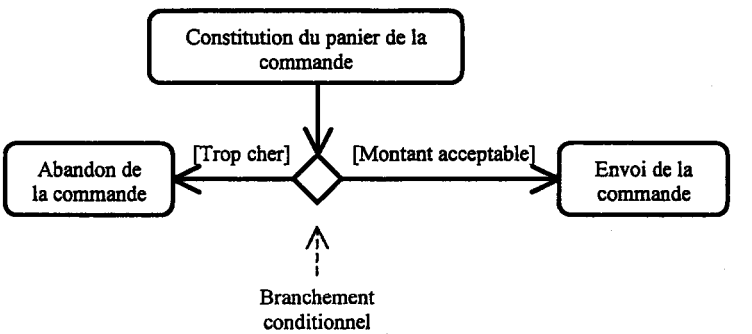

Figure 33 – Branchement conditionnel

Parfois, le flux de contrôle peut suivre deux chemins parallèles: c'est l'ouverture d'une fourche. Ensuite, les deux chemins se rejoignent dans une fermeture de synchronisation. En principe, l'ouverture et la fermeture doivent se répondre. La Figure 34 représente, dans le cas d'une demande de crédit bancaire en ligne, d'une part la recherche de produits adéquats, d'autre part l'évaluation du risque présenté par le client, ce qui permet de proposer une réponse.

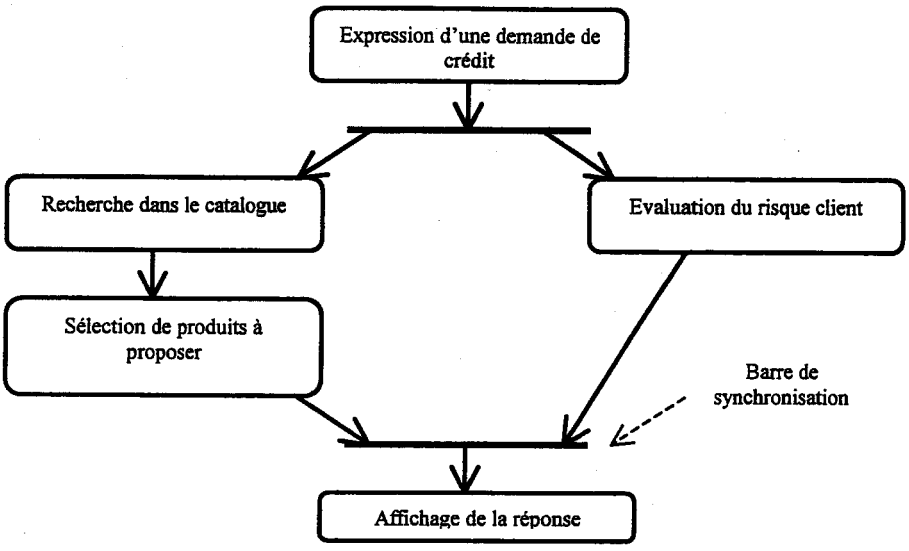

Figure 34 - Synchronisation

Cette liste n'est pas exhaustive. Des outils peuvent utiliser UML pour produire d'autres sortes de diagrammes, même si les neuf sortes précédemment citées sont les plus fréquemment utilisées.

UML offre aussi des mécanismes d'extension qui lui permettent de définir de nouveaux éléments et relations de manière à étendre et à adapter le vocabulaire de modélisation. Ces mécanismes sont : les stéréotypes, les étiquettes et les contraintes. Les stéréotypes permettent en particulier de définir de nouvelles classes d'éléments et de relations.

COMPOSANTS POUR LA MODELISATION DES PROCESSUS METIER EN PRODUCTIQUE, BASES SUR CIMOSA

Résumé

Ce travail de thèse est une partie du projet CAS (*CIMOSA Application Server*) qui consiste à mettre en place un système d'information (S.I.) utile à la conception de systèmes et d'applications d'entreprises. Au cœur de ce S.I. se trouve un référentiel commun pour une chaîne d'entreprise.

En effet, La thèse est focalisée sur la conception de ce référentiel qui permet de modéliser les systèmes d'entreprise à travers leurs processus métier. Ceci doit être réalisé en respectant les objectifs communs fixés dans la chaîne d'entreprise. Ainsi, les modèles résultants sont capitalisés dans une bibliothèque de systèmes propre à chaque entreprise, où ils sont partagés pour concevoir des applications manipulant les processus métier modélisés.

Le référentiel offre une méthodologie de modélisation intégrée dans un méta-modèle basé sur l'approche de CIMOSA.

L'implémentation physique de ce méta-modèle a été réalisée par des composants «autonomes» et réutilisables que nous avons appelé : Composants de Modélisation en Entreprise (*EMC «Enterprise Modeling Components»*).

La conception du méta-modèle et des composants est formalisée dans le langage unifié de modélisation UML (*Unified Modeling Language*) pour faciliter des améliorations ultérieures et des travaux développement informatique.

Afin d'illustrer notre propos, une maquette logicielle d'un framework a été développée en se référant aux composants EMC du référentiel pour automatiser la modélisation des systèmes d'entreprise. La capitalisation des modèles réalisés est assurée par le module S.I.R.S. (Système Interactif de Recherche de Séquences) qui fonctionne sous DOS.

La maquette a été développée en orientée objet avec le langage Visual Basic.

Mots clefs : *Modélisation, Projet CAS, Système d'entreprise, Processus Métier, Référentiel, Méta-Modèle, CIMOSA, Composant, EMC, UML, Framework.*

CIMOSA-BASED COMPONENTS FOR BUSINESS PROCESSES MODELING IN MANUFACTURING

Abstract

This thesis work is a part of the project CAS (CIMOSA Application Server) which consists in setting up an information system (I.S.) useful for enterprise systems and applications design. The core of this I.S. is a common referential for a chain of enterprises.

Indeed, the work thesis was focusing on designing such referential, which allows to model enterprise systems through their business processes. This must be fulfilled by respecting the common objectives of the chain. Thus, the resulting models are capitalized in a systems repository of each enterprise, where they are shared to develop business process applications.

The referential offers a methodology of modeling integrated in a meta-model based on CIMOSA approach. The physical implementation of this meta-model was realized by "autonomous" and reusable components, which we called: "Enterprise Modeling Components" (E.M.C.).

The design of the meta-model and the components is formalized by the Unified Modeling Language (U.M.L.) to facilitate subsequent improvements and computing development.

To validate the approach, a software prototype of a framework has been developed while referring to the E.M.C. components. The framework objectives are automating the enterprise systems modeling. The models capitalization ensured by the module S.I.R.S. (*Système Interactif de Recherche de Séquences* "Interactive System for Sequences Search") which turns under DOS. The prototype has been developed with Visual BASIC language in object-oriented programming.

Keywords: *Modeling, Project CAS, Enterprise system, Business process, Referential, Meta-model, CIMOSA, Component, EMC, UML, Framework.*