MASTER EN SCIENCES INFORMATIQUES

Aide à la spécification des systèmes d'information

Leonard, P.

Award date:
1991

Awarding institution:
Universite de Namur

FACULTE NOTRE DAME DE
LA PAIX
Institut d'Informatique
Rue Grand Gagnage
Namur

AIDE A LA SPECIFICATION DES SYSTEMES D'INFORMATION

1991

Promoteur : F. Bodart

P. LEONARD

RESUME

Ce travail se situe dans le cadre des outils d'aide à la modélisation des système d'information (SI). La difficulté d'exploitation des méthodes de modélisation et la complexité croissante des besoins des entreprises requièrent une aide adéquate au projet et à l'organisation afin de supporter l'analyste dans son travail de modélisation.

La formalisation des méthodes de modélisation représente le premier pas nécessaire dans la création de cette aide. La manipulation des formalismes définis et la définition de règle de guidage en rapport avec la création de démarches personnelles dans un souci de complétude et de contrôle du travail d'analyse donnent le deuxième pas et la marche à suivre dans l'élaboration d'un outil d'aide.

Ce travail propose des concepts permettant la formalisation des méthodes tant au niveau de leur vocabulaire que de leur grammaire et permettant la définition de règles de guidage relativement aux particularités du projet, de l'organisation et à la personnalisation de la démarche d'analyse. Ces propositions ont été élaborées dans le cadre d'un prototype expérimental.

ABSTRACT

This study is part of the search for high quality tools related to an information system development environment. The difficulty to use correctly methodologies and the companies growing complex needs require an effective help to support the project analyst with his modelisation work.

The first step to achieve this help is to formalize the methodologies into accurate structures. The manipulation of the defined structures and the definition of guiding rules related to a personal process creation and linked with comprehensive specifications and an analysis control give the guiding steps to build a helping tool.

This study intends to propose concepts in view of formalizing methodologies on both levels, vocabulary and grammar, and to allow the definition of the guiding rules related to the specificity of the project and the personalization of the analysis process. These suggestions have been analysed from an already defined experimental prototype.

TABLE DES MATIERES

TABLE DES FIGURES

CHAPITRE I LES SYSTEMES D'INFORMATION

SECTION 1.1 INTRODUCTION

L'évolution inverse des coûts software et hardware apporte dans le monde informatique la nécessité de développer des systèmes d'information plus fiables, plus évolutifs et plus adéquats aux besoins des entreprises utilisatrices.

Cette nécessité associée à la complexité croissante des systèmes d'information réclame de nouveaux outils informatiques, notamment dans le domaine de la C.A.O. (Conception Assistée par Ordinateur). Notre travail s'inscrit dans ce créneau et se propose d'aborder plus précisément le domaine de l'aide à la spécification des systèmes d'information.

Nous ne proposons pas un outil en tant que tel et encore moins une méthode de spécification, notre objectif est simplement d'essayer de fixer les notions relatives à tout outil d'aide à la spécification, de poser les fondations nécessaires et d'établir un énoncé des différents objectifs et contraintes que doit remplir un tel outil. Dans ce travail, nous aborderons l'aide à la spécification des SI en tant que telle. Nous en définirons les objectifs et les enjeux pour aborder ensuite les différents concepts et principes de la modélisation d'un SI. Ces concepts et principes nous permettrons de dégager les éléments clés d'un système d'aide à la modélisation.

SECTION 1.2 CADRE GENERAL DE L'ETUDE

Section 1.2.1 La notion de système d'information

Les entreprises et les organisations connaissent, depuis longtemps, le rôle primordial joué par l'information dans le fonctionnement de leur société. Quelque soit son support, l'information s'inscrit directement soit en définition, soit en paramètre dans les contextes opératoires et décisionnels des processus de production.

Dans ce cadre, la manipulation de l'information et sa fluidité au sein de l'organisation doivent être d'une efficacité et d'une efficience maximale afin d'apporter à chacun le contexte informationnel nécessaire à l'exécution de sa mission.

La représentation schématique d'une organisation, montrée en Figure 1.1 et empruntée à Le Moigne [LEM77], nous présente l'organisation composée de trois systèmes : un système opérant (contexte opératoire), un système de décision (contexte décisionnel) et un système d'information (contexte informationnel).

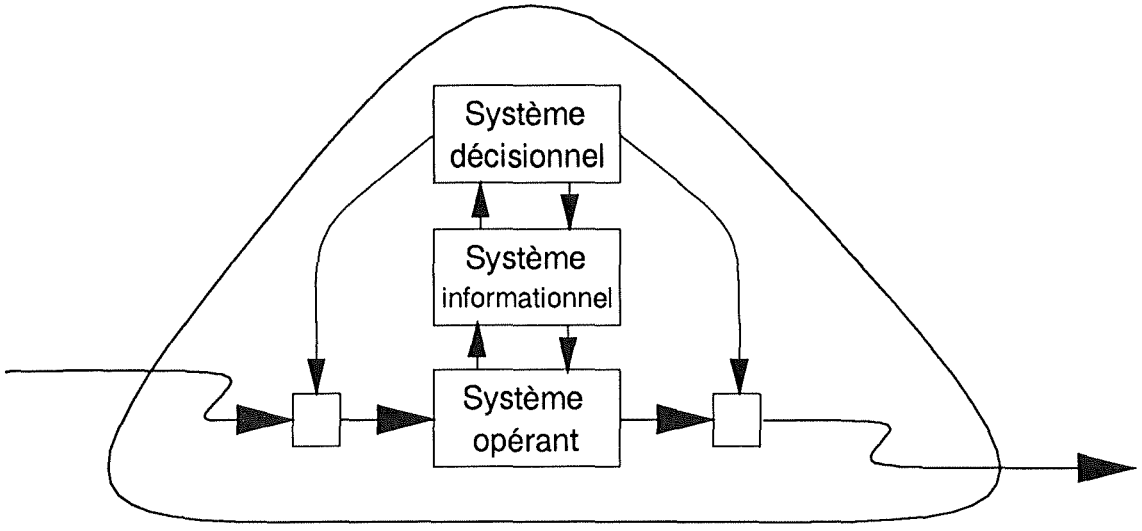


FIGURE 1.1

Dans ce schéma, nous pouvons également observer la contribution (feed back) du système de décision et du système opérant à l'alimentation du système d'information.

A ce jour, la définition de système d'information n'est pas encore définitivement arrêtée et plusieurs auteurs en ont une vision différentes. Citons par exemple, J. Gigh et L. Pipino [GIG86] qui caractérisent un système d'information comme étant :

> *"a collection of multilevel and recursively related subsystems where at least one person of a certain psychological type within some organizational context faces a problem of a given class for which evidence, rationality and logic are needed to arrive at a solution (that is, to select some course of action) and that the evidence is made available through some mode of presentation".*

De cette définition, nous retiendrons le fait de considérer un système d'information comme un ensemble de sous-systèmes.

Le groupe GALACSI [GAL84] définit le système d'une organisation sociale comme étant :

> *"l'ensemble des moyens, humains et matériels, et des méthodes se rapportant au traitement des différentes informations rencontrées dans les organisations".*

Cette définition n'est guère explicite et nous lui préfererons celle donnée par C. Rolland [ROL88] où il considère le système d'information d'une organisation comme un ensemble formé :

> *"- de collections de données, représentations partielles, en partie arbitraires mais nécessairement opératoires, d'aspects pertinents de la réalité de l'organisation sur lesquels on souhaite être renseigné. Ces collections inter-reliées, aussi cohérentes que possible, sont mémorisées et communiquées dans le lieu, le moment et la présentation appropriés aux acteurs qui en ont l'usage,*
>
> *- de collections de règles qui fixent le fonctionnement informationnel. Ces règles traduisent ou sont calquées sur le fonctionnement organisationnel. Partie intégrante du SI, ces règles doivent être connues des acteurs qui utilisent le SI. Elles leur sont nécessaires pour l'interprétation et la manipulation des collections de données,*
>
> *- d'un ensemble de procédés pour l'acquisition, la mémorisation, la transformation, la recherche, la communication et la restitution des renseignements,*

- d'un ensemble de ressources humaines et de moyens techniques intégrés dans un système, coopérant et contribuant à son fonctionnement et à la poursuite des objectifs qui lui sont assignés."

La notion de système d'information que nous retenons contient tous les éléments cités par C. Rolland en ajoutant qu'un système d'information peut se décomposer en sous-systèmes associés à des domaines particuliers de l'organisation. Nous appellerons désormais système d'information (SI), le système d'information d'une organisation ou un de ses sous-systèmes.

Section 1.2.2 Cycle de vie du SI

Sans entrer dans des querelles d'école, nous pouvons schématiser la conception d'un SI selon les étapes traditionnelles suivantes [ALVA88]:

"- Etude d'opportunité: Préparation d'un avant-projet de solution à partir des besoins exprimé par l'organisation;

- Analyse conceptuelle: Spécification d'une solution détaillée indépendante de tout moyen de réalisation;

- Conception technique: Description précise d'une solution qui prend en compte les caractéristiques logiques des moyens de réalisation;

- Réalisation: Production d'une solution exécutable en fonction des caractéristiques réelles des matériels, , des logiciels et de l'organisation;

- Utilisation et Maintenance: Utilisation avec correction éventuelle et évolution du système opérationnel."

Le schéma suivant (Figure 1.2) présente une vision linéaire du cycle de vie d'un SI; toutefois, il faut savoir qu'à l'issue de certaines étapes, des points de contrôle sont instaurés en vue de détecter les erreurs éventuelles. Ces points de contrôles induisent généralement des retours en arrière qui ne sont pas repris dans ce schéma.

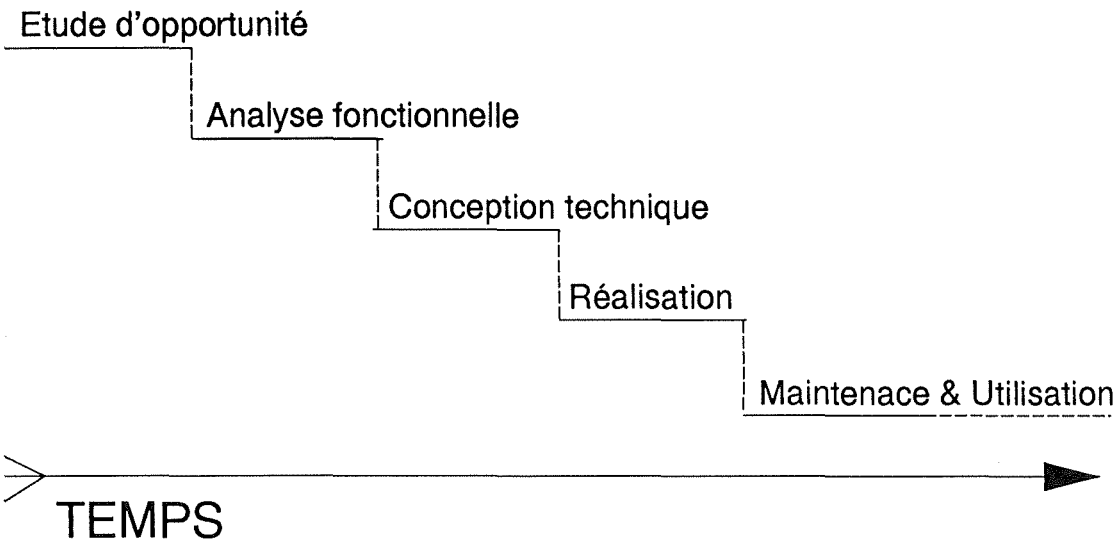


FIGURE 1.2

Dans le cadre de notre étude, nous focaliserons notre attention sur l'étape d'analyse conceptuelle avec comme objectif la spécification d'une aide à la réalisation de cette étape.

En terme plus précis, l'analyse conceptuelle consiste à spécifier une solution détaillée du SI désiré par l'entreprise en dehors du contexte d'implémentation de ce SI. Pour ce faire, la solution se situe à un niveau purement logique ignorant tout des contraintes techniques d'implémentation.

Les solutions proposées se présentent sous forme de schémas conceptuels; principalement , d'une part, nous aurons un ensemble de schémas relatifs à l'identification des données manipulées par le SI et d'autre part, un ensemble de schémas relatifs à l'identification des traitements et ressources mis en oeuvre par le SI.

SECTION 1.3 LA SPECIFICATION DES SI

Section 1.3.1 Concepts et principes

Avant d'aborder le thème principal de notre travail, à savoir l'aide à la spécification des SI, il nous semble important de préciser certains concepts. En effet, la littérature informatique abonde de termes tels "modèle", "méthode",... sans jamais en donner le sens précis, et parfois dans un emploi erroné ou en concurrence avec d'autres notions.

1) Définitions

a Méthodes

Le "PETIT ROBERT", Ed. 1973, nous propose pour la notion de méthode la définition suivante :

"Manière de faire quelque chose suivant une certaine habitude, selon une certaine conception ou avec une certaine application."

En 1990, le "PETIT ROBERT", nous propose les définitions suivantes:

"Ensemble des règles, des principes normatifs sur lesquels reposent l'enseignement, la pratique d'un art."

"Ensemble de démarches raisonnées, suivies pour parvenir à un but."

De ces définitions, se dégagent deux éléments essentiels à toute méthode: le but à atteindre et la manière d'y arriver. A ces deux éléments, nous rajouterons les ressources nécessaires à mettre en oeuvre pour atteindre l'objectif fixé.

Dans le contexte de notre travail, le but à atteindre se définit simplement par l'expression précise du résultat final. La manière d'y arriver définit quant à elle tous les éléments nécessaires à l'organisation de la tâche à accomplir et elle en précise tous les résultats intermédiaires. Il est à remarquer que le passage d'un résultat intermédiaire à un autre peut faire l'objet d'une méthode particulière plus précise que à la méthode 'première'. Les ressources nécessaires concernent tous les moyens:

le matériel, le personnel, le temps et le capital, à mettre en oeuvre pour arriver au résultat final. Il est évident que les détails de l'exploitation de ces ressources appartiennent à la manière d'opérer l'exécution de la tâche

Nous nous proposons de classer les méthodes en fonction de leur domaine c-à-d la tâche couverte depuis le point initial jusqu'au résultat final. Dans le cadre de la conception des SI, nous retenons trois ensembles particuliers de méthodes; soit les méthodes de conception, les méthodes de modélisation et les méthodes de spécification.

b Modèles

Le "Petit Robert", Ed. 1973, nous sert ici aussi de référence et nous propose la définition suivante:

"Représentation simplifiée d'un processus, d'un système."

En 1990, nous y retrouvons la même définition ainsi que la définition suivante:

"Personne, fait ou objet qui en font le représentant d'une catégorie".

Ces définitions ne peuvent entièrement nous satisfaire. En effet, dans le monde informatique, la notion de modèle est beaucoup plus complexe comme nous le montre F. Bodart dans [BOD83] en nous proposant la définition suivante :

" Un modèle est formé de concepts et de règles relatives à leur utilisation."

L.O. ALVARES nous propose une définition identique dans [ALVA88]:

Un modèle est "... un ensemble de concepts et de règles relatives à leur utilisation pour fixer le vocabulaire et le type d'abstraction."

Nous devons remarquer que nous voulons différencier les notions de modèle et de langage afin d'éviter que les concepts à mettre en oeuvre pour obtenir la représentation d'une "réalité" se confondent avec le mode de représentation choisi. Pour notre part, nous acceptons la définition proposée par L.O. ALVARES et dès

lors, comprenons la notion de modèle comme étant un ensemble de concepts proposé par une méthode pour définir, spécifier un aspect particulier d'un système d'information.

2) Méthodes de conception

Le domaine associé aux méthodes de conception couvre l'entièreté du cycle de vie d'un SI; le stade initial se définit par l'ensemble vide et le stade final par un système d'information opérationnel en phase de maintenance. D'après F. Bodart dans [BOD83], une telle méthode est :

"relative à l'emploi des ressources en vue de maîtriser les différentes étapes du cycle de vie d'un SI."

Les ressources auxquelles F. Bodart fait référence sont principalement l'organisation, les modèles et les outils.

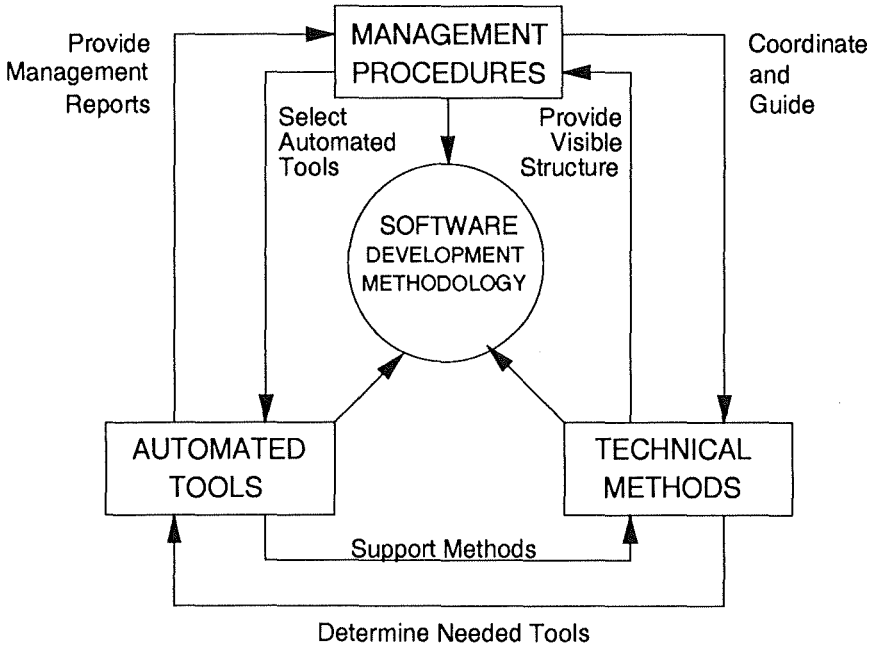


FIGURE 1.3

Le schéma présenté à la figure 1.3 et proposé par A.I. Wasserman dans [WAS82] en montre l'organisation.[1]

A.I. Wasserman commente son schéma de la manière suivante :

> *"... a methodology includes technical methods to assist in the critical tasks of problem solving, documentation, hierarchical decomposition, design representation, coding, systematic testing, and software configuration management. Such a methodology also includes management procedures to control the process of development and the deployment of these technical methods... The technical methods provide the intermediate results that are needed for effective managerial control, while the management procedures serve to allocate technical resources and support the development organization.*
>
> *Finally, automated support exists for the purpose of enhancing the effectiveness of the developer, with technical needs serving to drive the development of new automated tools... Management procedures determine the nature of the automated support that is provided... "*

Nous ne nous attarderons guère plus sur les méthodes de conception; toutefois, il nous faut remarquer que les décisions stratégiques sur les outils, points de contrôle et modèles à mettre en oeuvre en fonction de l'existant et des besoins de l'organisation se prennent à ce niveau et influent au premier degré sur la qualité du système d'information produit tant sur le fond que sur la forme.

1 *Nous y retrouvons les ressources référencées par F. Bodart.*

3) Méthodes de modélisation

Nous abordons ici le cadre spécifique de notre travail avec les méthodes associées directement à l'analyse conceptuelle; ces méthodes s'insèrent directement dans le cadre des méthodes de conception. Nous définissons le domaine des méthodes de modélisation par un état initial correspondant à l'expression des besoins de l'organisation et par un état final correspondant à l'expression d'une ou plusieurs solutions logiques reprenant au minimum un état des données et des processus nécessaires au fonctionnement du SI (Schéma conceptuel minimum).

L.O. Alvares dans [ALVA88] propose pour les méthodes de modélisation les quatre composantes suivantes sur base de [ROL86]:

> "- ***Les modèles***: *Ensembles de concepts et de règles relatives à leur utilisation pour fixer le vocabulaire et le type d'abstraction;*
>
> - ***Les langages***: *descriptions formelles des images du système d'information selon les modèles retenus;*
>
> - ***Les démarches***: *processus opératoires par lesquels s'organisent et s'effectuent le travail d'analyse, de modélisation, de description et de spécification du SI;*
>
> - ***Les outils***: *logiciels de documentation, d'aide à la spécification, d'évaluation, de traduction, de simulation."*

Ces composantes nous satisfont amplement. Toutefois, nous voulons apporter une précision sur la notion de "langage". Nous ne considérons les langages ni comme des outils ni comme des moyens (concepts) associés aux modèles mais comme éléments à part entière de la méthode de modélisation. Dans le cadre qui nous occupe, les langages permettent la représentation formelle d'une solution à partir de concepts appartenant aux modèles utilisés. Ils mettent à notre disposition une écriture formalisée afin de permettre le contrôle et la communication des solutions au sein de l'organisation et entre les membres de l'équipe d'analyse.

4) Méthodes de spécification

Ces méthodes sont en relation directe avec les méthodes de modélisation; elles en partagent le domaine et en possèdent les différentes composantes excepté les démarches.

En effet, les méthodes de spécification sont équivalentes à ces démarches. Nous avons choisi d'employer la notion de méthode de spécification pour des raisons de précision linguistique. En effet, dans la littérature informatique, les notions de démarche et de méthode de modélisation sont souvent confondues et/ou mélangées; nous avons tenu à préciser la différence afin d'éviter tout malentendu dans la suite de notre travail.

De plus, Nous voulons faire la distinction entre ce que les méthodes de modélisation proposent comme démarches d'utilisation des concepts définis dans leurs modèles et la démarche que peut utiliser un analyste ou une organisation dans la spécification d'un SI particulier.

En résumé, nous avons d'un côté un aspect "théorique" avec les méthodes de modélisation (IDA, REMORA, SADT,...) qui proposent des modèles, des langages, des démarches et des outils. Au sein de ces méthodes, les démarches définissent des règles de mise en oeuvre des concepts proposés par les modèles et des règles d'organisation des modèles. D'un autre côté, nous avons l'aspect "pratique" avec les méthodes de spécification qui proposent des règles de mises en oeuvre des modèles , des langages et des outils d'une méthode de modélisation dans le cadre d'une démarche particulière propre à un analyste ou à une organisation.

Section 1.3.2 Exemples

Nous voulons dans ce paragraphe vous montrer les différents aspects d'un concept de modélisation au travers des différentes notions soulevées dans les paragraphes précédents.

Pour rester dans le cadre strict de notre travail, nous ne prendrons pas en compte l'aspect relatif aux méthodes de conception, nous éviterons également d'envisager la relation entre les méthodes de conception et les méthodes de modélisation (- spécification) afin de

ne pas surcharger notre travail. Nous sommes cependant conscient de la nécessité de prendre en compte de telles relations dans le cadre plus explicite d'une étude complémentaire sur les méthodes de conception.

Il existe actuellement plusieurs méthodes de modélisation plus ou moins complexes, nous citerons par exemple MERISE, SADT, IDA, SSA, REMORA. La méthode MERISE servira ici d'illustration à notre approche des différentes notions de méthode.

Section 1.3.3 MERISE, Méthode de modélisation

La méthode MERISE s'organise autour de la différence entre données et traitement et relativement aux différentes étapes de conception du système; en ce qui nous concerne, nous nous limiterons à l'étape d'analyse conceptuelle. Nous avons donc à notre disposition deux modèles à savoir, le modèle conceptuel des données (MCD) et le modèle conceptuel des traitements (MCT); les figures 1.4 et 1.5 donnent pour chacun de ces modèles son insertion dans le contexte complet de l'organisation et une représentation schématique des concepts manipulés.

Les concepts sont définis de la manière suivante:

'Agent Externe':

Représente les partenaires extérieurs à l'organisation: client, fournisseur,...

'Domaine d'activité':

Représente Le découpage de l'organisation en domaine vise à obtenir des sous-ensembles homogènes, relativement indépendants et stables dans le temps.

Un domaine est un champs d'activité de l'organisation qui doit pouvoir être considéré comme un sous-système.

'Evénement/Résultat':

C'est la sollicitation concrète dont l'apparition est de nature à déclencher l'exécution de traitements déterminés.

'Modèle conceptuel des données':

Un modèle conceptuel des données sert à définir avec précision le vocabulaire et les concepts manipulés dans un domaine, en s'assurant de leur unicité. Il s'agit donc de décrire la sémantique du système d'information du domaine, modélisée à travers des objets, des relations et des propriétés.

'Modèle conceptuel de traitements':

Un modèle conceptuel de traitements permet de représenter les tâches à effectuer pour passer de l'événement déclencheur du processus étudié au résultat attendu.

'Objet':

Un objet est un concept global d'information décrit, sous forme extensive, par une liste de propriétés qui lui sont spécifiques et qui n'appartiennent à aucun autre objet.

Un objet est doté d'une existence propre dans la mesure où il peut être identifié de façon autonome.

'Opération':

Une opération est un ensemble de tâches exécutées de façon ininterrompue en réaction à un ou plusieurs événements synchronisés.

La notion d'opération sert à décomposer un processus en modules de traitement qui s'enchaînent.

'Processus':

Vu comme une boite noire, un processus est un grand module de traitements représentant la réaction de l'organisation à une catégorie d'événements extérieurs.

Un processus peut également être défini comme un ensemble de procédures de même nature, c'est à dire l'ensemble des chemins possibles permettant de passer d'une famille d'événements aux résultats correspondants.

'Propriété':

On appelle propriété une information élémentaire, c'est à dire le plus petit élément d'information qui a un sens pour le domaine étudié.

'Relation':

Une relation représente soit:

- Une association entre deux objets; dans ce cas, elle n'est pas porteuse de propriété;

- Un concept global d'information décrit par une liste de propriétés spécifiques et dont l'identifiant est la concaténation des identifiants de deux ou plusieurs objets.

Dans les deux cas, la relation n'a pas d'existence propre.

'Synchronisation':

La règle de synchronisation détermine les conditions requises pour déclencher une opération.

On l'exprime sous la forme d'une proposition logique portant sur la présence simultanée ou exclusive des événements déclencheurs de l'opération.

'Tâches':

Elle représente le plus petit élément de traitement intervenant dans une opération ou dans une phase.

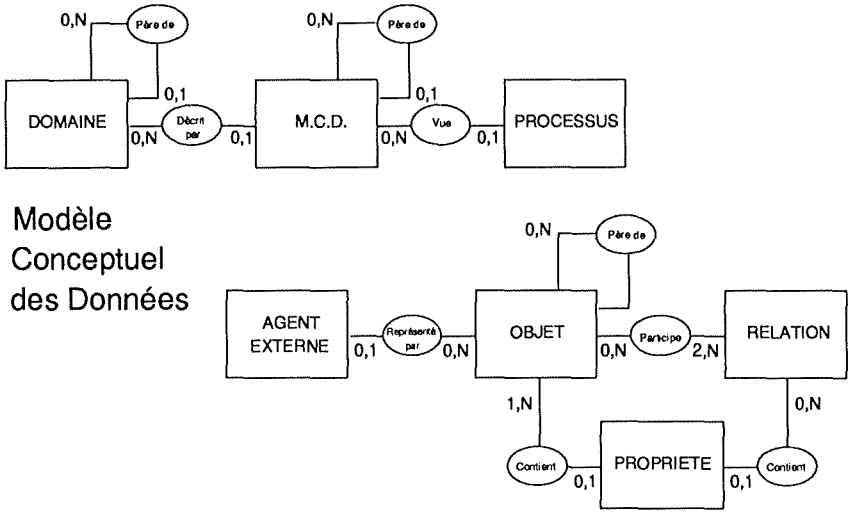


FIGURE 1.4

MERISE met à notre disposition des langages formels et graphiques relatifs à chacun des modèles. Actuellement, seuls les langages graphiques sont utilisés, en liaison avec des interfaces graphiques (Cfr. les figures 1.4 et 1.5 pour le langage d'expression graphique d'un MCD).

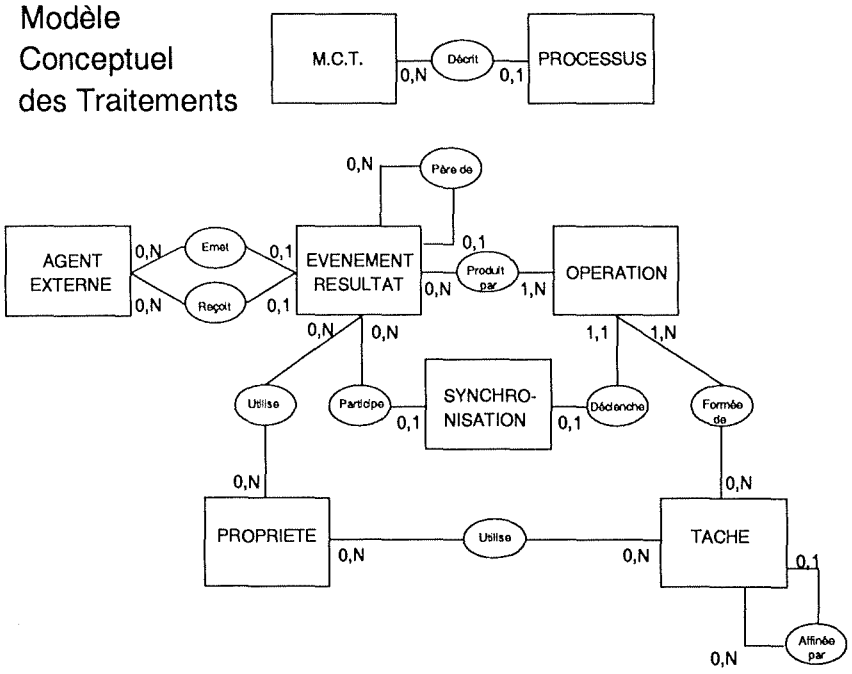


FIGURE 1.5

Plusieurs outils mettent en oeuvre un support à la spécification dans le cadre de la méthode MERISE, par exemple Message Conception [CEC87], PACBASE [CGI87], MEGA [ROU87] et Conceptor [VES87].

MERISE propose deux démarches complémentaires, une démarche par niveau et une démarche par étape.

La démarche par niveau établit la différence de spécification d'un point de vue conceptuel, organisationnel et technique. Etant donné que nous nous sommes limité à l'analyse fonctionnelle, nous avons retenu uniquement le niveau conceptuel.

La démarche par étape se déroule en quatre phases, une étude préalable, une étude détaillée, la phase de réalisation et la mise en vigueur.

Suite à cette limitation à l'analyse fonctionnelle, nous examinerons uniquement les étapes d'étude préalable et d'étude détaillée.

En fait, l'étude préalable permet de préciser les frontières du domaine de l'analyse, de modéliser différentes solutions et de mesurer les gains et les coûts de chaque solution. En finalité

L'étude détaillée vise à identifier de façon exhaustive et détaillée l'ensemble des données et des traitements de la solution retenue.

Section 1.3.4 MERISE, Méthode(s) de spécification

Dans le cadre de la conception d'un SI, nous avons défini les méthodes de spécification comme étant la particularisation d'une démarche d'une méthode de modélisation dans le cadre d'une organisation et d'un projet donnés.

Dans notre exemple, la méthode de modélisation choisie par l'organisation est la méthode MERISE. Le projet que nous avons choisi pour exemple est le calcul d'une prime au salaire de chaque employé de l'organisation relativement à la réalisation des objectifs fixés pour chaque employé. Le paiement des primes n'est pas pris en compte car il appartient au domaine des finances de l'organisation. La notion d'objectif n'est pas encore définie au sein de l'organisation.

Sur base de la définition du projet, la notion d'"agents externe" présentée dans MERISE n'est pas nécessaire et ne sera donc pas prise en compte dans la méthode de spécification.

Etant donné que la notion d'objectif n'existe pas encore dans l'organisation, la priorité est donnée à la définition de cette notion (et uniquement de celle-là) relativement à chaque domaine de l'organisation. La deuxième étape de ce projet concerne la spécification d'une

procédure d'évaluation de l'objectif et du calcul de la prime en fonction de cette évaluation. Finalement, l'analyse devra aborder l'intégration du projet au système de calcul du salaire existant actuellement.

Préalablement au projet, il faudra définir les différents domaines de l'organisation, les données et traitements relatifs au calcul du salaire actuel.

Nous venons de définir les grandes lignes de la démarche de la méthodes de spécifications et ses 5 principaux modèles:

1 - Définition des domaines;
2 - Définition des données et traitements du système existant;
3 - Définition de la notion d'objectif;
4 - Définition des nouveaux traitements;
5 - Intégration du nouveaux système.

Au niveau de chaque modèle, l'analyste peut organiser son travail librement et diviser chaque modèle en sous-modèle relativement à des objectifs particuliers qu'il se fixe. Normalement, ces objectifs restreignent encore l'emploi des concepts offerts par la méthode.

Par exemple, par rapport à chaque modèle, l'analyste peut décider de travailler premièrement sur les données et seulement après aborder l'analyse des traitements. Par rapport aux traitements, il peut décider d'envisager les opération(s) et leurs résultat(s) sans tenir compte de leur enchaînement (point de synchronisation).

Dans le modèle relatif à la définition de la notion d'objectif, l'analyste peut dans un premier temps n'envisager que la définition des objets et de leur(s) propriété(s) pour aborder ensuite les relations existants entre ces objets.

Nous remarquons que l'analyste dispose librement des concepts de la méthode de modélisation pour organiser son travail relativement aux objectifs imposés par l'organisation.

Globalement, l'organisation a donc défini l'ensemble des concepts qu'elle va manipuler dans le cadre de ce projet en particulier, cet ensemble est en fait un sous-ensemble

des concepts de la méthode de modélisation. Elle a défini une série de modèle (étape) et de points de contrôle relativement à la spécification du projet. Face à ces étapes, l'analyste a défini de nouveaux modèles en regroupant d'une manière différente les concepts utilisés.

Finalement, par rapport au travail de spécification en lui-même, l'analyste décidera d'une démarche personnelle.

Tout le schéma d'organisation du travail d'analyse que nous venons d'illustrer définit en fait une méthode de spécification. Nous y trouvons les modèles (Etapes organisationnelle de l'analyse et définition des concepts utilisés), les langages (Héritage direct de la méthode de modélisation en rapport avec les concepts repris dans les modèles) et les démarches (Organisation du travail d'analyse). Les outils sont également hérités de la méthode de modélisation.

CHAPITRE II L'AIDE A LA SPECIFICATION

SECTION 2.1 OBJECTIFS ET ENJEUX

Comme nous l'avons dit dans la Section 1.2.2, le cycle de vie d'un SI n'est pas linéaire et dans la pratique, les retours en arrière en vue de corriger les erreurs découvertes au point de contrôle sont fréquents. Ces retours en arrière consomment du temps et de l'argent, ce qui augmente la facture du développement du SI.

Le premier objectif de l'aide à la modélisation des SI est de permettre aux analystes de fournir des résultats d'une plus grande qualité tant au niveau de leur cohérence et de leur complétude qu'au niveau de leur adéquation aux besoins de l'entreprise. Cette amélioration contribuerait directement à une diminution des retours en arrière et donc à une diminution des coûts de développement.

Dans un deuxième temps, il est constaté une différence au niveau des investissements en travail et en capitaux octroyés aux différentes étapes de la conception d'un SI. Cette situation provient de la négligence accordée aux premières étapes du cycle de vie d'un SI et au passage prématuré à la réalisation. De ce fait, bon nombre d'erreurs passent inaperçues, et la maintenance du SI pour la correction de ces erreurs gonfle le budget initial octroyé. Plus grave encore, il arrive qu'au terme de la réalisation, le SI obtenu ne correspondent pas ou plus complètement aux besoins et désirs de l'organisation. Ici encore, la maintenance du projet pour une meilleure adéquation aux souhaits de l'entreprise va gonfler les budgets et accaparer le maximum des investissements effectués.

De plus, cet accroissement des problèmes de maintenance bouleverse les planning des cellules de développement et pose le problème du respect des délais fixés au cours de l'analyse.

Le deuxième objectif de l'aide à la modélisation des SI est de fournir un meilleur cadre de travail aux analystes afin de diffuser et d'améliorer les méthodes de modélisation; les outils proposés devraient engendrer un attrait pour leur utilisation. Dans cette optique, l'analyse conceptuelle pourrait obtenir une plus grande part des investissements et grâce à l'effort fourni, nombres d'erreurs pourraient être évitées. Le surcroît d'investissement octroyé à l'analyse conceptuelle serait largement compensé par un allégement des coûts de maintenance; indirectement, nous obtenons une diminution des coûts de développement.

L'apparition de nouvelles techniques de programmation représente une ouverture dans la création d'outils sophistiqués. L'emploi de ces nouvelles techniques devrait permettre le développement d'outils d'aide à la spécification munis de puissantes interfaces conviviales, de procédures complexes de contrôle et de traitement des erreurs, de procédures performantes de recherche et de tri. De tels outils devraient sans conteste faciliter l'approche des analystes vis à vis des méthodes de modélisation et en permettre une plus large diffusion. Cette "vulgarisation" des méthodes de modélisation devrait améliorer la qualité des SI conçus.

Dans cette vulgarisation, deux éléments ne doivent pas être négligés:

- d'une part, l'analyse conceptuelle s'inscrit dans un cycle et la réalisation de l'entièreté de ce cycle doit être assurée. De ce fait, les outils d'aide à la spécification doivent s'inscrire dans l'ensemble des outils d'aide couvrant le cycle de conception. Dans le cadre d'une méthode de conception particulière, ces outils doivent travailler de concert, en symbiose avec l'évolution du projet;

- d'autre part, le développement d'un SI s'effectue rarement à charge d'une seule personne; une équipe entière collabore à sa réalisation. L'aide fournie doit permettre le travail simultané de plusieurs analystes et viser à satisfaire l'équipe dans son travail de collaboration et l'analyste individuel dans sa démarche personnelle.

Si nous examinons le commentaire de A.I. Wasserman sur l'organisation d'une méthode de conception (Cfr. 2)), il en ressort différents points afférents aux qualités des outils d'aide à la conception.

Dans un premier temps, la nécessité de fournir un maximum de documentation pour aider le management dans sa fonction de contrôle et d'organisation représente un point clé dans les fonctions à offrir; l'outil ne doit pas seulement être un support aux analystes mais à toute l'organisation. Dans ce contexte, la possibilité de partage de l'outil et la définition de niveau d'accès ne doivent pas être négligée, nous dirons même qu'il importe d'en tenir compte.

Dans un deuxième temps, il apparaît également que le management doit pouvoir exercer un contrôle dans l'organisation du travail d'analyse. De ce fait, les outils doivent se présenter comme un ensemble modulaire souple permettant une hiérarchisation des tâches indépendante de leur organisation interne.

SECTION 2.2 POUR UNE AIDE INTELLIGENTE

Il n'est pas de notre intention de dénigrer l'aide apportée aux utilisateurs au travers des outils existants en la qualifiant de "bête" mais d'envisager la notion d'aide à l'utilisateur dans le cadre d'une action dynamique. En effet, bon nombre d'outils actuellement mis en oeuvre proposent une aide passive aux utilisateurs sous forme de conseil à l'utilisation. Notre ambition est de définir une aide active propre aux besoins de l'utilisateur. Dans ce cadre, nous envisageons de définir des niveaux d'aide relatifs aux différentes fonctionnalités de l'outil.

Section 2.2.1 Conseil d'utilisation du système

Ce premier niveau détermine l'aide définie sous forme de conseil ou d'enseignement à l'utilisateur. Ce niveau concerne des points d'aide statiques au sein de l'outil. Ces points s'associent à des fonctions et l'aide fournie concerne l'emploi de la machine (hardware) dans le cadre de ses fonctions ou l'emploi des fonctions dans le cadre de l'outil. Cette aide correspond à l'établissement d'un mode d'emploi interactif au sein de l'outil.

Exemples :

- *Définition du clavier et des fonctionnalités des touches,*
- *Définition de l'environnement de travail,*
- *Définition des fonctions,*
- ...

Section 2.2.2 Personnalisation du système

Le deuxième niveau d'aide détermine une aide dynamique définie sous forme de règles d'exploitation de l'outil. Ce niveau concerne l'enchaînement automatique de fonctions dans le cadre d'une méthode de spécification donnée. Ce niveau est une option mise à la disposition de l'utilisateur afin de lui éviter le passage successif par différents menus de commande et de lui permettre d'assurer d'une certaine manière la cohérence de son travail de spécification.

Notre volonté par rapport à ce niveau d'aide va plus loin que la simple définition de 'macros' telle que nous pouvons l'apercevoir dans d'autres applications; nous voulons aller plus loin que la simple juxtaposition de fonctions. Notre propos est de définir des niveaux de dynamisme du système d'aide; en passant du step by step à la dérivation de spécification à partir d'autre spécification.

Exemples :

- *Obligation pour l'analyste de définir le niveau de tous les processus*
- *Dérivation du niveau d'un processus par l'intermédiaire de la relation 'Processus Père' et de la hiérarchie des niveaux.*
-...

Section 2.2.3 Historique d'utilisation du système

Le troisième niveau d'aide est un niveau implicite à l'outil que nous proposons. En effet, il concerne l'ensemble de l'outil et détermine une aide dynamique permettant à l'utilisateur d'accéder à l'information relative aux travaux déjà effectués. L'aide telle que définie ci-avant vise à assurer à l'utilisateur la consultation des données relatives au système d'information en cours de spécification et ce, à tout moment, quel que soit l'environnement de travail.

Cette consultation relative au travail déjà effectué ne concerne pas seulement les spécifications correctes saisies mais également les erreurs rencontrées, les corrections apportées et la définition de l'état final attendu.

Située entre les deux niveaux que nous venons de définir, nous définissons comme une aide la possibilité de définir la quantité d'information contenue dans les interfaces. Nous pensons en effet que des écrans trop riches ou trop pauvres en information peuvent mécontenter l'utilisateur. De ce fait, nous envisageons une définition des écrans sur base des besoins exprimés par l'utilisateur. Nous estimons que cette aide est dynamique car il

doit être possible d'envisager la représentation d'un même concept sous plusieurs formats en fonction du contexte de manipulation de ce concept et il appartiendra au système de gérer ces différentes représentations.

Il faut préciser ici que nous ne parlons pas de mémoriser diverses versions d'un même objet mais sur base d'une définition, pouvoir donner plusieurs représentation d'un même objet.

Section 2.2.4 Conseil sur le travail effectué

Cette aide vise à offrir à l'utilisateur des conseils par rapport à l'état de son travail. Il est évident qu'une telle aide est très difficile à mettre en place et qu'elle repose sur des connaissances très subjectives (définition d'une bonne analyse).

La réalisation de ce niveau d'aide doit se faire dans le cadre d'un système-expert dont une des bases de connaissances doit contenir les règles pour définir les critères de qualité de spécifications établies dans un domaine particulier.

Cette aide devrait aborder non seulement le cadre d'un contrôle a posteriori des spécifications effectuées mais également un contrôle 'online' lors de la saisie de ces spécifications.

SECTION 2.3 APPROCHE TRADITIONNELLE VERSUS SYSTEME EXPERT

L'objectif de ce paragraphe n'est certes pas d'établir une critique de la conception traditionnelle des logiciels mais de préciser les raisons d'un choix incontestablement orienté vers une conception de style système expert.

La première de ces raisons est relative à la structure des données. En effet, il est inconcevable d'envisager une définition ferme et rigide des données manipulées par notre outil. Notre étude vise la spécification d'un outil générique capable de supporter toute méthode de modélisation et dans le cadre d'une de ces méthodes toute méthode de spécification. De ce fait, étant donné le nombre variable des concepts manipulés au travers des méthodes de modélisation et au sein d'une méthode de spécification, il est impossible de spécifier exactement les données manipulées par l'outil. Dans ces conditions, nous avons besoin d'une structure de données évolutives du genre "frames".

La deuxième raison de notre choix concerne les procédures informatiques de traitement de l'information. La principale différence entre les deux approches réside dans la perception qu'elles ont du problème. La conception traditionnelle aborde les problèmes en terme de fonctionnalités. A son opposé, la conception de style système expert propose une approche du problème en terme de logique et de déduction.

A notre avis, il ne semble pas exister de véritable opposition entre les deux approches mais un problème d'adéquation de la technique à employer face à l'élaboration d'un système. Dans le cadre d'une expertise, le raisonnement et la déduction priment sur la fonction mais au sein d'un système informatique, existe également tout le problème de la gestion de la machine et de son environnement; dans ce contexte, la fonction ne peut être négligée.

Afin de clarifier notre pensée, nous avons dégager plusieurs types de procédures informatiques: les procédures de 'processus', les procédures d'interaction et les procédures de gestion des interactions et envisager pour chacun de ces types l'approche la plus appropriée. Le schéma de la figure 2.2 en montre l'organisation.

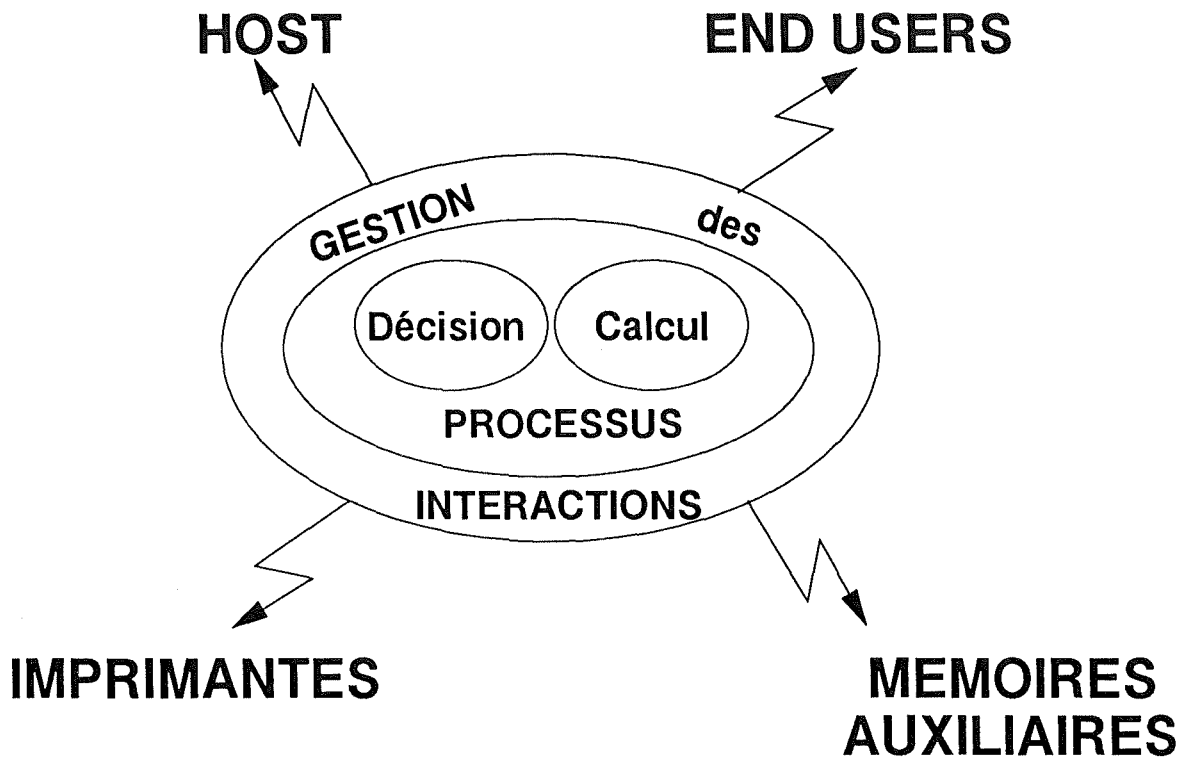


FIGURE 2.2

Les procédures de 'processus' se divisent en deux groupes, les procédures de calcul et les procédures de décision. Les procédures de calcul concernent l'établissement de résultats mathématiques. Ces procédures seront de préférence envisagées de manière traditionnelle. Les procédures de décision concernent quant à elles, la demande d'interaction avec l'environnement, le contrôle et l'interprétation des résultats issu de calcul et/ou des interactions avec l'environnement et le contrôle de la dynamique des procédures. Ce type de procédures sera plus facilement envisagé dans le cadre d'une conception de style système expert.

Les procédures de gestion des interactions concernent la vérification des paramètres circulant entre l'environnement et les procédures de 'processus'. Cette vérification s'effectue à un niveau purement syntaxique: vérifier l'appartenance d'une valeur à un domaine; assurer l'ordre dans une lecture (écriture) séquentielle, détection d'erreurs lors de l'exécution d'une

interaction... Ces procédures peuvent s'envisager selon les deux approches, le choix dépend de la qualité des données manipulées et de la richesse des contrôles à effectuer; toutefois, nous pensons que l'approche traditionnelle est la plus fréquente et la mieux adaptée.

Les procédures d'interactions effectuent la liaison entre les procédures de gestion des interactions et l'environnement. La conception de ces procédures dépend essentiellement de la qualité de l'environnement qu'elles mettent en oeuvre. En effet, chaque procédure est propre à un type d'environnement et du contexte de manipulation des données dans cet environnement dépend le choix pour un style de conception. Si l'environnement est complexe et met en oeuvre des mécanismes de déduction pour obtenir un résultat, la procédure d'interaction sera vraisemblablement envisagée dans une approche système-expert. Si l'environnement est complètement déterminé et fixe dans l'obtention de résultat, la procédure d'interaction sera vraisemblablement envisagée dans une approche traditionnelle.

Pour terminer, nous ajouterons que dans la majorité des cas, la conception d'une procédure d'un SI n'appartient pas de manière systématique et exclusive à l'un des trois types proposés mais fait généralement référence aux trois simultanément. L'objectif de notre distinction est d'éviter toute confusion, d'éviter de concevoir de manière traditionnelle au sein d'un environnement système expert et inversement.

Notre volonté est donc de concevoir un système mixte où la plus grande part est octroyée à la mise en oeuvre d'un système expert mais où la manipulation d'information sortant du contexte des bases de connaissances est conçue de la manière la plus efficace possible et vraisemblablement dans le cadre d'une approche traditionnelle afin de ne pas gonfler inutilement les bases de connaissances.

CHAPITRE III
POUR UNE APPROCHE SYSTEME-EXPERT

SECTION 3.1 L'ORGANISATION D'UN SYSTEME EXPERT

D'une manière générale et dans la littérature courante, l'organisation d'un système expert est représentée par le schéma de la figure 3.1, schéma où figurent les trois éléments suivants : une base de connaissance, un moteur d'inférence et une interface.

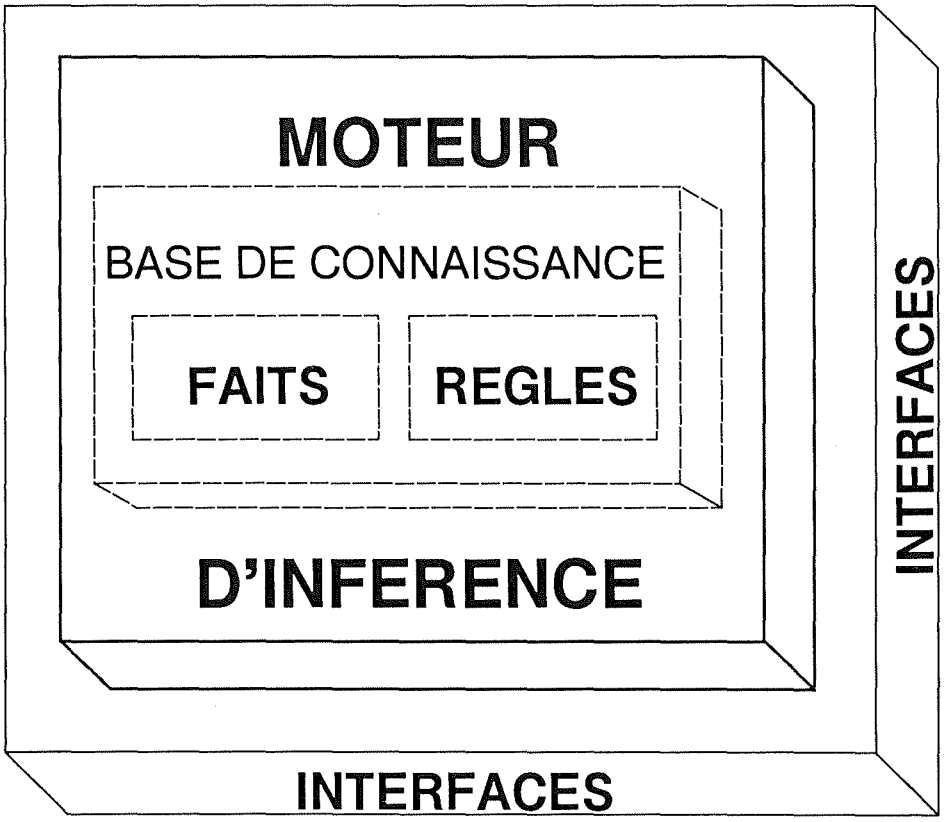


FIGURE 3.1

Section 3.1.1 La base de connaissance

Telle que nous la concevons, la base de connaissance est un ensemble d'information répertoriée en deux grandes catégories: les faits et les règles. Les faits d'une base de connaissance correspondent aux données contenues dans une base de données traditionnelle,

ils représentent les objets contenus dans l'univers de travail auquel correspond la base de connaissance. Les règles correspondent à l'établissement de contraintes sur ces objets ou à la description de propriétés dynamiques de ces mêmes objets.

Exemples de contraintes:

- *Obligation pour tel objet d'avoir telle ou telle propriété;*
- *Organisation d'une hiérarchie au sein de d'une classe d'objet;*
- ...

Exemples de propriétés dynamiques :

- *Dérivation de propriété à partir d'une hiérarchie définie au sein d'une classe d'objet;*
- *Dérivation de propriété à partir de certaine relation définie entre différents objets;*
- ...

Selon l'environnement de conception choisi, nous pouvons bénéficier du concept d'héritage. Ce concept établit une hiérarchisation des faits telle que les descendants d'un fait donné héritent des caractéristiques de leur(s) ascendant(s); elle induit implicitement des règles de manipulation des faits. Cette notion d'héritage est disponible dans une formalisation des faits de type "frames"; dans un tel environnement on ne parlera plus de faits mais d'objets et de classe d'objets

Section 3.1.2 Le moteur d'inférence

Le moteur d'inférence est un outil plus ou moins sophistiqué de mise en oeuvre des connaissances contenues dans une ou plusieurs bases. Comme son nom l'indique, il établit un mécanisme d'inférence sur les règles et les faits d'une ou plusieurs bases de connaissance afin de vérifier différentes thèses émises par l'utilisateur et d'en déduire le cas échéant de nouveaux faits.

Nous ne nous attarderons pas sur les problèmes d'inférence; nous noterons simplement qu'il est possible d'inférer en avant et en arrière. L'inférence en avant ou chaînage avant des règles correspond à déterminer si les faits soutiennent la thèse énoncée soit une conception des règles de style:

"Si j'ai X alors j'ai Y".[2]

Exemples:

- *Si mon objet a un nom, un niveau et émet/reçoit des messages*
  alors mon objet est un processus.
- *Si mon objet a un nom, une nature et émet/reçoit des messages*
  alors mon objet est une interface.
- *Si mon objet a un nom et est émis/reçu par une interface et/ou un processus*
  alors mon objet est un message.

L'inférence en arrière ou chaînage arrière des règles correspond à déterminer si la thèse est vérifiée par les faits connus, soit une conception des règles de style: "

Pour avoir Y, il faut X".[2]

Exemples:

- *Pour avoir un processus, il faut que l'objet ait un nom, un niveau et émette/reçoive des messages.*
- *Pour avoir une interface, il faut que l'objet ait un nom, une nature et émette/reçoive des messages.*
- *Pour avoir un message, il faut que l'objet ait un nom et soit reçu/émis par un processus et/ou une interface.*

Il existe actuellement des moteurs d'inférence travaillant soit en chaînage avant, soit en chaînage arrière, soit offrant les deux possibilités dans le cadre d'un même système expert; le contrôle de l'inférence étant défini par le concepteur au sein même de la règle.

2 *X = Faits et Y = Thèse*

Section 3.1.3 L'interface

Dans le cadre de notre introduction, nous avons abordé les types de procédures relatives aux interactions à savoir les procédures de gestion d'interaction et les procédures d'interaction en elles-mêmes. L'interface telle que nous la concevons contient à proprement parler ces deux types de procédures. Contrairement à ce qui est souvent émis dans la littérature informatique courante, nous ne limitons pas notre notion d'interface au simple contact utilisateurs-système expert (plus particulièrement le gestionnaire de la base de connaissances) mais nous envisageons tout l'ensemble des interactions entre le système expert et son environnement à savoir, utilisateurs bien évidemment mais aussi mémoires auxiliaires, hosts éloignés, imprimantes,...

Dans ces conditions, il faut envisager au cas par cas chaque procédure d'interaction et de gestion d'interaction afin de déterminer l'approche la mieux adaptée pour la conception de chacune d'elles. Nous pensons cependant que l'approche traditionnelle est la mieux adaptée à la conception d'une interface.

Toutefois, nous ne devons pas oublier l'aspect générique de notre système d'aide et dés lors, il est important de prévoir la paramétrisation de l'interface de telle manière que l'interaction se fasse éventuellement sur base de connaissance contenue au niveau du système-expert.

Exemples:

- Saisie d'une valeur de type T de longueur L avec M comme valeur maximum où T, L, M sont des connaissances définies dans le système expert;
- Affichage de la valeur V sur la ligne L dans la couleur C et le style S où V, L, C, S sont des connaissances définies dans le système-expert;
- Imprimer un rapport reprenant tous les processus définis
- Exporter vers le Host H les valeurs $V_1,...,V_n$ où les valeurs $V_1,...,V_n$ sont des connaissances définies dans le système-expert, l'adresse du host H et les informations relatives au protocole de communication font partie de la procédure d'interaction.
- ...

Finalement nous pouvons également considérer toute application extérieure au système-expert comme faisant partie de son environnement et dès lors si elle s'avère utile au fonctionnement du système-expert, nous devons prévoir un point de communication entre ces applications et le système au niveau de l'interface. Par exemple, c'est à ce niveau que nous devons prévoir la communication entre le système et les différents outils d'une méthode de conception. Les autres applications auxquelles nous pensons sont des procédures de tris, des programmes de mise en page et de production de rapport automatique,...

SECTION 3.2 LES CONNAISSANCES

L'aspect le plus important dans la conception d'un système expert est sans conteste l'établissement des connaissances nécessaires et suffisantes aux fonctionnement de ce système-expert. Dans un premier temps, nous analyserons simplement les connaissances relatives à la description des concepts manipulés au sein d'un système d'aide à la spécification; la participation de ces connaissances à la dynamique de l'outil sera envisagée dans le chapitre suivant.

Nous distinguerons pour la forme les notions de base de faits et de base de règles. Une base de faits ne contient que des faits relatifs à la description statique d'un domaine particulier.

D'une manière générale, nous pouvons déjà recenser diverses connaissances suite à nos précédentes discussions. En premier lieu et sans hésiter, nous pouvons citer les méthodes de modélisation et les méthodes de spécification. Nous pouvons considérer également le système en cours de spécification comme une source de connaissance sur le travail déjà effectué. L'aide mise à la disposition de l'utilisateur est également une connaissance quel que soit son niveau (Cfr. SECTION 2.2) non pas au niveau de la forme mais au niveau de son contenu, notons qu'une partie de ces connaissances sera directement dérivée d'autres connaissances telles les connaissances relatives aux méthodes de modélisation.

Enfin, pour terminer, nous examinerons les connaissances relatives au domaine d'application du système d'information à spécifier et les connaissances associées à l'expérience d'analyse que peut acquérir un analyste par rapport à une méthode de modélisation.

Nous pourrions encore préciser les connaissances relatives à la création d'un méthode de spécification, les connaissances relatives à la dynamisation d'un outil d'aide à la spécification ou encore les connaissances relatives à la spécification d'une interface conviviale mais ces connaissances seront envisagées explicitement dans les chapitres suivants.

Section 3.2.1 Les méthodes de modélisation

Telles que nous les avons décrites dans le paragraphe 1.2.1.3, les méthodes de modélisation comprennent les éléments suivants : des modèles, des langages, des démarches et des outils.

1) Les modèles

La connaissance relative aux modèles s'établit sur une base de faits et une base de règles:

- La base de faits contient la liste exhaustive des définitions de concepts (objets, propriétés) manipulés dans tous les modèles associés à la méthode, ainsi que la définition des relations entre ces concepts.

Exemples:

- *CONCEPT (OBJET : processus),*
- *CONCEPT (OBJET : message),*
- *CONCEPT (PROP. : nature),*
- *CONCEPT (PROP. : niveau),*
- *Définition des domaines de valeurs :*
  - *EXTENSION (niveau : 'PH', 'AP', 'PR', 'FC'),*
  - *EXTENSION (nature : '00','01'),*
- *RELATION (est de niveau, processus, niveau)*
- *RELATION (est de nature, message, nature)*
- *RELATION (généré par, message, processus)*

- La base de règles contient la liste exhaustive des définitions de contraintes relatives aux concepts définis dans la base de faits tant au sein d'un modèle particulier qu'au niveau de plusieurs modèles. Ces règles expriment les contraintes sémantiques applicables sur les concepts de la base de faits et les propriétés dynamiques de ces mêmes concepts.

Exemples:

- Tout processus a un niveau.
- Tout processus de niveau 'AP' a pour père un processus de niveau 'PR'.
- Tout message a une nature.
- Tout message est généré par un processus.
- Le processus père d'un processus de niveau 'phase' est un processus de niveau 'application'.

2) Les langages

La connaissance relative aux langages s'établit sur une base de faits et une base de règles:

- La base de faits contient la liste exhaustive des définitions de formalismes associés aux concepts manipulés dans tous les modèles associés à la méthode, ainsi que la définition des formalismes associés aux relations entre ces concepts. Les formalismes définis concernent tous les moyens d'expression utilisés pour la spécification; nous pensons aux formalismes relatifs aux supports écrits, aux écrans mais aussi éventuellement, aux formalismes associés à des stockage sur mémoires auxiliaires, aux formalismes employés pour des communications avec des hosts éloignés, ...

Exemples:

- La définition de la codification de certaines valeurs:
CODE (niveau, PROJET, 'PR')
CODE (niveau, FONCTION, 'FC')
CODE (nature, EXTERNE, '01')
EXTENSION (nature, INTERNE, '00')
- La définition de la représentation graphique:
GRAPHIQUE (processus, statique, rectangle [L,l])
où L et l donnent les dimensions de la figure.
GRAPHIQUE (condition, cercle [R])
GRAPHIQUE (condition, losange [L,l])
- la définition de la représentation textuelle:
TEXTE (niveau, PROJET, FR, 'Le processus [X] est de niveau projet')
TEXTE (niveau, FONCTION, EN, 'The process [X] is of function level')

Premièrement, il faut noter qu'il existe une projection des concepts contenus dans la base de faits des modèles vers ceux contenus dans la base de faits du langage. En effet, nous ne pouvons pas prétendre concevoir un concept sans lui donner une représentation et inversement concevoir une représentation pour un concept qui n'existe pas.

Deuxièmement, nous pouvons remarquer au travers de ces quelques exemples, que la représentation d'un concept n'est pas unique mais qu'elle dépend du contexte d'utilisation de la représentation, que ce soit au niveau linguistique *(EN, FR dans TEXTE)* ou au niveau du mode de représentation choisi *(CODE, GRAPHIQUE, TEXTE)*.

- La base de règles contient la définition des contraintes relatives aux concepts définis dans la base de faits tant au sein d'un modèle particulier qu'au niveau de plusieurs modèles. Ces règles expriment les contraintes syntaxiques applicables sur les concepts de la base de faits ainsi que les contraintes applicables sur les contextes d'utilisation afin de limiter l'emploi de mode de représentation d'un concept à certains contexte bien précis.

<u>Exemples:</u>

- Le nom de tout objet est en vingt caractères
- Le nom d'un processus est toujours en majuscule.
- Le nom d'un relation multivaluée commencent toujours par une majuscule.
- Le nom d'une fonction monovaluée commence toujours par une minuscule.
- Le GRAPHIQUE ne peut pas être associé à du TEXTE au sein d'un même écran.
- ...

Il ressort assez clairement des exemples que nous venons d'exprimer que la base de connaissances relatives aux langages est en relation directe avec la spécification des interfaces du système d'aide. Nous envisageons cette interface comme un système à part entière indépendant du système d'aide; la cohérence entre les deux systèmes étant établie au travers de la projection devant exister de la base de connaissance 'MODELES' vers la base de connaissance 'LANGAGES'.

3) Les démarches

Nous avons dans la section Section 1.3.1 établi un lien étroit entre les méthodes de spécification, les méthodes de modélisation en général et les démarche des méthodes de modélisation en particulier.

Dans ce contexte, l'élaboration d'une base de connaissance relative aux démarches associées à une méthode de modélisation représente peu d'intérêt, étant donné leur pauvreté et le risque de redondance avec les bases de connaissances associées aux méthodes de spécification.

Cependant, nous pouvons établir certaines contraintes que les méthodes de spécification doivent respecter afin de préserver la cohérence et la complétude ainsi que la philosophie de la méthode de modélisation associée.

De ce fait, nous définissons une série de règles relatives à l'établissement de démarches particulières (méthodes de spécification) au sein d'une méthode de modélisation. Ces règles seront contenues dans la base de connaissance relative aux démarches d'une méthode de modélisation.

Exemples:

- *Définition d'un degré de complétude minimale d'un schéma de données.*
- *Définition de points de passages obligés.*
  La conception du schéma de la statique des traitements ne peut pas s'envisager avant la réalisation du schéma des données.
- *Définition d'incohérence au niveau d'association de règles de contrôles.*
- ...

Notons que bien que les exemples proposés soient simples, il ne faut pas oublier que ces règles représentent des aspects obligatoires imposés dans les démarches de la méthode de modélisation.

4) Les outils

Il nous apparaît de prime abord que les outils ne relèvent pas d'une connaissance pertinente pour le fonctionnement de notre système d'aide. Toutefois, nous pouvons remarquer que la connaissance des formats des données utilisées en entrée de ces outils peut être perçue comme pertinente pour la production de résultats intermédiaires utilisables par ces outils. Nous négligerons volontairement cet aspect et vous renvoyons à des études menées dans le cadre de méthode de conception afin de mieux cerner le contenu de telles connaissances; sachez toutefois qu'en ce qui nous concerne, nous envisagerions ces connaissances dans le cadre de la mise en oeuvre de bases de connaissances relatives aux interfaces associées à l'environnement des systèmes d'aide à la spécification des SI.

Section 3.2.2 Les méthodes de spécifications

Telles que nous les avons définies dans le paragraphe 4), les méthodes de spécifications représentent un potentiel de connaissance uniquement au niveau des règles de mise en oeuvre d'une démarche particulière propre à un analyste ou à une organisation. Dans ces conditions, nous devons établir une base de connaissance propre à chaque méthode de spécification définie, c'est-à-dire créée par un analyste ou une organisation. Cependant, d'un point de vue pratique, nous ne pouvons pas envisager une telle présentation des connaissances relatives aux méthodes de spécification. En effet, nous sommes persuadés que ces connaissances appartiennent à l'analyste et/ou à l'organisation et dés lors, il leur appartient de concevoir eux-mêmes la base de connaissance relative à leur méthode de spécification.

De ce fait, la représentation de ces connaissances passe par une définition exhaustive des règles de contrôles et de 'pilotage' dans l'utilisation des différents concepts des modèles et langages de la méthode de modélisation et par un outil permettant sur base de ces règles de contrôle et de 'pilotage' de concevoir, configurer la base de connaissance d'une méthode

de spécification particulière. Cet outil de configuration et le principe du pilotage seront étudiés dans le CHAPITRE IV CONFIGURATION D'UNE METHODE DE SPECIFICATION et le CHAPITRE V PILOTAGE DES SPECIFICATIONS

En ce qui concerne les règles de contrôle, leur définition a été effectuée au niveau de la définition des règles associées aux modèles et aux langages des méthodes de modélisation. Par contre, les règles de 'pilotage' ne sont encore définies nulle part excepté quelques-unes au niveau des règles reprises dans la base de connaissance associée aux démarches. Afin de rester cohérent, nous insérons également dans cette base la liste exhaustive des règles de 'pilotage' permettant la configuration d'une méthode de spécification.

Exemples de règles de pilotage:

- *Il est interdit de définir des processus.*
- *La définition du niveau d'un processus est obligatoire.*
- *les messages doivent être définis en premier lieu.*
- ...

Section 3.2.3 Le système d'information

Nous ne voulons pas parler ici des particularités du système d'information, ces connaissances étant retenues au niveau des connaissances définies par rapport au domaine d'application. Notre point de vue est de considérer que les spécifications effectuées sont une source de connaissance qu'il faut mettre à la disposition de l'analyste et sur laquelle vont s'exercer certains contrôles par rapport à l'introduction de nouvelles spécifications.

Nous avons typiquement ici une base de faits dont l'état initial est le vide et l'état final donne une représentation d'un système d'information. C'est au niveau de cette base que nous associerons les faits reprenant des éventuels remarques et commentaires de l'analyste par rapport à son travail de spécification ainsi que le relevé des différentes erreurs rencontrées et des corrections apportées.

Section 3.2.4 Le domaine d'application

Chaque domaine d'application présente des particularités propres que l'on peut prendre en compte systématiquement lors de l'élaboration d'un système d'information dans ce domaine. Nous pouvons prévoir la définition de règles de contrôle par rapport à ces particularités soit d'un point de vue strict (provoquant une erreur) soit sous un aspect de conseil à l'analyste afin de lui proposer une solution mieux adaptée au domaine d'application.

Les particularités du domaine d'application peuvent se présenter sous forme de structure d'objets, de traitements définis préalablement ou de relation avec d'autres systèmes d'information déjà en place (ou spécifiés).

Exemples:

- *Définition de rapports légaux.*
- *Définition du liens avec d'autres domaines:*
  Comptabilité, Finance, Personnel, ...
- *Méthode d'amortissement particulière.*
- ...

Section 3.2.5 Connaissances d'un expert en modélisation de SI

Pour la définition d'une aide établie sous forme de conseils relatifs au travail de spécification (Cfr Section 2.2.4), l'examen de telles connaissances est une étape obligée. Cependant, le caractère subjectif de ces connaissances les rendent difficilement crédibles et sujettes à de nombreuses controverses.

Nous pouvons répartir ces connaissances en fonction de la méthode de modélisation utilisée et en fonction du domaine d'application. Attention, il nous faut remarquer que ces connaissances ne sont pas inhérentes à la méthode ou au domaine mais relèvent de l'expérience du travail d'analyse acquise par un expert en relation avec telle ou telle méthode et/ou tel ou tel domaine.

A ce niveau, nous trouverons par exemple la définition d'une bonne méthode de spécification, la définition de critères prépondérants dans la spécification de certains objets, la définition d'une bonne analyse (état de complétude, de cohérence). Toutes ces connaissances ne transparaîtront pas systématiquement dans une base spécifique mais pourront être par exemple à l'origine de la définition de la base de connaissance d'une méthode de spécification.

SECTION 3.3 SYNTHESE DU SYSTEME-EXPERT

Nous allons dans cette section effectuer la mise en place d'un point de vue organisationnel de notre système-expert d'aide à la spécification des SI.

Partant de ce que nous venons de spécifier dans les sections précédentes et relativement au cadre de cette étude, nous négligerons les aspects de performance du système-expert, nous éviterons donc la discussion portant sur les qualités des moteurs d'inférence relativement aux modes de chaînages (avant - arrière), aux critères de sélection de règles, à la rapidité d'exécution. Nous imposerons uniquement comme limite à l'environnement de conception du système d'aide la possibilité de gérer des règles comme variables du système. Cette nécessité provient de notre choix de pouvoir créer une méthode de spécification sur base de la définition exhaustive des règles de contrôles et de pilotage associée à la méthode de modélisation utilisée.

Notre intérêt dans les chapitres suivants va se porter principalement sur le fonctionnement et l'organisation du système d'aide. Nous venons d'en préciser le contenu, il nous reste maintenant à en étudier la forme et la mise en place; la figure 3.2 nous en donne un premier aperçu.

Nous avons en premier lieu une série de bases de connaissances relatives à la définition d'une méthode de modélisation. Nous avons ensuite une interface permettant la création et la maintenance des connaissances contenues au sein de ces bases, cette interface doit permettre la création et la modification d'un fait ou d'une règle relativement aux modèles, aux langages, aux outils et aux démarches. [3]

La base relative au domaine de spécification est une base particulière n'appartenant pas à la méthode de modélisation mais spécifiquement à chaque domaine du SI à spécifier. Toutefois,

3 *Nous devons garder à l'esprit que la base de connaissances relatives aux démarches contient un relevé exhaustif des faits et règles permettant le contrôle et le pilotage du travail de spécification.*

nous pensons que la maintenance de cette base de connaissance doit se faire également par l'intermédiaire de cette interface; il en est de même pour la base relative aux connaissances d'un expert en matière d'analyse.

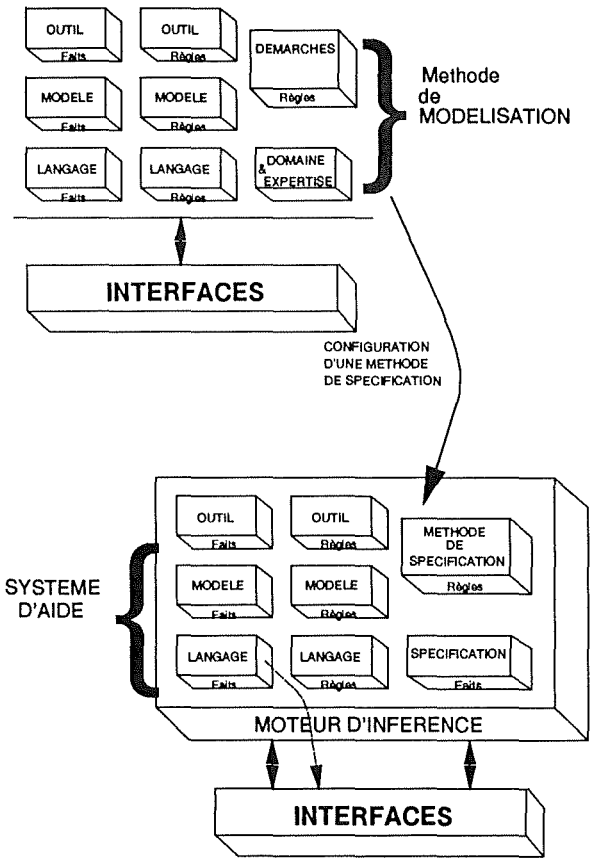


FIGURE 3.2

Par rapport à la manipulation de ces connaissances, nous avons également besoin de règles de contrôle vérifiant la cohérence entre les différentes bases : emploi de concepts identiques, surveillance de l'isomorphisme entre langages et modèles, unicité des concepts,...

Ces connaissances vont servir de base à l'élaboration de méthodes de spécification. Pour ce faire, nous avons besoin d'un outil permettant d'extraire et d'organiser les connaissances nécessaires à la création d'une méthode de spécification.

Pour exploiter cette méthode de spécification, nous avons besoin d'un système d'aide qui guidera l'analyste dans son travail de spécification.

CHAPITRE IV
CONFIGURATION D'UNE METHODE DE SPECIFICATION

SECTION 4.1 LA NOTION DE MODELE

Dans la pratique, les spécifications d'un SI s'expriment au moyen de langages particuliers graphiques et/ou textuels propres à la méthode de modélisation utilisée. Ces langages offrent en fait aux analystes la représentation de concepts aptes à définir formellement les différents aspects d'un SI.[4]

Pour éviter une trop grande confusion dans les travaux de spécification et pour permettre l'instauration de points de contrôle, les méthodes de modélisation regroupent généralement au sein de modèles les concepts relatifs à un même aspect du travail de spécification d'un SI; par exemple, l'aspect dynamique des traitements, l'aspect statique des flux d'informations, la définition des structures de 'données',...

Vu de cette manière, il est évident qu'un même concept peut appartenir à plusieurs modèles. La différence s'effectuera sur base de la représentation et des propriétés des concepts mises en oeuvre séparément dans chaque modèle.[5]

Toutes les contraintes sémantiques exprimées sur les concepts et relatives à l'aspect particulier mis en exergue dans un modèle sont également jointes à la définition des concepts.

Les démarches de la méthode de modélisation définissent une ou plusieurs (généralement une) mise en oeuvre des concepts et des contraintes dans le cadre particulier de chacun des modèles.

Cette vision de la méthode de modélisation ne va pas à l'encontre de nos définitions du CHAPITRE I Section 1.3.1. En ce qui concerne les méthodes de spécification, l'approche est plus délicate car la notion de modèle ne correspond plus parfaitement à notre définition. Mais avant d'en dire davantage, proposons d'une manière globale le point de vue relatif aux méthodes de spécification.

4 *Vue inverse de la projection MODELES vers LANGAGES*

5 *Vu de cette manière et en faisant l'hypothèse que chaque modèles possède une représentation unique pour chacun de ses concepts, la projection définie dans les sections précédentes se transforme en une bijection.*

Pareillement aux méthodes de modélisation, les méthodes de spécification regroupent les concepts afin d'éviter une trop grande confusion dans les travaux de spécification et de permettre l'instauration de points de contrôle. Plus particulièrement, les méthodes de spécification proposent une découpe de l'activité de spécification en différentes étapes. Comme nous l'avons fait remarquer plus haut, cette découpe ne poursuit pas un objectif de regroupement par rapport à un aspect particulier de la spécification d'un SI mais plutôt un objectif de regroupement par domaine du SI ou par progression linéaire du travail de spécification (vision pyramidale du travail de spécification); la figure 4.1 montre schématiquement ces trois approches.

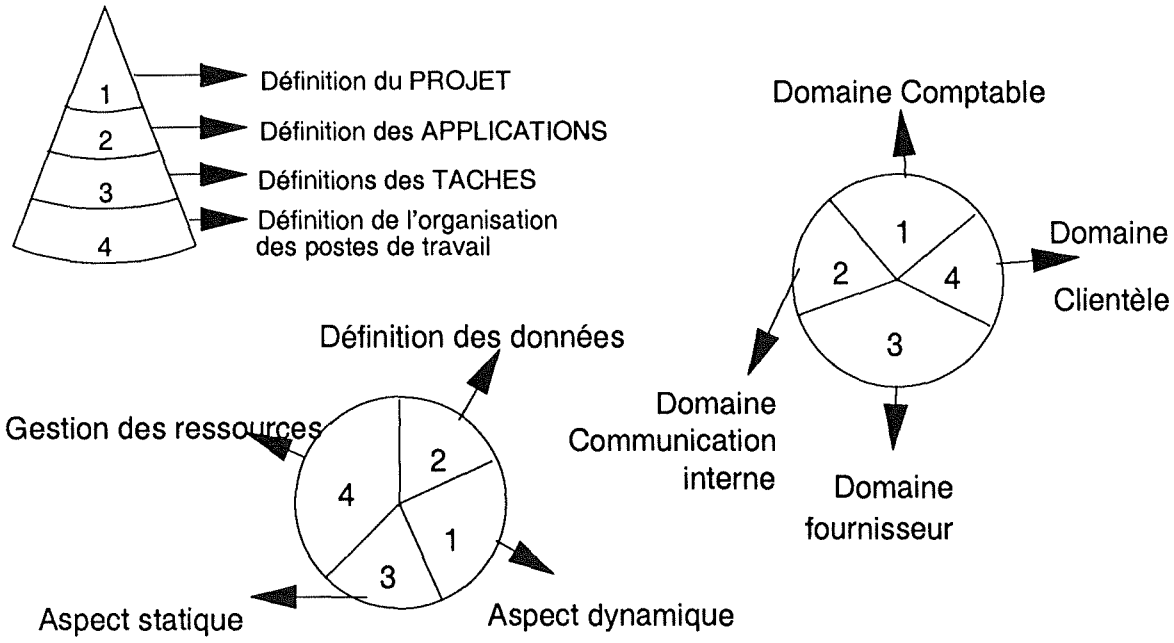


FIGURE 4.1

Plus explicitement, nous dirons que ces étapes sont définies par rapport à un objectif que les spécifications doivent satisfaire. Généralement, cet objectif, définition d'une étape exprime une limite d'exploitation des concepts offerts par la méthode de modélisation et une définition de la cohérence et de la complétude des spécifications dans leur ensemble.

Pour ce qui nous concerne, nous faisons l'hypothèse que l'objectif d'une étape définit un modèle de validation des spécifications par rapport à un sous-ensemble des concepts de la méthode de modélisation. Nous définissons la méthode de spécification comme comprenant l'ensemble de ces modèles.

Au niveau des langages, les concepts retenus dans les modèles de la méthode de spécification sont exprimés au travers des représentations définies au niveau de la méthode de modélisation.

Or, les langages utilisés au niveau de la méthode de modélisation sont définis par rapport aux modèles de cette méthode (Principe de la bijection définie ci-avant). Ce principe soulève le problème important de la relation devant exister entre méthode de modélisation et méthode de spécification. Afin de garder cohérente cette relation, nous devons admettre que la découpe en modèle effectuée au niveau de la méthode de spécification n'abolit pas la définition des modèles établies dans la méthode de modélisation mais au contraire doit la conserver au sein de ses propres modèles.

Cet état de fait nous permet de garder la méthode de spécification cohérente (et complète) par rapport à la méthode de modélisation, particulièrement au niveau des contraintes définies sur les concepts. En effet, nous avons affirmé plus haut que les modèles définis dans la méthode de modélisation contenait non seulement les concepts mais également les contraintes définies sur ces concepts et relatives à l'aspect particulier mis en exergue dans le modèle. Sans la conservation des modèles de la méthode de modélisation, la méthode de spécification ne permettait pas nécessairement la validation de ces contraintes.

De plus, si nous devions faire disparaître la définition des modèles telle qu'exprimées dans la méthode de modélisation, nous nous trouverions en état d'incohérence par rapport aux langages puisque nous aurions à notre disposition des représentations de concepts pour lesquels le concept en tant que tel n'existerait plus ou serait détourné de son sens premier.

En résumé, les modèles de la méthode de spécification organisent le travail de spécification en fonction d'objectifs particuliers définis pour chacun d'eux. Au sein de ces modèles, les

concepts utilisés s'organisent selon le schéma proposé au sein des modèles de la méthode de modélisation afin de permettre la validation des contraintes définies et afin de permettre leur représentation dans les langages associés.

Relativement aux modèles de la méthodes de spécification, nous devons également prévoir la définition de contraintes définissant les restrictions particulières d'exploitation des concepts retenus au niveau de chaque modèle.

Relativement aux démarches, notre objectif est d'arriver à associer à chacun des modèles de la méthode de spécification une démarche particulière; cette démarche est définie par rapport aux règles de 'pilotage' (Cfr. Section 3.2.2).

Dés lors, à notre avis, l'idée générale d'un outil d'aide à la spécification d'un SI doit être d'une part de permettre au chef de projet de définir et d'affiner précisément une méthode de spécification à l'aide de règles (principalement au niveau de ses modèles et de ses démarches) et d'autre part, pendant le travail de spécification de contrôler le déroulement des spécifications par rapport aux démarches de la méthode de spécification et de vérifier que les contraintes d'intégrité relatives aux concepts des modèles sont respectées.

Nous confirmons ainsi notre idée d'un système dual où la première partie concerne la création d'une méthode de spécification par rapport à la méthode de modélisation et la deuxième partie concerne le contrôle de l'organisation et la validation du travail de spécifications par rapport à la méthode créée. Nous appellerons dorénavant la première partie 'Interface de configuration' et la deuxième 'Interface de pilotage'.

SECTION 4.2 FORMALISATION DANS UN METALANGAGE

Avant de continuer plus avant dans l'étude de notre système d'aide à la spécification d'un SI, nous devons faire une légère digression. En effet, nous parlons sans cesse de concepts, de connaissances, de langages mais nous n'avons pas encore abordé le problème de la formalisation de toutes ces informations en un langage compréhensible par le moteur d'inférence de notre système.

A cet effet, nous avons décidé de prendre le langage **Z** [ABR78] comme métalangage d'expression des concepts d'une méthode de modélisation et par conséquent des concepts d'une méthode de spécification. Ce choix n'est pas exclusif et nous aurions pu choisir un langage de type **Entité - Relation - Attribut**. Les critères qui ont influencé ce choix reposent principalement pour la simplicité de ce langage et sa puissance d'expression. De plus, l'absence de hiérarchie dans la formalisation des concepts (pas de différence entre entité et attribut) permet à notre avis une meilleure *"mise à plat"* de la méthode de modélisation. Ce phénomène offre nous le pensons, une meilleure précision dans la définition des concepts, dans la compréhension des règles de contrôle et dans la définition des règles de pilotage.

Il est évident que le langage Z ne permet pas directement l'utilisation d'un moteur d'inférence; nous devons encore à ce stade effectuer une transcription des énoncés Z dans un langage supporté par un moteur d'inférence. Nous pensons cependant que le passage par un langage intermédiaire est une étape nécessaire afin de permettre une plus grande facilité de communication de la méthode et des concepts définis, et de rester indépendant des exigences techniques de la réalisation du système.

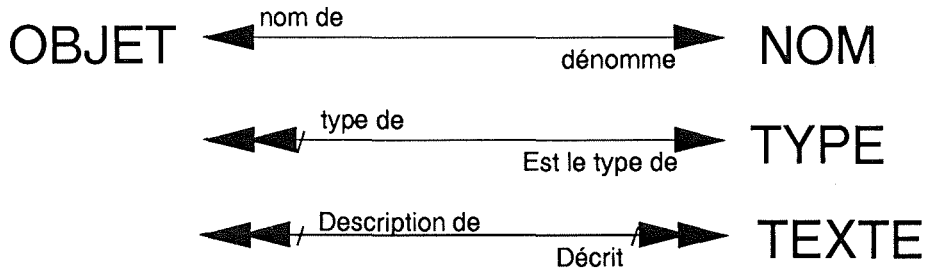
Actuellement, nos recherches se sont établies dans un contexte d'utilisation du langage PROLOG et de son moteur d'inférence comme support à la réalisation de notre système d'aide; toutefois, elles auraient pu s'envisager dans un contexte de type 'FRAMES' avec comme support des outils de style 'NEXPERT-OBJECT'.

Section 4.2.1 Le langage Z

Le principe de Z est de permettre la description des classes d'information appelées "ensembles d'entités" et la définition entre deux ensembles d'entités des relations binaires appelées "associations". Ces associations sont caractérisées par des sortes de fonctions qui précisent les rôles joués par les ensembles d'entités arguments des associations. Ces rôles correspondent généralement à des interprétations de ces relations dans le sens d'un ensemble source vers un ensemble cible. Ces définitions peuvent être complétées par des formules logiques pour exprimer des contraintes sur les ensembles ou sur les relations.

Exemples:

Représentation en Z de la structure d'un objet en terme de nom, de type et de description.



OBJET, NOM, TYPE, TEXTE sont les quatre ensembles d'entités introduits dans cet exemple, trois associations binaires les relient et déterminent ainsi six fonctions décrivant les rôles joués par les ensembles d'entités au sein de l'association. La première association exprime le fait qu'un objet doit avoir un et un seul nom et qu'à un nom ne peut correspondre qu'un et un seul objet. La deuxième association exprime le fait qu'un objet doit avoir un et un seul type et qu'à un type peut correspondre plusieurs objets. La dernière association exprime le fait qu'un objet peut avoir zéro, une ou plusieurs description et qu'à une description peut correspondre plusieurs objets.

1) Les ensembles d'entités

Un ensemble d'entité peut-être désigné par :

- un nom caractérisé par un identifiant formé de lettres majuscules;

Exemples: *PROCESSUS, ENTITE, MESSAGE, ENTIER, STRING;*

- une définition en extension;

Exemples: *{projet, application, phase, fonction};*

- des opérations ensemblistes d'union, d'intersection, de différence et de produit cartésien sur des ensembles d'entités;

Exemples:

$PROCESSUS \cup INTERFACE,$

$ENTITE \times ROLE \times \{(0,1),(1,1),(1,N),(0,N)\}$

- une définition en intention;

Exemples: $\{p \in PROCESS \mid card(Subparts\_are(p)) \leq 1\};$

- une combinaison des cas ci-dessus.

Exemples: $ENTITE \times ROLE \times \{(0,1),(1,1),(0,N),(1,N)\};$

Nous appelons un ensemble composite un ensemble d'entités formé par un produit cartésien où au moins un des ensembles est optionnel.

Exemples:

Soient $A = \{a_1, a_2, a_3\}, B = \{b_1, b_2\} \wedge C = \{c_1, c_2, c_3\}.$

L'ensemble $E = A \times B^{\circ} \times C$ *est un ensemble composite et par exemple,*

$(a_1, -, c_2) \in E, (a_3, b_2, c_3) \in E mais (a_2, b_2, -) \notin E.$

Actuellement, au niveau des opérations ensemblistes pour la définition des ensembles d'entités, notre étude se limite à l'emploi du produit cartésien (×) correspondant à un 'ET' logique et de l'union ∪ correspondant à un 'OU' logique. Les autres opérateurs (intersection et différence) présentent des difficultés au niveau de leur gestion et ne semblent pas offrir un intérêt pour la formalisation d'une méthode de modélisation.

Vu sous l'aspect de la gestion des ensembles d'entités au sein de notre système, la définition en intention d'ensemble d'entité s'effectuera par la définition d'un ensemble

générique et la définition de formules Z restreignant le domaine des valeurs accessibles pour cet ensemble; mais d'un point de vue pratique, nous émettons les mêmes remarques que pour les opérateurs ensemblistes d'intersection et de différence.

2) Les associations

La description d'une association est réalisée par :

- la désignation des ensembles d'entités;
- la définition précise des fonctions: nom, portée partielle ou totale sur l'ensemble source et cardinalité monovaluée ou multivaluée sur l'ensemble cible.

Représentation Graphique et Conventions d'identification des Fonctions

----------> ***fonction monovaluée totale***
--------/-> ***fonction monovaluée partielle***
---------->> ***fonction multivaluée totale***
--------/->> ***fonction multivaluée partielle***

Le nom d'une fonction monovaluée est un identifiant formé par des caractères minuscules et le nom d'une fonction multivaluée est un identifiant commençant par une lettre majuscule.

Nous devons également noter qu'une association n'est complètement définie que par la double définition d'une fonction allant d'un ensemble source vers un ensemble cible et de sa fonction inverse.

Les ensembles sources et cibles sont des ensembles d'entités tels que nous les avons définis dans la section précédente.

3) Les formules

Les formules sont définies par des expressions logiques obéissant aux règles classiques de l'algèbre et du calcul de la logique des prédicats du 1er ordre. Nous ne les rappellerons pas ici.

SECTION 4.3 LES OBJETS ET LES RELATIONS

Le but de la formalisation d'une méthode de modélisation est de constituer un ensemble d'énoncés qui définissent les connaissances relatives à cette méthode. Cette formalisation s'effectue dans le langage Z et établit la définition d'ensembles d'entités, d'associations binaires entre ces ensembles et de formules logiques exprimant les règles de la démarches, les règles de pilotage et les contraintes applicables sur les entités et les relations.

Typiquement, les ensembles d'entités font références à des concepts précis de la méthode de modélisation tandis que les associations font références à des liens existant ou pouvant exister entre ces concepts.

Exemples:

*Prenons les concepts de **'processus'** et de **'message'** dans le cadre de la méthode DSL/IDA. Nous pouvons définir les liens suivants entre ces deux concepts:*

Un processus peut générer un ou plusieurs messages.

Un message peut être généré par un ou plusieurs processus.

Ces connaissances nous sont fournies par la définition de ces concepts et de ces relations dans la méthode DSL/IDA. Nous pouvons formaliser ces 2 aspects sous la forme suivante:

*Nous avons donc deux ensembles d'entités, à savoir **PROCESSUS** et **MESSAGE**, qui représentent les concepts de processus et de message définis dans la méthode DSL/IDA. Nous avons également une association définie entre ces deux ensembles; elle est caractérisée par les fonctions **'Generates'** et **'Generated by'**, fonctions multivaluées et facultatives. Cette association est bien entendu également définie dans la méthode DSL/IDA.*

Dans le cadre de cette formalisation, nous avons défini une classification des ensembles d'entité en deux catégories distinctes. La première catégorie reprend les entités associées à un concept de la méthode de modélisation permettant la spécification d'une réalité effective de spécification. Nous appelons réalité effective de spécification un concept de la méthode dont l'identification passe par un nom et une description.

La deuxième catégorie reprend les ensembles regroupant les entité associées avec les entités définissant une propriété d'une réalité effective de spécification.

Les principes de formalisation mis en oeuvre dans le langage Z et la distinction faite entre les réalités effectives de spécification et les propriétés de ces réalités nous permettent de dégager les notions d'objet et de relation.

Section 4.3.1 Les objets

La notion d'objet s'associe directement avec la première catégorie d'entités définie ci-dessus; les réalités effectives de spécification. En effet, un objet correspond à une entité identifiée par un nom et une description. Cependant, ces deux propriétés ne sont pas suffisantes pour garder la référence à l'ensemble d'entités et nous avons donc ajouté la notion de type d'un objet.

Nous définissons le type d'un objet en regroupant dans une même classe tous les objets appartenant au même ensemble d'entité tel que défini en Z; soit toutes les entités relatives à un même concept de la méthode de modélisation.

La définition des types d'objet se dérive donc directement des différents ensembles d'entités associés aux concepts définissant des réalités effectives de spécification. Si cette dérivation est automatique, la définition des réalités effectives de spécification l'est moins. En effet, le responsable chargé de la formalisation de la méthode de modélisation doit décider d'accorder ou non le statut de réalité effective aux différents concepts proposés par la méthode et ce choix peut ne pas être toujours évident.

Exemples:

Prenons comme méthode de modélisation, la méthode IDA et dans cette méthode prenons en particulier le concept de 'message'. Nous définirons le type d'objet 'message' comme étant la classe des objets appartenant à l'ensemble d'entités MESSAGE définis dans l'exemple ci-dessus. Un objet de type 'message' correspondra à un élément de cette ensemble.

Soit les bons de commandes arrivant chaque matin dans une entreprise de vente par correspondance, ces bons peuvent selon un certain point de vue définir un élément de spécification relatif au concept de 'message' de la méthode IDA. Nous pouvons dire de même pour les factures envoyées aux clients, pour les rappels de paiement,...

Chacun de ces éléments de spécification définissent une occurrence d'entité dans l'ensemble MESSAGE. De ce fait, ces éléments définissent une occurrence d'objet de type 'message'.

Prenons maintenant le concept de ***condition****, nous pouvons l'associer au concept de message de la manière suivante:*

- un message peut être ou ne pas être émis suivant la valeur d'une certaine condition.

Nous avons ici un concept que nous pouvons nommer et décrire, toutefois, étant donné son contexte d'utilisation, nous pouvons le définir comme étant une propriété facultative du concept de message. Le choix d'établir le concept de ***condition*** *comme réalité effective de spécification et donc de définir le type d'objet* ***'condition'*** *appartient à l'expert chargé de la formalisation de la méthode de modélisation.*

Section 4.3.2 Les relations

La notion de relation s'associe, quant à elle, à la notion de fonction utilisée dans le langage Z pour la définition des associations entre ensembles d'entités.

Une relation définit un lien entre un élément de l'ensemble source et un élément de l'ensemble cible, ensembles précisés par la fonction définie dans le langage Z. Une relation s'identifie par l'identité des deux éléments qu'elle unit.

Nous définissons la notion de type de relation de la même manière que pour les objets. Le type d'une relation est issu de la fonction associée.

Exemples:

Reprenons nos concepts de ***message*** *et de* ***processus*** *ainsi que les fonctions suivantes:*

- Un processus peut générer un ou plusieurs messages.

- Un message peut être généré par un ou plusieurs processus.

La formalisation en Z nous donne:

*L'association se définit par les deux fonctions **'Generates'** et **'Generated by'** et par les ensembles d'entité **MESSAGE** et **PROCESSUS**. Les deux fonctions définissent respectivement les types de relation **'generates'** et **'generated by'**.*

Lors de la spécification des types de relation sur base des fonctions définissant les associations Z, nous pouvons rencontrer quelques difficultés. En effet, certaines fonctions peuvent se révéler inutiles dans le cadre du travail de spécification.

Dans notre exemple et relativement à la connaissance que nous avons de la méthode IDA, nous pouvons affirmer qu'il est important pour l'analyste de pouvoir spécifier l'association entre les ensembles **MESSAGE** et **PROCESSUS** à partir des deux fonctions; les deux types de relations sont donc nécessaires.

Toutefois, cela n'est pas toujours le cas; et dans le cadre d'une association entre un ensemble d'entités relatives à des réalités effectives de spécification et un ensemble d'entités relatives à une propriété, l'association n'est souvent intéressante que dans le sens 'concept vers propriété'; l'autre sens ne présentant qu'un intérêt pratique dans la validation de règles de contrôles. Inversement, les associations entre types d'objet seront généralement représentées par deux relations distinctes dépendantes l'une de l'autre.

Du point de vue de la gestion des relations, l'aspect obligatoire d'une fonction sera évacué de la définition de la relation et redéfini dans une règle de contrôle applicable sur la relation.

Le changement de vocabulaire que nous venons d'introduire par rapport au langage Z peut paraître lourd et inutile dans un premier temps. Cependant, nous verrons dans le cadre du chapitre suivant que ces notions se rattachent à des faits précis.

D'une manière générale, l'objectif poursuivi par ce changement de vocabulaire est premièrement de préciser différents domaines parmi les concepts de la méthode de spécification et deuxièmement de préparer la mise en place de l'interface de pilotage[6].

6 *Pour parler indifféremment d'objet et de relation, nous utiliserons la notion d'élément de spécification.*

SECTION 4.4 LES REGLES

Les règles sont formalisées en terme de formules Z exprimées sur la formalisation en Z des concepts de la méthode de modélisation. Ces formules définissent des contraintes d'intégrité applicables sur les concepts. Nous appellerons désormais ces règles des règles élémentaires de gestion.

A côté de ces règles élémentaires de gestion et pouvant également être définies sous forme de formule Z, nous avons les règles de pilotage qui définissent un éventail si possible exhaustif des règles de contrôle du travail de spécification du point de vue de son organisation (pas du point de vue de la validité des spécifications).

Section 4.4.1 Les règles élémentaires de gestion

Les règles élémentaires de gestion ne couvrent en fait qu'une partie des contraintes définies sur les concepts de la méthode de modélisation. En effet, pour des raisons pratiques, certaines contraintes seront directement immergées dans l'interface de pilotage.

Ces contraintes immergées correspondent à des contrôles propres à l'utilisation du langage Z comme métalangage et de l'emploi des notions d'objets et de relations. Ces contrôles sont principalement:

- *Validation de l'unicité des noms d'objet.*
- *Validation de l'appartenance d'un objet à un type donné (dans le cadre d'une relation).*

Les règles élémentaires de gestion ne comprennent pas non plus les contrôles relatifs aux spécificités des langages utilisés, à savoir l'appartenance d'une valeur à un domaine particulier, les règles d'orthographe pour les noms des différents concepts de la méthode,...

Section 4.4.2 Les règles de pilotage

Les règles de pilotage définissent des contraintes dans l'organisation du travail de spécification. Bien que dans certains cas, elles peuvent avoir le même énoncé que des règles élémentaires de gestion, il ne faut pas les confondre.

Exemples:

*Prenons la règle interdisant la spécification d'élément du SI relatifs à la notion de **message**.*

*Du point de vue des règles de contrôle, cela signifie que dans le cadre du modèle associé, la spécification d'un objet de type **message** n'est pas permise.*

*Du point de vue des règles élémentaires de gestion, cela signifie qu'il ne peut pas exister d'occurrence d'objet de type **message** dans l'ensemble des spécifications et cette contrainte doit être vérifiée dans tous les modèles de la méthode de spécification.*

Nous voyons donc qu'une règle exprimée de la même manière peut faire référence à des contraintes tout à fait différentes. Nous étudierons les règles plus en profondeur dans le chapitre suivant.

Section 4.4.3 Les paquets de règles

La configuration de la méthode de spécification s'effectue en plusieurs étapes dont la première consiste à nommer la méthode et dont la deuxième consiste à définir l'organisation et les objectifs des modèles.

Ces deux étapes terminées, il nous reste à définir l'organisation du travail de spécification au sein de chaque modèle de la méthode de spécification et relativement à tous les concepts des modèles de la méthode de modélisation. Cette étape doit s'effectuer par rapport aux règles élémentaires de gestion et aux règles de pilotage.

Le déroulement de cette étape s'effectue selon le schéma suivant:

I. Pour la méthode de spécification en général;

il faut définir les types d'objets et les types de relations utilisés dans la méthode;

(utilisation d'une règle élémentaire de gestion précisant l'interdiction).

il faut définir les types d'objet et de relations obligatoires;

(utilisation d'une règle élémentaire de gestion précisant l'obligation).

il faut définir les contraintes à appliquer sur les types d'objets et les types de relations.

(utilisation de règles élémentaires de gestion précisant l'unicité de certaines relations, la cardinalité de certains ensembles, les minimums et maximum de certaines décompositions,...)

II. Pour chaque modèle de la méthode de spécification;

- il faut définir les contraintes à vérifier en entrée du modèle;

(utilisation de règles élémentaires de gestion).

- il faut définir les contraintes à vérifier en sortie du modèle;

(utilisation de règles élémentaires de gestion).

- il faut définir les types d'objets et les types de relations utilisés dans le modèle;

(utilisation d'une règle de pilotage précisant l'interdiction).

- il faut définir les types d'objets et les types de relations obligatoires;

(utilisation d'une règle de pilotage précisant l'obligation).

- il faut définir l'organisation du travail de spécification par rapport aux objets;

(utilisation de règles de pilotage précisant l'ordre des saisies par rapport aux objets).

a. Pour chaque type d'objet obligatoire:

- il faut définir l'organisation du travail de spécification par rapport aux relations associées à cet objet;

(utilisation de règles de pilotage précisant l'ordre des saisies des différentes relations associées à cet objet)

- il faut définir les contraintes d'intégrité applicables;

(utilisation de règles élémentaires de gestion précisant les contraintes à respecter pour la création ou la modification de l'objet)

- il faut définir les représentations utilisées (point de vue du langage).

b. Pour chaque type de relation obligatoire:

- il faut définir l'organisation du travail de spécification par rapport à chaque élément des ensembles cible et source de cette relation;

(utilisation de règles de pilotage précisant l'ordre des saisies de chaque élément des ensembles cible et source de la relation)

- il faut définir les contraintes d'intégrité applicables;

(utilisation de règles élémentaires de gestion précisant les contraintes à respecter pour la création ou la modification de la relation)

- il faut définir les représentations utilisées (point de vue du langage).

Etant donné la relation existant entre les langages et les modèles de la méthode de modélisation et étant donné que nous gardons le schéma d'organisation de ces modèles au sein des modèles de la méthode de spécification, nous pouvons dériver systématiquement les structures de langages nécessaires à la représentation des concepts.

En résumé, nous avons une méthode de modélisation formalisée dans laquelle les notions d'objets et de relations sont précisées et un ensemble de règles de pilotage et nous avons à notre disposition une marche à suivre pour configurer une méthode de spécification.

Le déroulement de cette marche à suivre fournit

Pour la méthode:

- *un ensemble de modèles ;*
- *un ensemble[7] de règles élémentaires de gestion associé à chaque objet et relation;*

Pour chaque modèle:

- *un ensemble de règles élémentaires de gestion définissant les contraintes en entrée;*

7 Cet ensemble définit les contraintes générales applicables sur l'ensemble des spécifications; ces contraintes seront héritées systématiquement par les modèles.

- *un ensemble de règles élémentaires de gestion définissant les contraintes en sortie;*
- *un ensemble de règles élémentaires de gestion associé à chaque type d'objet;*
- *un ensemble de règles de pilotage associé à chaque type d'objet;*
- *un ensemble de règles élémentaires de gestion associé à chaque type de relation.*
- *un ensemble de règles de pilotage associé à chaque type de relation.*
- *un ensemble de règles relatives à l'organisation du travail de spécification au sein du modèle.*

Nous appellerons désormais les ensembles de règles des paquets de règles; chaque paquet faisant référence à un objet particulier, une relation particulière ou un aspect particulier de la méthode. L'organisation des paquets de règles et le paquet relatif à l'organisation du travail de spécification définissent en fait la démarche associée à la méthode de spécification.

SECTION 4.5 LES METAREGLES

Il est évident que la configuration d'une méthode de spécification ne s'effectue pas sans contrôle de cohérence ni complétude; en effet, tout amalgame de règles élémentaires de gestion et de règles de pilotage ne définit pas une méthode de spécification correcte.

Le principe des métarègles est identique au principe des règles élémentaires de gestion; cependant, la validation de ses métarègles ne s'effectue pas sur la saisie de spécification mais sur la définition d'une méthode de spécification.

Dans le même ordre d'idée, nous pourrions définir des métarègles de pilotage; toutefois, étant donné la définition des notions d'objet et de relation, la définition d'une méthode de spécification peut toujours s'effectuer selon le même schéma quel que soit la méthode de modélisation originelle.

Les métarègles assurent la cohérence des règles élémentaires de gestion et de pilotage selon les trois points de vue suivants:

- ***L'exclusion:*** *une règle est en opposition avec une autre règle; la vérification d'une des règles entraîne forcément l'échec de la seconde.*

- ***L'inclusion:*** *la restriction imposée par une règle est contenue dans la restriction imposée par d'autres règles.*

- ***L'union:*** *la vérification d'une règle n'est valable que conjointement avec la vérification d'autres règles.*

Il nous faut ajouter que la vérification des métarègles ne s'applique pas seulement au niveau d'un paquet de règles mais au niveau de toute la méthode de spécification.

La cohérence d'une méthode de spécification n'est cependant pas assurée à 100% par le mécanisme des métarègles, il faut donc rester critique par rapport à la méthode de spécification et si possible la soumettre à un test préliminaire. Ce test permettra de valider la méthode par rapport à sa définition première.

L'amélioration des validations effectuées par les métarègles passe par une formalisation stricte et la création d'une typologie précise des règles. Ces deux aspects ont été développés par L. Alvares dans sa thèse [ALVA88] et seront envisagés sous un angle complémentaire dans la SECTION 5.3 LA NOTION DE REGLE.

CHAPITRE V PILOTAGE DES SPECIFICATIONS

SECTION 5.1 LA NOTION D'EVENEMENT

La validation des spécifications par rapport aux règles élémentaires de gestion de la méthode de spécification doit s'effectuer à certains moments de la saisie des spécifications.

Ces moments particuliers du travail de spécification correspondent à des points statiques préalablement définis. Ces point statiques sont intégrés dans l'interface de pilotage relativement à des événements particuliers issus de la manipulation du système d'aide à la spécification.

Actuellement, cinq types d'événements particuliers ont été définis; il s'agit des événements relatifs à l'entrée dans un modèle, à la sortie d'un modèle, à la fin de la spécification d'un objet, à la fin de spécification d'une relation et à la fin du travail de spécification.

A chacun de ces événements, nous pouvons associer un paquet de règles défini dans la méthode de spécification; le paquet de règles associé spécifiquement à la méthode s'associe à l'événement de fin du travail de spécification, le paquet de règles associé spécifiquement à l'entrée dans un modèle et le paquet associé spécifiquement à la sortie d'un modèle s'associe respectivement à l'entrée et à la sortie d'un modèle, finalement, les paquets associés à chaque objet et à chaque relation dans le cadre particulier de chaque modèle s'associent respectivement à la fin de spécification d'un objet et à la fin de spécification d'une relation.

En fait, l'occurrence d'un événement d'un certain type dans un contexte particulier déclenche la validation du paquet de règles associé à l'événement dans le cadre de ce contexte. Pratiquement, à chaque type d'événement fera référence un ensemble d'événements particuliers définis par rapport à la méthode de spécification.

Exemples:

Soit dans le cadre de la méthode de modélisation IDA, une méthode de spécification Méthode-M comprenant deux modèles (M1 et M2), les objets 'processus' et 'message' et la relation 'generates'.

Le type d'événement 'Fin du travail de spécification' fera référence à l'ensemble {'Fin du travail de spécification dans la méthode Methode-M'}.

Les types d'événement 'Début et Fin de modèle' feront référence à chacun des deux modèles de la méthode de spécification.

Les types d'événement 'Fin de spécification de l'objet' et 'Fin de spécification de la relation' feront respectivement référence aux ensembles suivants:

{'Fin de spécification de l'objet process dans le modèle M1', 'Fin de spécification de l'objet message dans le modèle M1', 'Fin de spécification de l'objet process dans le modèle M2', 'Fin de spécification de l'objet message dans le modèle M2'}

{'Fin de spécification de la relation generates dans le modèle M1', 'Fin de spécification de la relation de la relation generates dans le modèle M2'}

Relativement aux règles de pilotage, la chose est plus ardue. Toutefois, il semble que nous pouvons garder la notion d'événement et créer simplement de nouveaux types propres à ce genre de règles.

Dés lors, nous pouvons dégager principalement les types sur bases des événements suivants: début d'un modèle, début d'une création, modification, annulation ou consultation de spécification d'un type d'objet ou d'un type de relation, début d'une fonction particulière, fin d'un paquet de règles élémentaires de gestion évalué à vrai, fin d'une règle élémentaire de gestion évaluée à faux.

SECTION 5.2 L'INTERFACE

Nous allons dans cette section étudier les différents services que doit proposer un système d'aide à la spécification tel que nous l'envisageons.

Nous étudierons respectivement le design général de l'interface, les modes conversationnels possibles, les contrôles et validations de saisies, l'intégration d'une conduite automatique et la chronologie du travail de spécification.

Section 5.2.1 Le design général de l'interface

Le design général de l'interface de pilotage se définit autour de la notion d'écran, de fenêtre et de champ de saisie. Un écran est composé de fenêtres et une fenêtre comprend un ou plusieurs champs de saisie. Les notions d'écran, de fenêtre et de champ de saisie sont physiquement indépendantes des concepts techniques mis en oeuvre dans la conception d'une interface.

Dans notre optique, un écran correspond à la spécification d'un objet, une fenêtre porte sur la spécification d'une relation. Un champ porte sur la spécification d'un élément de l'ensemble cible ou source d'une relation (limité à une valeur unique par champ).

Ces associations définissent donc une hiérarchie dans le travail de spécification. Cette hiérarchie implique que la définition d'une relation ne peut s'effectuer qu'à partir de la spécification d'un objet. Plus spécifiquement, notre objectif est de déduire et de limiter à partir du type de l'objet en spécification, les types de relation pouvant être spécifiés; le type de l'objet et l'objet lui-même devant participer à l'ensemble source de la relation.

A la clôture de chacun de ces éléments (écran, fenêtre, champ de saisie) correspond une validation de la saisie effectuée; la clôture de l'élément étant permise si la saisie vérifie la validation.

La clôture d'un champ de saisie entraîne la vérification de règles complètement immergées dans l'interface de pilotage. Ces règles correspondent par exemple à la vérification de l'unicité d'un nom d'objet, à la correspondance entre un nom d'objet et le type exigé pour cet objet dans le cadre d'une relation...

En plus de ces règles, l'interface vérifie également que la valeur introduite vérifie les règles syntaxiques définies dans les langages de la méthode; par exemple l'appartenance de la valeur à un domaine, la mise en forme (toute majuscule),...

La clôture d'un champ de saisie correspond au passage à un autre champ de saisie.

La clôture d'une fenêtre produit une occurrence d'un événement du type lié à la fin de spécification d'une relation; la nature de la relation est donnée par le nom de la fenêtre fermée.

La clôture d'une fenêtre correspond à une fonction bien précise définie au niveau de l'interface et entraîne la clôture des champs de saisie associés à la fenêtre. Si la fenêtre ne comporte qu'un seul champ de saisie alors la clôture du champ de saisie correspond à la clôture de la fenêtre.

La clôture d'une fenêtre correspond à la fin de la spécification d'une relation, si l'ensemble cible de cette relation utilisait dans sa définition un type d'objet, alors notre hypothèse à ce propos est de prévoir le contrôle d'existence de cet objet. Dans l'affirmative, le type de l'objet référencé doit correspondre au type de l'objet exigé par la relation; dans la négative, le choix est donné soit de créer l'objet, soit de laisser la création en suspens.

Du point de vue de la cohérence des spécifications, la première solution est la plus sage, cependant cette création doit être permise par les règles de pilotage définie au niveau du modèle en cours. Dans le cas où les règles de pilotage ne permettrait pas la définition de ce type d'objet, il faudrait normalement refuser la spécification de cette relation.

La clôture d'un écran produit une occurrence d'un événement du type lié à la fin de spécification d'un objet; le type de l'objet est donné par le nom de l'écran clôturé.

La clôture d'un écran correspond à une fonction bien précise définie au niveau de l'interface et entraîne la clôture des fenêtres et indirectement des champs de saisie associés à l'écran. Si l'écran ne comporte qu'une seule fenêtre alors la clôture de la fenêtre correspond à la clôture de l'écran.

D'une manière générale, la hiérarchie que nous venons de définir sera toujours respectée même par rapport à l'identification d'un objet (Définition de son nom et de sa description).

Le type d'un objet doit présenter la particularité de pouvoir être déduit directement de la fonction déclenchée par l'analyste. Ce phénomène doit être prévu afin d'améliorer l'ergonomie des écrans en fonction du type d'objet demandé. De plus, l'aide à apporter à l'analyste dépend essentiellement du type de l'objet à spécifier, c'est pourquoi la définition de fonctions particulières propres à la spécification de chaque objet nous semble la mieux adaptée; la définition de ces fonctions provenant de la particularisation d'une fonction générique pour chaque type d'objet défini dans la méthode de spécification[8].

Section 5.2.2 Les types d'interface

La notion de type d'interface est définie pour préciser le contexte introduit par le mode conversationnel utilisé par l'analyste dans le cadre de son travail.

Nous définissons 4 modes conversationnels, à savoir la consultation, la création, la modification et l'exécution de commandes ponctuelles.

1) Les interfaces de consultation

Les interfaces de consultation sont les interfaces les plus simples, elle ne provoquent aucune interaction avec l'analyste excepté au niveau de commandes de défilement des données.

8 *Excepté pour les types d'objet interdits par une règle élémentaire de gestion employée au niveau de la méthode.*

Le principe de ces interfaces est de présenter à l'analyste le contenu de la base de connaissances contenant les spécifications du SI selon certaines modalités de présentation définies d'une part au niveau de la méthode de spécification et d'autre part au niveau des fonctionnalités de l'interface.

Les modalités de présentation définies au niveau des fonctionnalités de l'interface concernent le mode de présentation; par exemple présentation sous forme de liste d'objet ou de relation, présentation sous une forme graphique, présentation ponctuelle d'un objet ou d'une relation.

La précision de ces modalités dépend essentiellement de la puissance que l'on veut donner aux interfaces de consultation et sont indépendantes de toute méthode. La puissance des interfaces est définie par divers critères; à savoir, la richesse des modes de présentation, l'instauration de critère de sélection dans le cadre de liste, la précision des graphiques,...

Les modalités de présentation définies au niveau des méthodes de spécifications sont contenues dans la base de connaissances attachées aux langages de la méthode.

Ces modalités définissent en fait les caractéristiques de mise en oeuvre des modalités définies ci-dessus. Elles définissent les textes de présentation, les formes graphiques à utiliser, les couleurs de présentation, le positionnement dans l'écran,...

Du point de vue d'un éventuel contrôle de la consultation, il n'apparaît pas nécessaire de la limiter par rapport à un modèle de la méthode de spécification, ni de se limiter au design général de l'interface. Au contraire, dans le cadre d'une aide à l'analyste, il est important de laisser l'accès libre à l'ensemble des spécifications et de pouvoir présenter les spécifications dans une multitude de formats échappant aux notions d'écran, de fenêtre et de champ de saisie.

2) Les interfaces de modification

Les interfaces de modification sont des interfaces de consultation dans lesquelles les interventions de l'analyste sont autorisées. Ces interfaces gardent les caractéristiques

des interfaces de consultation mais sont limitées au niveau des modes présentations; le mode principal sera la présentation ponctuelle d'un objet et de ses relations; accessoirement, et en fonction de la puissance de ces interfaces, la présentation graphique pourra être disponible.

Attention, la modification concerne l'altération de valeurs contenues au niveau des données du SI. Notre vision ne conçoit pas la définition d'une nouvelle relation relative à un objet déjà existant comme une modification de l'objet mais bien comme la création d'une relation.

Les libertés laissées à l'analyste dépendent de modalités définies au niveau de la méthode de spécification et au niveau des fonctionnalités de l'interface.

Les modalités définies au niveau de l'interface concernent le mode de présentation (Voir ci-dessus) et les possibilités de modification offertes. En effet, les modifications possibles peuvent être ponctuelles, limitées à l'objet en lui-même ou peuvent être héritées indirectement par d'autres objets ou relations suite à l'exploitation des définitions de relations entre objets.

Exemples:

La modification du nom d'un objet doit normalement entraîner la modification de l'identification de cet objet dans toutes ses relations.

Dans la méthode IDA, la modification du niveau d'un processus doit normalement entraîner la modification du niveau de ses processus 'pères' et processus 'fils'.

Les modalités définies au niveau de la méthode de spécification concernent d'une part les caractéristiques des modes de présentation et sont contenues dans la base de connaissance relative aux langages (Cfr 1) Les interfaces de consultation) et d'autre part la définition des contrôles à appliquer sur les modifications effectuées.

La modification doit se concevoir relativement à un modèle de la méthode de spécification. Il ne doit en effet pas être permis de modifier certains aspects d'un objet ou d'une relation si cet aspect n'est pas sous le contrôle du modèle en cours.

Le contrôle des modifications s'effectue par rapport aux paquets de règles associés à l'élément modifié et ce dans le cadre du modèle en cours en particulier et de la méthode en général. La gestion de ces contrôles nous oblige ici à utiliser le design général de l'interface et à travailler dans un contexte d'écrans, de fenêtres et de champs de saisie.

La modification s'effectue toujours au niveau d'un champ de saisie et la hiérarchie des contrôles est maintenue c'est à dire que la validation de la modification s'effectuera via les contrôles associés au champ modifié, via les contrôles associés à la fenêtre contenant ce champ et via les contrôles associés à l'écran comprenant la fenêtre.

La dynamique introduite au niveau des règles de pilotage doit ici être oubliée. En effet, nous pensons que la modification est une action ponctuelle sous le contrôle total de l'analyste; il nous semble donc inutile de gérer une dynamique du travail de spécification dans le cadre des modifications.

3) Les interfaces de création

Les interfaces de création sont des cas particuliers des interfaces de modification. Les modalités de présentation sont identiques ainsi que la gestion des contrôles de saisies. La différence réside dans la possibilité de définir des règles de pilotage précisant l'organisation du travail de spécification.

Ces règles contenues dans le paquet associé à l'organisation du travail de spécification d'un modèle définissent l'enchaînement des écrans dans une session de travail, l'enchaînement des fenêtres dans un écran et l'enchaînement des champ de saisie dans une fenêtre.

Ces règles ne sont pas obligatoires ou peuvent être limitées à certains types d'objet ou de relation. Pour les éléments de spécification qui ne seraient pas couverts par de telles règles, l'organisation du travail de spécification est laissée au libre arbitre de l'analyste.

4) Les interfaces d'annulation

L'annulation doit s'envisager différemment selon l'élément de spécification visé par l'annulation. L'annulation d'une relation entre un objet et une propriété peut s'envisager comme une modification particulière où la valeur initiale de la propriété est remplacée par la valeur absence. La seule validation à effectuer consiste à vérifier que cette valeur est admise pour cette propriété dans le modèle en cours.

L'annulation d'une relation entre objets doit normalement vérifier les règles définies par rapport au type de la relation et aux types d'objet mis en présence dans la relation. Cette annulation doit donc contrôler si l'annulation de la relation inverse est admise et prendre en charge cette deuxième annulation.

L'annulation d'un objet doit vérifier l'annulation de toutes les relations définies pour cet objet; seulement, ces annulations ne doivent pas vérifier les règles définies par rapport au type de la relation ni par rapport au type de l'objet. Cependant, ces annulations doivent vérifier si l'annulation des relations inverses des relations entre objet sont admises et prendre et charge ces annulations indirectes.

L'annulation de l'objet en soi ne doit vérifier aucune contrainte particulière par rapport au type d'événement 'Fin de spécification d'un objet'.

5) Les interfaces de commande

Les interfaces de commande sont relatives à l'installation de fonctionnalités particulières au sein de l'interface de pilotage; Nous pensons par exemple à la génération de rapport, au déclenchement de contrôles sporadiques par rapport à une ou plusieurs règles élémentaires de gestion, à la mise en format de données afin de permettre leur utilisation par des outils extérieurs,...

Les interfaces de commande concernent principalement l'environnement de l'interface de pilotage et leur mise en place reste largement indépendante de toute méthode de spécification. Toutefois, la mise en place d'une commande de validation sporadique d'une ou plusieurs règles doit prendre ses références dans la méthode de

spécification et doit veiller à vérifier les métarègles de configuration d'une méthode afin de ne pas permettre la validation d'une règle incohérente avec le modèle en cours et la méthode définie.

Les interfaces de commande peuvent se concevoir selon des perspectives différentes du design général des interfaces et à l'exception du cas cité dans le paragraphe précédent, elles ne doivent normalement pas respecter les contraintes exprimées par les modèles.

Section 5.2.3 Chronologie du travail de spécification

Dans sa thèse [ALVA88], L. ALVARES décrit formellement les différents aspects de la formalisation d'une méthode de modélisation. Dans sa description, il définit démarches et étape d'une démarche de la manière suivante:

> *"Une démarche dans le cadre d'une méthode peut être (ainsi) caractérisée par une séquence de modèles et une étape de démarche est la transition entre deux modèles successifs."*

Que ce soit au niveau des méthodes de modélisation ou au niveau des méthodes de spécification, cette définition ne nous convient qu'à moitié. En effet, une vision linéaire de l'enchaînement des étapes par rapport aux modèles définis est trop restrictive par rapport aux nécessités du travail de spécification et par rapport à la définition des modèles.

Par exemple, par rapport au travail de spécification, il arrive souvent qu'un aspect particulier du SI doive être retravaillé en fonction de particularités découvertes dans l'analyse d'autres aspects. Nous pensons que les différents aspects d'un SI vu par une méthode de modélisation ou de spécification ne sont pas totalement indépendants par rapport au travail d'analyse. De ce fait, la fin du travail de spécification par rapport à un aspect du SI se produit lors de la fin du travail de spécification de tous les aspects et non pas lors du passage de la spécification de cet aspect vers la spécification d'un autre aspect.

Par rapport à la définition des modèles, nous sommes convaincus qu'au niveau des méthodes de modélisation, l'organisation des modèles est indépendante d'une séquence

quelconque. Par exemple, l'analyse des données ne précède pas nécessairement l'analyse des traitements et inversement. Notre idée est que bien souvent, le travail s'effectue dans une sorte de parallélisme en fonction des besoins de l'analyste et de l'organisation.

La notion de séquence introduite dans la définition de L. ALVARES est donc trop limitative et doit être remplacée au profit d'une notion ensembliste. Nous arrivons donc à la définition suivante :

> Une démarche dans le cadre d'une méthode est caractérisée par un ensemble de modèles et une étape de démarche est la transition entre deux modèles.

Toutefois, la notion de séquence peut être maintenue par l'intégration dans chaque modèle de règles de contrôles définissant les conditions préalables et nécessaires à l'entrée dans ce modèle.

> **Exemples:**
>
> *Soit les modèles ordonnés M1, M2, M3;*
> *M1 n'a pas de condition d'entrée et a pour objectif O1;*
> *M2 a O1 comme condition d'entrée et O2 comme objectif;*
> *M3 a O2 comme condition d'entrée et O3 comme objectif avec O3 représentant le stade final des spécifications.*

D'un point de vue pratique, nous devons encore assurer la cohérence des spécifications d'un modèle à l'autre. Pour ce faire, nous proposons d'associer chaque élément de spécification aux modèles dont il est originaire ou qui a autorisé sa modification.

Par rapport à cette association, plusieurs attitudes sont possibles:

- Soit l'association s'effectue au niveau même de l'élément de spécification et elle établit un historique du travail de spécification. Dans ce cas, l'association peut soit être utilisée dans l'implémentation d'une gestion automatique des contrôles du travail de spécification par rapport aux différents modèles.

- Soit l'association est définie au niveau du modèle et est utilisée à des fins restrictives c'est à dire qu'un élément de spécification ne peut être défini ou modifié que dans un seul modèle.

A notre avis, la solution la plus riche mais aussi la plus difficile à réaliser est de réaliser l'association par rapport aux éléments de spécification c'est à dire de créer un historique de la création et de la modification d'un élément de spécification en référence des modèles où ont été effectuée ces modifications et créations afin de permettre la validation des règles élémentaires de gestion associées au type de l'élément de spécification et aux modèles présents dans cet historique.

Exemple:

Soit les modèles M1, M2, M3;
M1 permet la spécification du type d'objet O1, M2 la spécification des types d'objet O1 et O2, M3 la spécification des types d'objet O1, O2 et O3;

Soit l'objet O de type O1 créé dans le modèle M1 et modifié dans le modèle M3.

Toute modification introduite au niveau du modèle M2 sur les spécifications effectuées dans les autres modèles devra non seulement vérifier les règles associées au modèle M2 mais aussi les règles associées aux modèles M1 et M3.

Section 5.2.4 Validation et contrôles des spécifications

Comme nous l'avons déjà mentionné dans la SECTION 4.4, la validation des spécifications s'effectue par rapport aux règles élémentaires de gestion, les règles de pilotage interviennent quant à elles au niveau de l'organisation du travail de spécification.

Précédant la validation des règles élémentaires de gestion, nous avons évidemment certains contrôles qui sont effectués relativement aux contraintes définies dans les langages et aux règles immergées dans l'interface de pilotage. Les contraintes de langages correspondent à la définition des règles syntaxiques (type de caractère: alphabétique ou numérique, structure de codification,...), des domaines de valeurs, etc. Les règles immergées correspondent à la définition des contraintes établies sur les notions d'objets et de relation; par exemple, unicité des noms d'objet, contrôle par rapport au caractère mono- ou multivalué d'une relation, contrôle d'existence, adéquation de type dans une relation,...

Comme nous le savons déjà, les règles élémentaires de gestion sont groupées en paquet défini par rapport à la méthode de spécification, à l'entrée et à la sortie de chaque modèle, à chaque type d'objet et à chaque type de relation.

La vérification des règles élémentaires de gestion est déclenchée par la génération d'une occurrence d'un événement d'un type donné dans le cadre d'un modèle particulier du travail de spécification et relativement à un type donné d'élément de spécification. Les règles soumises à la validation sont contenues dans le paquet associé à l'élément en cours de spécification, aux modèles présents (en outre le modèle en cours) dans l'historique de cet élément et relativement au type d'événement rencontré.

Comme nous l'avons dit dans la Section 5.2.1, la fin de la spécification d'un élément n'est effective que si la vérification des règles est positive. Dans le cas contraire, plusieurs solutions se proposent à nous. Premièrement, seule une mention indiquant l'erreur est présentée à l'analyste. Cette solution est la plus simple est ne présente aucune caractéristique particulière.

Deuxièmement, il est également fait mention de l'erreur mais en plus une solution est proposée en guise de conseil; aucune autre action n'est engagée.

Troisièmement, il est fait mention de l'erreur et le conseil est donné mais en plus, le choix est donné à l'analyste d'effectuer la correction sur base de la solution proposée par le système. Selon le choix de l'analyste, le système effectuera ou non la correction automatique de l'erreur ou si cette correction n'est pas possible sans intervention de l'analyste, le système passera dans l'environnement permettant la correction de l'erreur.

Quatrièmement, il est fait mention de l'erreur mais la correction ou le passage dans l'environnement de correction s'effectue automatiquement sans laisser la possibilité à l'analyste la possibilité du choix.

D'une manière générale, nous ne prendrons pas parti pour une solution particulière, notre objectif est de permettre le choix de la solution lors de la configuration de la méthode de spécification. En effet, nous pensons qu'une même règle élémentaire de gestion peut s'envisager dans le cadre des quatre scénarii que nous venons de définir et que le choix d'un

scénario particulier doit s'effectuer au moment de la configuration de la méthode de spécification en fonction du type de règles, des modèles et de leurs concepts, des connaissances de l'analyste, de son expérience, du SI à spécifier et d'autres facteurs propres à l'organisation.

Le mode de correction d'une erreur doit donc s'envisager comme paramètre des règles élémentaires de gestion et définit en fait un ensemble de règles de pilotage. Dans ce contexte, l'installation d'un système de diagnostic complexe qui analyserait l'ensemble des erreurs commises et fournirait un ensemble de solutions peut s'envisager indépendamment de l'interface de pilotage. Ce système disposerait de ces propres règles de pilotage afin de diriger la correction des erreurs d'une manière cohérente.

En effet, actuellement, l'ordre de validation des règles élémentaires de gestion au sein d'un paquet n'est géré par aucun mécanisme, l'ordre de vérification est l'ordre défini lors de la création du paquet. De ce fait, les erreurs sont seulement détectées ponctuellement.

Vu sous cet angle, on imagine facilement comment un tel système peut devenir ennuyeux à utiliser. De plus, la fragmentation des erreurs ne permet pas toujours à l'analyste de situer l'entièreté de son erreur et donc ne lui donne pas les moyens d'apprendre à éviter ces erreurs.

Section 5.2.5 Pilotage effectif des saisies

Notre volonté par rapport à la définition des paquets de règles de pilotage est de laisser un maximum de liberté au configurateur de la méthode de spécification bien que nous ayons déjà émis plusieurs restrictions à ce sujet.

La première de ces restrictions provient de l'organisation de l'interface de pilotage en terme d'écran, de fenêtre et de champ de saisie et de la hiérarchie existant entre ces concepts. Le choix pour cette hiérarchisation a été effectué afin de diminuer le nombre et la complexité des règles élémentaires de gestion et de pilotage en installant un contrôle des accès à la spécification des relations.

La deuxième restriction concerne l'organisation du travail de correction des erreurs. En effet, nous considérons les erreurs comme des exceptions au travail de spécification, et il nous paraît plus cohérent de gérer ces exceptions de manière systématique et dirigiste même si le choix du degré de dirigisme est laissé au configurateur.

La troisième restriction n'est pas affirmée mais nous semble nécessaire, elle concerne la mise en place de création d'objet dans le cadre de la spécification d'une relation. En effet, les associations définies en Z peuvent relier deux ensembles d'objets; dés lors la spécification d'une relation de ce type, l'objet de l'ensemble source est connu puisqu'il a permis la spécification de la relation mais l'objet de l'ensemble cible n'est pas encore nécessairement connu.

Dans le cas où l'objet n'est pas connu, il nous semble plus satisfaisant de garder le même principe que pour les erreurs et de considérer ce cas comme une exception. Dans ce contexte, il nous paraît également plus cohérent de gérer cette exception de manière systématique et dirigiste. Malheureusement dans ce contexte-ci, le choix du degré de dirigisme n'est plus laissé au configurateur mais est directement présent dans l'interface. Simplement, nous devons permettre à l'analyste de refuser la création de l'objet cible mais dés lors la spécification de sa relation ne peut plus être acceptée et doit être modifiée ou annulée.

A part ces trois restrictions à la liberté du configurateur de la méthode de spécification, les autres règles de pilotage lui sont offertes sans obligation simplement pour lui permettre de définir l'environnement et l'organisation du travail du (des) analyste(s).

L'organisation du travail de spécification doit s'envisager sur trois niveaux distincts; primo dans le cadre d'un modèle et relativement aux types d'objet, secundo dans le cadre d'un type d'objet et relativement aux types de relation et tertio dans la cadre d'un type de relation et relativement aux éléments de la relation.

Relativement au troisième niveau, l'organisation est limitée à la définition de l'ordre des saisies dans les différents champs. Pour les deux autres niveaux, l'organisation du travail de spécification peut se concevoir de deux manières différentes: soit horizontalement soit verticalement.

Horizontalement, la spécification se fait par étapes successives. La première étape défini un minimum de spécification exigé; ce minimum étant défini par type d'objet. Ensuite, l'ensemble des spécifications déjà effectuées est passé en revue afin d'obtenir les spécifications exigées par la deuxième étape; et ainsi de suite jusqu'au moment où les spécifications vérifient l'objectif du modèle.

Verticalement, la spécification se fait ponctuellement et complètement objet par objet; l'analyste doit terminer la spécification d'un objet de manière complète par rapport au modèle en cours.

Ce type d'organisation n'est malheureusement possible que par rapport à des spécifications obligatoires et unitaires. Dans le cadre des types d'objet et relativement aux relations multivaluées, ce type d'organisation présente de grandes lacunes.

Notre hypothèse relativement à ces lacunes est de concevoir un système de saisie cyclique où l'option est laissée à l'analyste de quitter un cycle de spécification pour passer dans le cycle suivant après éventuellement un contrôle sur l'exécution du caractère obligatoire de la spécification. Bien évidemment, les cycles se définissent par rapport à chaque type d'objet et non pas pour l'ensemble des types.

SECTION 5.3 LA NOTION DE REGLE

Dans la SECTION 4.4, nous avons étudié la notion de règles sous l'angle des besoins de l'interface de configuration. Dans cette section, nous allons l'affiner par une étude établie à partir des besoins de l'interface de pilotage.

Section 5.3.1 Définition d'une règle

Nous formalisons la définition d'une règle dans le langage Z selon le schéma suivant:

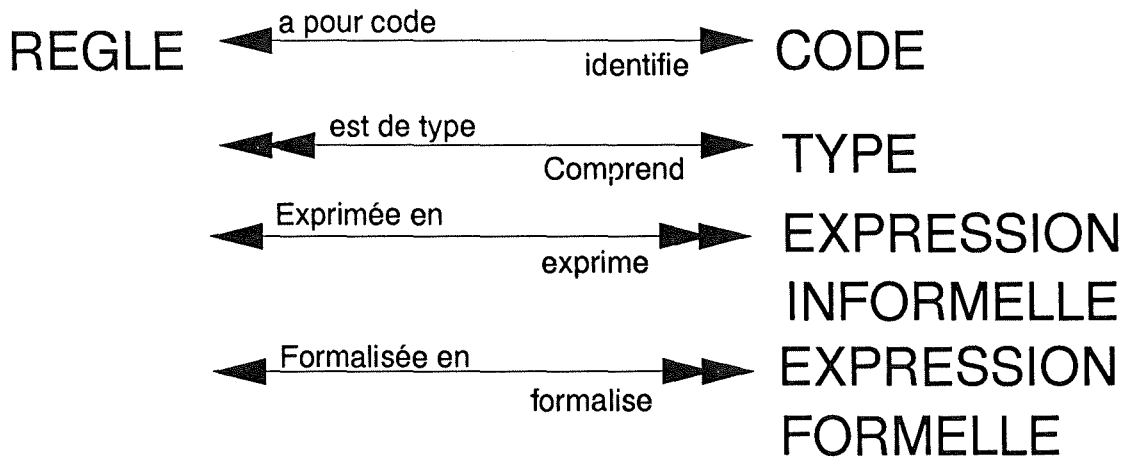


FIGURE 4.2

Cette définition est établie aussi bien pour les règles élémentaires de gestion que pour les règles de pilotage.

Les règles sont donc définies par un code unique, un type, une ou plusieurs expressions informelles et une ou plusieurs expression formelles.

La codification ne présente qu'une utilité pratique dans le cadre de l'identification d'une règle.

L'expression informelle d'une règle correspond à l'expression en langage naturel de la signification de la règle. La possibilité d'avoir plusieurs expressions informelles pour une même règle provient de la souplesse des langages naturels.

L'expression formelle d'une règle correspond à sa représentation sous forme de formule dans le langage Z. La possibilité d'avoir plusieurs expressions formelles pour une même règle provient également de la souplesse du langage Z mais aussi du fait qu'une règle peut être vérifiée à partir de concepts différents selon qu'elle soit déclenchée dans le cadre d'un écran, d'une fenêtre ou d'un champ de saisie. Nous étudierons cet aspect plus profondément dans la Section 5.3.3 Portée d'une règle.

Section 5.3.2 Les types de règles

La typologie que nous proposons dans cette section n'est pas contradictoire avec la typologie proposée par L. Alvares [ALVA88]; nous pensons même qu'elles sont complémentaires.

Nous rappellerons que l'expression informelle d'une règle [ALVA88] n'est pas identifiante et que bien souvent, la formalisation d'une règle passe d'abord par le changement de son expression informelle.

Exemples:

Soit la règle élémentaire de gestion associée à la méthode IDA et obligeant la présence unique d'un processus de niveau projet:

- Il doit exister un et un seul processus de niveau 'PROJET'.

Quelle que soit la typologie employée, cette règle doit être décomposée en deux règles plus simples, une définissant l'obligation d'avoir un processus de niveau 'PROJET' et l'autre définissant son unicité soit:

- Il doit exister au moins un processus de niveau 'PROJET'.

- Il ne peut exister qu'un et un seul processus de niveau 'PROJET'.

Soit la règle de pilotage limitant la spécification au processus de niveau 'PHASE':

Il est interdit de spécifier des processus dont le niveau est différent de 'PHASE'

Cette règle semble à première vue définir une interdiction; cependant, une autre formalisation de cette règle présentera une nouvelle expression informelle dans les termes suivants:

Le niveau de tout processus spécifié dans ce modèle est 'PHASE'.

Notre typologie s'établit selon deux grands axes, à savoir les règles élémentaires de gestion et les règles de pilotage. Nous rappellerons à titre d'information que les règles élémentaires de gestion servent à vérifier la validité des spécifications par rapport à la méthode, elles représentent en fait la grammaire de la méthode et que les règles de pilotage servent à diriger, à organiser le travail d'analyse, elles représentent en fait la démarche de la méthode de spécification.

1) Les règles élémentaires de gestion

a Contrôle syntaxique.

Les règles vérifiant la syntaxe des spécifications sont définies au niveau de la base de connaissance associée aux langages. Elles sont des mécanismes pris en charge automatiquement par l'interface.

La correction des erreurs détectées par ces règles est de deux types; Soit la correction est automatique et transparente pour l'analyste, soit l'erreur est mentionnée et la correction est attendue avant de permettre la poursuite des saisies.

Ces règles sont uniquement déclenchées par la clôture d'un champ de saisie.

b Contrôle sémantique.

Les régles vérifiant la sémantique des spécifications sont définies sur deux niveaux.

Premièrement, nous avons les règles immergées dans l'interface de pilotage et qui vérifient la sémantique des spécifications par rapport aux concepts définis en support de l'interface. Nous pensons aux notions d'objet et de relation. Nous pensons par exemple aux règles de gestion des types d'objet et de relation, aux règles vérifiant l'unicité des noms d'objet,...

Ces règles sont également validées lors de la clôture d'un champ de saisie et seront généralement associées à des règles de pilotage exigeant la correction de l'erreur avant de permettre à l'analyste de continuer.

Deuxièmement, nos avons les règles élémentaires de gestion choisies par le configurateur et intégrées dans la méthodes de spécification. Elles assurent l'intégrité des spécifications par rapport à la méthode.

Ces règles sont déclenchées lors d'occurrence d'événement selon les mécanismes définis dans la SECTION 5.1.

2) Les règles de pilotage

La typologie des règles de pilotage est la plus difficile à gérer, en effet, selon le point de vue où l'on se place, les définitions de classes changent et les règles se regroupent différemment.

Par exemple, il est évident que l'utilisation de négations affecte directement la classe à laquelle appartient une règle; il en est de même par l'utilisation des verbes 'devoir' et 'pouvoir'

Partant de là, nous avons malgré tout décidé de présenter un échantillon de différentes classes sans avoir la prétention d'associer chaque règle à sa classe ni d'en avoir une définition exhaustive.

Les règles de pilotage sont déclenchées lors d'occurrence d'événement selon les mécanismes définis dans la SECTION 5.1.

a Les règles immergées.

Les règles immergées correspondent aux règles de pilotage gérant la hiérarchie définie en terme d'écrans, de fenêtre et de champs de saisie ainsi qu'aux règles gérant l'obligation d'identification d'un objet.

Le premier ensemble de règles immergées est facile à appréhender, il constitue le coeur de l'interface de pilotage. Ce sont ces règles qui gèrent l'enchaînement écran-fenêtre, fenêtre-champ de saisie, et retour. Ce sont elles également qui gèrent les occurrences d'événement.

Le deuxième ensemble de règles immergées correspond à la mise en place de certaines contraintes sur l'organisation des spécifications. Précédemment, nous avons dit que la spécification d'une relation s'effectuait à partir d'un objet, le type de l'objet et l'objet lui-même devant participer à l'ensemble source de la relation. Afin de rester cohérent, nous prévoyons la définition de règles de pilotage obligeant l'identification d'un objet avant la spécification de toute autre relation.

b Des règles directes et indirectes

Les règles indirectes correspondent aux règles de pilotage associées aux mécanismes de correction des erreurs détectées par les règles élémentaires de gestion. Nous appelons ces règles indirectes car elles ne sont mise en oeuvre que d'une manière indirecte.

Les règles directes correspondent à toutes les autres règles de pilotage. Elles sont choisies par le configurateur lors de l'élaboration de la méthode de spécification et sont mise en oeuvre directement par l'utilisation des fonctions offertes par l'interface de pilotage.

c Des règles passives et des règles actives.

Les règles de pilotage passives focalisent l'attention de l'analyste sur certains points sans le forcer dans l'organisation de son travail. Elles définissent les limites de son domaine sans toutefois, y établir un chemin précis. Typiquement, nous aurons dans cette classe, toutes les règles définissant des conseils dans le travail d'analyse.

Les règles de pilotage actives gèrent automatiquement le travail de l'analyste. Elles définissent non seulement le domaine de travail mais également le chemin à

suivre. Nous aurons typiquement dans cette classe les règles d'interdiction. En effet, il est stupide de laisser la possibilité à l'analyste de spécifier quelque chose et de le lui refuser suite à une interdiction.

d L'obligation, l'interdiction et le conseil.

Les règles de pilotage relatives à l'obligation permettent au configurateur d'obliger l'analyste à spécifier certains aspects du SI ou à préciser certaines spécification avant de pouvoir en aborder d'autres.

Les règles d'obligation peuvent se concevoir soit en règles passives, par exemple, l'analyste ne peut clôturer un écran ou une fenêtre sans avoir établi la spécification exigée; soit en règles actives, par exemple, l'environnement de saisie des spécifications obligatoires est présenté à l'analyste et il ne peut pas en sortir tant qu'il n'y a pas satisfait.

Les règles d'interdiction empêchent l'analyste d'accéder aux environnements de spécification d'éléments de spécifications particuliers.

Les règles exprimant le conseil sont des règles destinées à guider l'analyste dans son travail. Généralement, ces règles seront définies en association avec les règles élémentaires de gestion dans le cadre de la correction des erreurs détectées.

Section 5.3.3 Portée d'une règle

La notion de portée d'une règle est une nouvelle notion que nous voulons introduire. En effet, nous avons établi un lien concret entre les règles et la notion d'événement. Cependant, toutes les règles ne sont pas associables à tous les types d'événement. Certaines restrictions sont à prévoir.

De plus, la validation de règle doit se concevoir différemment selon le type d'événement qui la déclenche.

Dans ce contexte, la portée d'une règle définit le type d'événement auquel la règle s'associe. De ce point de vue, nous pouvons dégager que l'expression formelle d'une règle se définit sur base de sa portée et qu'à une expression informelle pourront correspondre plusieurs expressions formelles.

En effet, il est inutile de créer de nouvelles règles dues au fait de la notion de portée. Une règle identifiée par un code, représentée par une expression informelle et jointe à une portée particulière définit l'expression formelle à utiliser lors de la validation.

Le schéma de la figure 4.3 donne la représentation graphique en langage Z de la définition d'une règle sur base de la notion de portée.

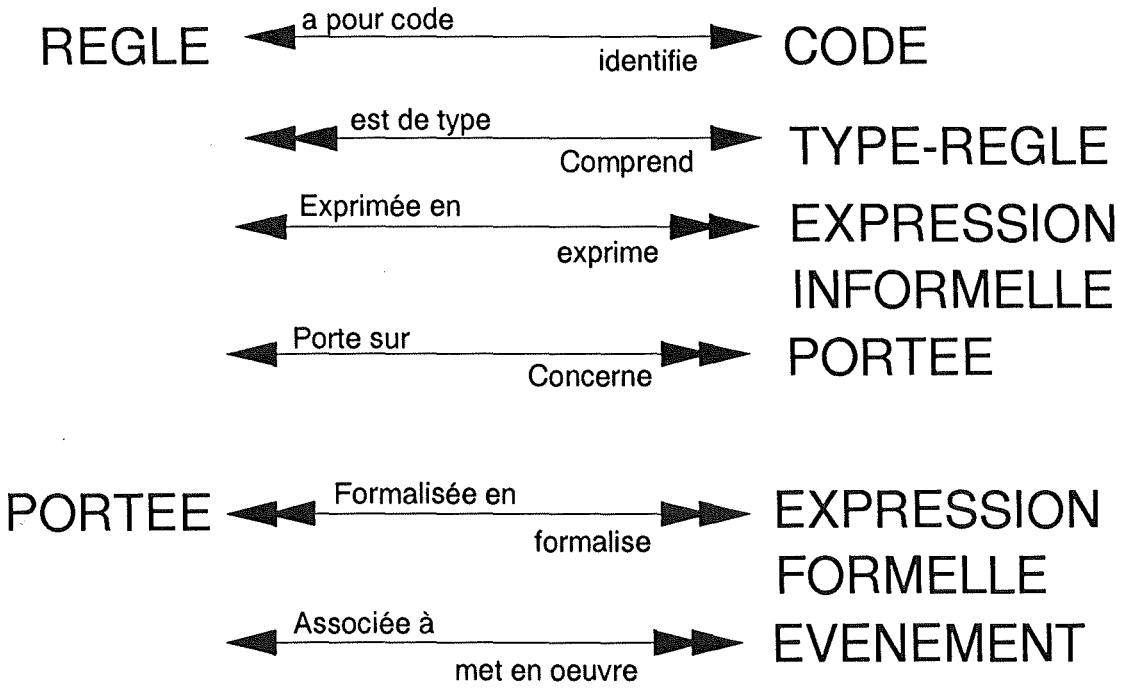


FIGURE 4.3

Cette définition s'établit dans le cadre de la formalisation des règles par rapport à la méthode de modélisation. Relativement aux méthodes de spécification, nous devons introduire une restriction par rapport à la fonction **'Formalisée en'** qui devra être unique

(monovaluée) et non plus multiple (multivaluée). En effet, dans le cadre de la validation d'une règle par rapport à une occurrence d'un événement particulier, une et une seule expression formelle devra être applicable.

CHAPITRE VI CONCLUSION

Au départ de ce travail, notre objectif était de définir un cadre générique à la conception d'un outil d'aide à la spécification supportant toute méthode et pouvant s'adapter à toutes organisations et à tous projet.

Le chemin parcouru depuis est long et parfois un peu étriqué, mais nous pensons avoir accompli en partie notre objectif; en tous les cas, nous pensons avoir précisé la voie pour les travaux à suivre le présent ouvrage.

D'un point de vue complémentaire, il nous semble intéressant d'approfondir les notions de méthode de modélisation et de méthode de spécification afin de clarifier les notions de modèle et de démarche et de définir avec précision les recoupements existant entre les deux types de méthode.

Dans la continuité de ce travail, l'étape suivante dans la définition d'un outil d'aide générique repose sur la définition d'une typologie aussi exhaustive que possible des règles élémentaires de gestion et des règles de pilotage. En parallèle, la formalisation d'une ou plusieurs méthodes de modélisation dans le métalangage Z ou dans un autre métalangage ainsi que la création de plusieurs méthodes de spécification devraient soutenir par l'exemple le présent travail et la typologie des règles envisagée.

Afin d'améliorer la qualité de l'aide mise en oeuvre dans la correction des erreurs détectées lors de la validation des règles élémentaires de gestion, la définition de diagnostic complexe établi sur base de la typologie des règles élémentaires de gestion et des relations existant entre les différents types définis. Ces diagnostics devraient permettre la détection et la correction directe d'erreurs induites indirectement d'autres erreurs.

Dans le même ordre d'idée, l'étude des connaissances relatives à la qualité d'un travail d'analyse devraient permettre une meilleure complétude des spécifications et enrichir l'aide apportée à l'analyste tant au niveau de l'organisation de son travail qu'au niveau de la qualité de ses spécifications.

D'un point de vue technique, il est également intéressant d'envisager le partage de l'outil dans le cadre du travail en équipe. Par exemple, la validation des règles élémentaires de gestion ne doit plus uniquement s'envisager dans le cadre du travail individuel de chaque analyste mais également dans le cadre de la mise en commun des spécifications.

Finalement, nous ne devons pas oublier que cet outil doit s'inscrire comme source d'information et de documentation dans le cadre d'une ou éventuellement plusieurs méthodes de conception. Il faut donc envisager l'exploitation des connaissances contenues dans l'outil par des outils extérieurs permettant la formalisation de résultat et de rapport d'analyse, en particulier pour la production de schémas conceptuels complets, cohérents, communicables conformes et réalisables.

BIBLIOGRAPHIE

[ABR78] ABRIAL J.R.: Manuel du langage Z.
Note Z1 à Z15 non publiées, EDF, Paris, 1978.

[ALV88] ALVARES L.O.: Contribution à l'étude du pilotage de la modélisation des systèmes d'information.
Thèse de doctorat, Université J. FOURIER - Grenoble I
octobre 1986.

[BOD83] BODART F., PIGNEUR Y.: Conception assistée des applications informatiques I-étude d'opportunité et analyse conceptuelle.
Masson, Paris, 1983.

[BOU86] BOUZEGHOUB M.: SECSI: Un système expert en conception de systèmes d'information - modélisation conceptuelle de schémas de base de données.
Thèse de doctorat, Université Paris VI, mars 1986.

[CEC87] CECIMA: Message spécification, maquette, projet.
Journées Pratique des Méthodes et Outils Logiciels d'aide à la Conception de Systèmes d'Information,
Nantes, 23-24 septembre 1987.

[CGI87] CGI-Informatique: La station de travail PACBASE.
Journées Pratique des Méthodes et Outils Logiciels d'aide à la Conception de Systèmes d'Information,
Nantes, 23-24 septembre 1987.

[GAL84] GALACSI: Les systèmes d'information - analyse et conception.
Dunod Informatique, Paris, 1984.

[GIG86] GIGCH J. P., PIPINO L. L.: In search of a paradigm for the discipline of information systems.
Future Computing systems, Oxford University Press, 1 (1),
1986.

[LEM77] LEMOIGNE J.L.: La théorie du système général.
Presses Universitaires de France, Paris, 1977.

[PRO86] PROIX C.: Système expert pour la conception des systèmes d'information.
Information & Communication, AFCET, Paris, 3-5 juin 1986.

[ROL88] ROLLAND C., FOUCAUT O., BENCI G.: Conception des systèmes d'information: la méthode REMORA.
Editions Eyrolles, Paris, 1988.

[ROU87] ROUSSEL A.: Progiciels.
Journées Pratique des Méthodes et Outils Logiciels d'aide à la Conception de Systèmes d'Information,
Nantes, 23-24 septembre 1987.

[VES87] VESPA R., SECHER D.: Conceptor.
Journées Pratique des Méthodes et Outils Logiciels d'aide à la Conception de Systèmes d'Information,
Nantes, 23-24 septembre 1987.

[WAS82] WASSERMAN A.I.: Automated tools in the information system development environment
Automated tools for information systems design, Schneider H.J. et Wasserman A.I. (eds), North Holland, Amsterdam, 1982.

[WIL65] WILSON I.G., WILSON M.E.: Information, computers and system design.
Morgan Kaufman Publishers, Los Altos, 1987.

ANNEXES

Annexe I Formalisation d'une méthode de modélisation.

A Déclaration des faits relatifs aux items d'une méthode de modélisation

1) En PROLOG

type_objet (type)

Définition des types d'objet.

r_objet (écran, objet, relation, place (numéro), état)

L'objet de type *'objet'* est repris dans la relation *'relation'* comme sujet ou complément (*place*) à la position (*numéro*) de manière obligatoire ou facultaive (*état*).

Les notions de sujet et de complément sont définies selon que le type d'objet appartienne à l'ensemble cible ou à l'ensemble source de la relation.

L'écran *'écran'* identifie l'écran de saisie de cette relation relativement à ce type d'objet et aux caractéristiques définies. Un écran de saisie d'un type de relation relativement à un type d'objet peut être différent dans le cadre de l'utilisation des opérations ensemblistes.

type_relation (type, genre)

Définition des types de relation et de leur genre (mono- ou multi-valuée).

connecte (écran, relation, sujet, complément)

Définition des ensembles source et cible relié par la relation et identification de l'écran associé à la représentation de cette relation (l'union d'ensemble d'entité se définit par plusieur prédicats relatifs à une même relation).

extension (nom, définition)

Définition des ensembles définis en extension.

événement (nom _ événement)

Définition des types d'événements.

B Déclaration des faits liés à l'énoncé des règles de gestion

règle (code, type, exp. informelle)

Définition des règles à partir d'un code d'identification un type et une expression informelle. A ce niveau, nous n'avons pas encore élaboré un système permettant la manipulation des expressions formelles des règles.

portée (code règle, type portée, nom)

Définition de la portée des règles. Cette portée s'applique uniquement sur les types d'objets et de relations;nous ne l'avons pas encore définie sur les différents types d'événements. Actuellement le type d'événement est sous-entendu et est défini comme étant la fin de spécification de l'objet ou de la relation.

ensemble_interdit (valeur)
cardinalite _ minimale (valeur)
cohérence (valeur)
complétude (valeur)

Essai de définition de la typologie des règles.

C Déclaration des faits concernant la configuration d'une méthode

methode(nom_methode)
modele(nom_modèle)
initial(modèle initial)
suivant(modèle précédent,modèle suivant

Définition de la méthode par un nom et des modèles associés. Actuellement, nous avons gardé la notion de succession des modèles.

contient(paquet,règle)
comprend(modele,paquet)

Définition des paquets de règles par rapport aux types d'événement.(*contient*) et par rapport aux modèles (*comprend*)

D Déclaration des faits concernant la spécification d'une application

application(nom de l'application)
modele_courant(nom du modele courant)
objet(type,nom,description)
relation(type,sujet(s),complément(s))

E Exemple d'une méthode de spécification

1) Définition des concepts de la méthode de modèlisation

type_objet("PROCESS")
type_objet("INTERFACE")
type_objet("MESSAGE")

r_objet("ECRAN1","PROCESS","NIVEAU DE",sujet(1),'O')
r_objet("ECRAN1","PROCESS","MODE DE",sujet(1),'O')
r_objet("ECRAN2","PROCESS","GENERATES",sujet(1),'O')
r_objet("ECRAN2","PROCESS","RECEIVES",sujet(1),'O')
r_objet("ECRAN2","PROCESS","GENERATED BY",complement(1),'O')
r_objet("ECRAN2","PROCESS","RECEIVED BY",complement(1),'O')
r_objet("ECRAN1","PROCESS","PART OF",sujet(1),'O')
r_objet("ECRAN1","PROCESS","PART OF",complement(1),'O')
r_objet("ECRAN1","PROCESS","RESPONSABLES DE",complement(1),'O')
r_objet("ECRAN1","PROCESS","GERE PAR",sujet(1),'O')

r_objet("ECRAN1","INTERFACE","NATURE DE",sujet(1),'O')
r_objet("ECRAN1","INTERFACE","GENERATES",sujet(1),'O')
r_objet("ECRAN1","INTERFACE","RECEIVES",sujet(1),'O')
r_objet("ECRAN1","INTERFACE","GENERATED BY",complement(1),'O')
r_objet("ECRAN1","INTERFACE","RECEIVED BY",complement(1),'O')
r_objet("ECRAN1","INTERFACE","RESPONSABLES DE",sujet(1),'O')
r_objet("ECRAN1","INTERFACE","GERE PAR",complement(1),'O')

r_objet("ECRAN2","MESSAGE","NATURE DE",sujet(1),'O')
r_objet("ECRAN1","MESSAGE","GENERATES",complement(1),'O')
r_objet("ECRAN2","MESSAGE","GENERATES",complement(1),'O')
r_objet("ECRAN1","MESSAGE","RECEIVES",complement(1),'O')
r_objet("ECRAN2","MESSAGE","RECEIVES",complement(1),'O')
r_objet("ECRAN1","MESSAGE","RECEIVED BY",sujet(1),'O')
r_objet("ECRAN2","MESSAGE","RECEIVED BY",sujet(1),'O')
r_objet("ECRAN1","MESSAGE","GENERATED BY",sujet(1),'O')
r_objet("ECRAN2","MESSAGE","GENERATED BY",sujet(1),'O')

r_objet("ECRAN1","NIVEAU","NIVEAU DE",complement(1),'O')
r_objet("ECRAN1","NATURE","NATURE DE",complement(1),'O')
r_objet("ECRAN1","MODE","MODE DE",complement(1),'O')

```
type_relation("NIVEAU DE","MONO")
type_relation("MODE DE","MONO")
type_relation("NATURE DE","MONO")
type_relation("GENERATES","MULTI")
type_relation("GENERATED BY","MULTI")
type_relation("RECEIVES","MULTI")
type_relation("RECEIVED BY","MULTI")
type_relation("PART OF","MULTI")
type_relation("RESPONSABLES DE","MULTI")
type_relation("GERE PAR","MONO")

connecte("ECRAN1","NIVEAU DE",["PROCESS"],["NIVEAU"])
connecte("ECRAN1","MODE DE",["PROCESS"],["MODE"])
connecte("ECRAN1","NATURE DE",["INTERFACE"],["NATURE"])
connecte("ECRAN2","NATURE DE",["MESSAGE"],["NATURE"])
connecte("ECRAN1","GENERATES",["INTERFACE"],["MESSAGE"])
connecte("ECRAN2","GENERATES",["PROCESS"],["MESSAGE"])
connecte("ECRAN1","RECEIVES",["INTERFACE"],["MESSAGE"])
connecte("ECRAN2","RECEIVES",["PROCESS"],["MESSAGE"])
connecte("ECRAN1","GENERATED BY",["MESSAGE"],["INTERFACE"])
connecte("ECRAN2","GENERATED BY",["MESSAGE"],["PROCESS"])
connecte("ECRAN1","RECEIVED BY",["MESSAGE"],["INTERFACE"])
connecte("ECRAN2","RECEIVED BY",["MESSAGE"],["PROCESS"])
connecte("ECRAN1","PART OF",["PROCESS"],["PROCESS"])
connecte("ECRAN1","RESPONSABLES DE",["INTERFACE"],["PROCESS"])
connecte("ECRAN1","GERE PAR",["PROCESS"],["INTERFACE"])

extension("NIVEAU",["","PROJET","APPLICATION","PHASE","FONCTION"])
extension("MODE",["","INTERACTIF","BATCH","MANUEL"])
extension("NATURE",["","EXTERNE","INTERNE"])
```

ensemble_interdit("110")
cardinalite_minimale("210")

2) Exemples de règles et de leurs portées.

regle("r4","162" la hiérarchie des traitements ne doit pas être définie circulairement")	portee("r4","RELATION","PART OF") portee("r4","OBJET","PROCESS")
regle("r5","210","Il doit y avoir au moins un processus défini")	portee("r5","OBJET","PROCESS")
regle("r6","220","Chaque processus doit avoir un déclenchement")	portee("r6","RELATION","TRIGGERED BY") portee("r6","OBJET","PROCESS")
regle("r7","230","Chaque processus doit avoir au moins une entrée")	portee("r7","OBJET","PROCESS") portee("r7","RELATION","RECEIVES")
regle("r8","230","Chaque processus doit avoir au moins une sortie")	portee("r8","OBJET","PROCESS") portee("r8","RELATION","GENERATES")
regle("r11","254","Un message reçu par un processus décomposé doit être reçu également par au moins un des processus fils du processus décomposé")	portee("r11","OBJET","MESSAGE")
regle("r12","254","Un message généré par un processus décomposé doit être généré également par au moins un des processus fils du processus décomposé")	portee("r12","OBJET","MESSAGE")
regle("r13","170","Un message ne peut être à la fois généré et reçu par des interfaces")	portee("r13","OBJET","MESSAGE") portee("r13","RELATION","GENERATED BY") portee("r13","RELATION","RECEIVED BY")
regle("r14","210","Il doit y avoir au moins deux messages définis")	portee("r14","OBJET","MESSAGE")
regle("r15","230","Chaque message doit être reçu par un processus ou une interface")	portee("r15","OBJET","MESSAGE") portee("r15","RELATION","RECEIVED BY")

regle("r16","230","Chaque message doit être généré par un processus ou une interface")	portee("r16","OBJET","MESSAGE") portee("r16","RELATION","GENERATED BY")
regle("r17","220","Chaque interface doit être utilisée")	portee("r17","OBJET","INTERFACE")
regle("r18","210","Il doit avoir au moins une interface définie")	portee("r18","OBJET","INTERFACE")
regle("r19","230","Chaque condition doit avoir une description")	portee("r19","OBJET","CONDITION")
regle("r20","230","Chaque processus doit avoir un niveau")	portee("r20","OBJET","PROCESS") portee("r20","RELATION","NIVEAU DE")
regle("r22","164","Chaque processus fils d'un processus de niveau application doit être de niveau phase")	portee("r22","OBJET","PROCESS") portee("r22","RELATION","PART OF")
regle("r23","164","Chaque processus fils d'un processus de niveau phase doit être de niveau fonction")	portee("r23","OBJET","PROCESS") portee("r23","RELATION","PART OF")
regle("r24","230","Chaque processus doit être dans la hierarchie des traitements")	portee("r24","OBJET","PROCESS") portee("r24","RELATION","PART OF")
regle("r25","110","La spécification de messages est interdite")	portee("r25","OBJET","MESSAGE")
regle("r26","110","La spécification d'interface est interdite")	portee("r26","OBJET","INTERFACE")
regle("r27","130","La spécification de la hierarchie des traitements est interdite")	portee("r27","RELATION","PART OF")

3) Définition des modèles de la méthode de spécification

methode("**METHOD-M**")

modele("MODELE2")
modele("MODELE1")
initial("MODELE1")
suivant("MODELE1","MODELE2")

contient("Début du modèle MODELE1","r25")
contient("Début du modèle MODELE1","r26")
contient("Fin de spécification de l'objet PROCESS MODELE1","r4")
contient("Fin de spécification de l'objet PROCESS MODELE1","r11")
contient("Fin de spécification de l'objet PROCESS MODELE1","r22")
contient("Fin de spécification de l'objet PROCESS MODELE1","r23")
contient("Fin de spécification de l'objet PROCESS MODELE1","r20")
contient("Fin de spécification de l'objet INTERFACE MODELE1","r4")
contient("Fin de spécification de l'objet INTERFACE MODELE1","r11")
contient("Fin de spécification de l'objet INTERFACE MODELE1","r22")
contient("Fin de spécification de l'objet INTERFACE MODELE1","r23")
contient("Fin de spécification de l'objet MESSAGE MODELE1","r4")
contient("Fin de spécification de l'objet MESSAGE MODELE1","r11")
contient("Fin de spécification de l'objet MESSAGE MODELE1","r22")
contient("Fin de spécification de l'objet MESSAGE MODELE1","r23")
contient("Fin de spécification de la relation NIVEAU DE MODELE1","r20")
contient("Fin du modèle MODELE1","r4")
contient("Fin du modèle MODELE1","r11")
contient("Fin du modèle MODELE1","r22")
contient("Fin du modèle MODELE1","r23")

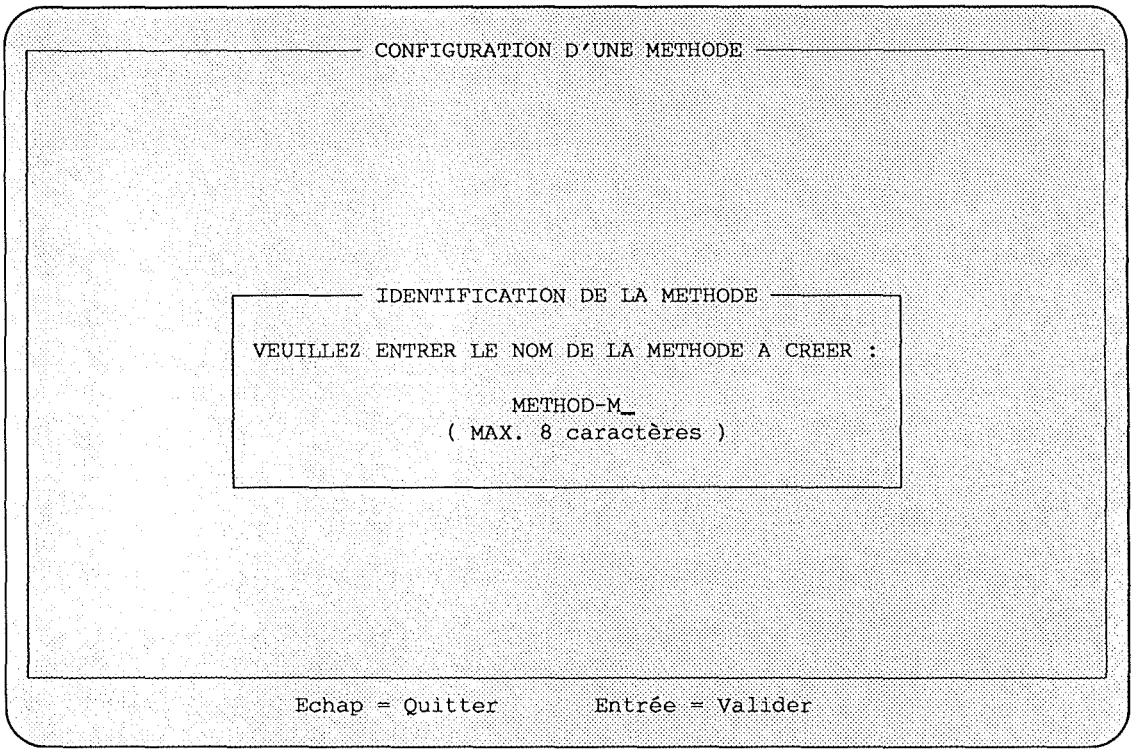
contient("Fin de spécification de l'objet PROCESS MODELE2","r4")
contient("Fin de spécification de l'objet PROCESS MODELE2","r11")

contient("Fin de spécification de l'objet PROCESS MODELE2","r22")
contient("Fin de spécification de l'objet PROCESS MODELE2","r23")
contient("Fin de spécification de l'objet PROCESS MODELE2","r18")
contient("Fin de spécification de l'objet PROCESS MODELE2","r6")
contient("Fin de spécification de l'objet PROCESS MODELE2","r7")
contient("Fin de spécification de l'objet PROCESS MODELE2","r8")
contient("Fin de spécification de l'objet INTERFACE MODELE2","r4")
contient("Fin de spécification de l'objet INTERFACE MODELE2","r11")
contient("Fin de spécification de l'objet INTERFACE MODELE2","r22")
contient("Fin de spécification de l'objet INTERFACE MODELE2","r23")
contient("Fin de spécification de l'objet MESSAGE MODELE2","r4")
contient("Fin de spécification de l'objet MESSAGE MODELE2","r11")
contient("Fin de spécification de l'objet MESSAGE MODELE2","r22")
contient("Fin de spécification de l'objet MESSAGE MODELE2","r23")
contient("Fin de spécification de l'objet MESSAGE MODELE2","r13")
contient("Fin de spécification de l'objet MESSAGE MODELE2","r14")
contient("Fin de spécification de la relation GENERATED BY MODELE2","r16")
contient("Fin de spécification de la relation RECEIVED BY MODELE2","r15")
contient("Fin du modèle MODELE2","r4")
contient("Fin du modèle MODELE2","r11")
contient("Fin du modèle MODELE2","r22")
contient("Fin du modèle MODELE2","r23")

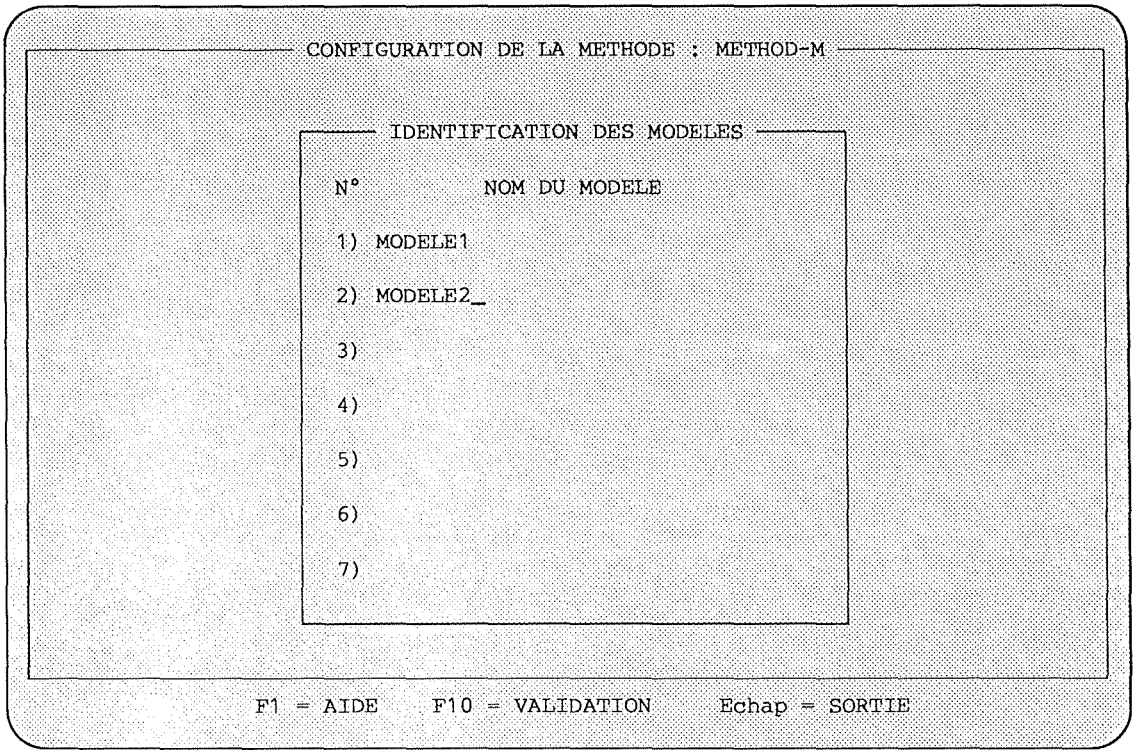
comprend("MODELE1","Début du modèle MODELE1")
comprend("MODELE1","Fin de spécification de l'objet PROCESS MODELE1")
comprend("MODELE1","Fin de spécification de l'objet INTERFACE MODELE1")
comprend("MODELE1","Fin de spécification de l'objet MESSAGE MODELE1")
comprend("MODELE1","Fin de spécification de la relation NIVEAU DE MODELE1")
comprend("MODELE1","Fin du modèle MODELE1")

comprend("MODELE2","Fin de spécification de l'objet PROCESS MODELE2")
comprend("MODELE2","Fin de spécification de l'objet INTERFACE MODELE2")
comprend("MODELE2","Fin de spécification de l'objet MESSAGE MODELE2")
comprend("MODELE2","Fin de spécification de la relation GENERATED BY MODELE2")
comprend("MODELE2","Fin de spécification de la relation RECEIVED BY MODELE2")
comprend("MODELE2","Fin du modèle MODELE2")

Annexe II Enchainement d'écrans lors de la configuration d'une méthode

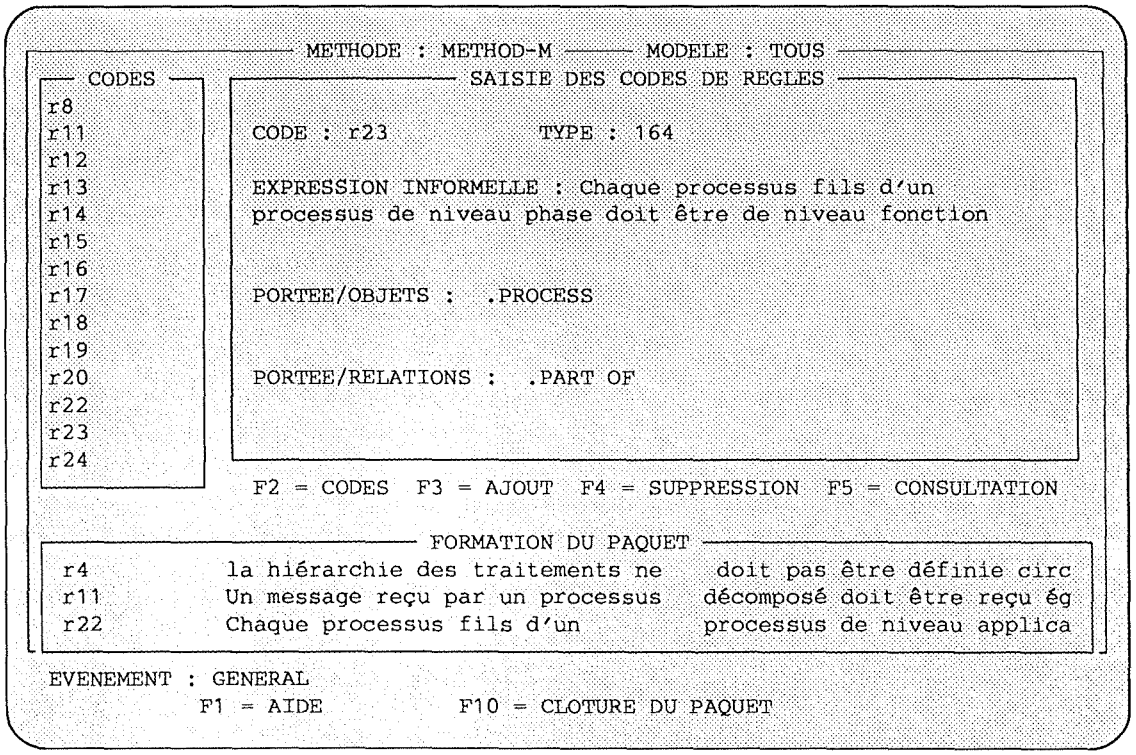


Cet écran permet la saisie du nom de la méthode de spécification que l'analyste veut configurer.

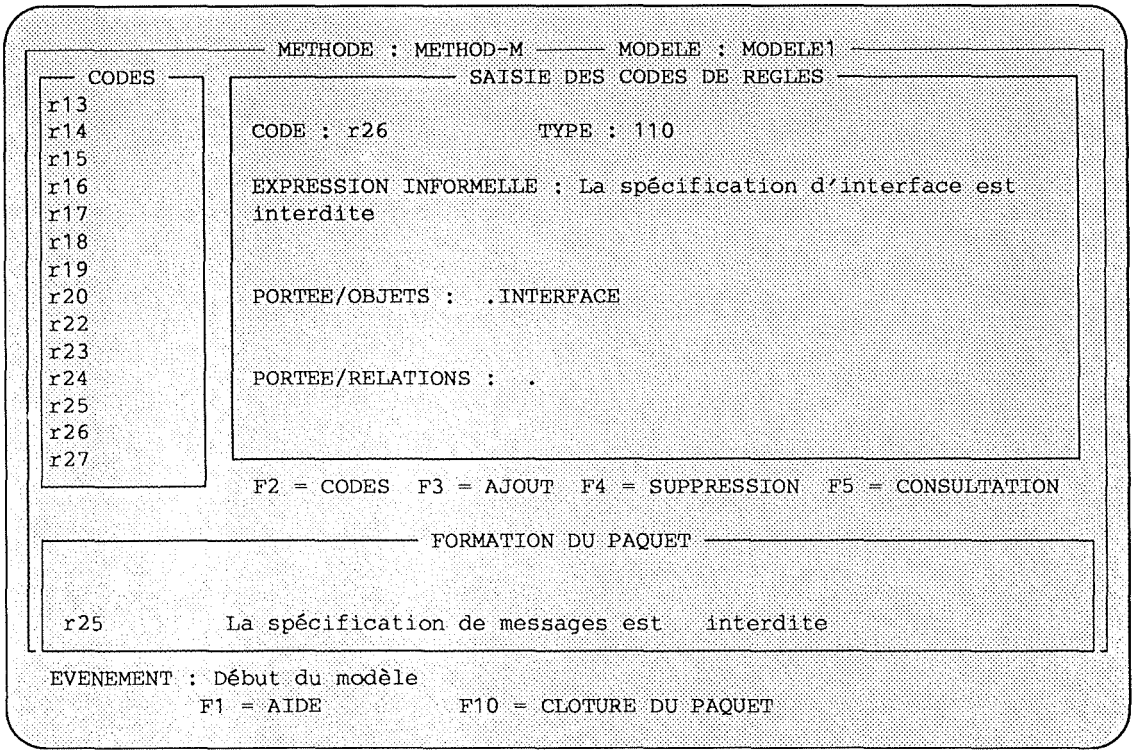


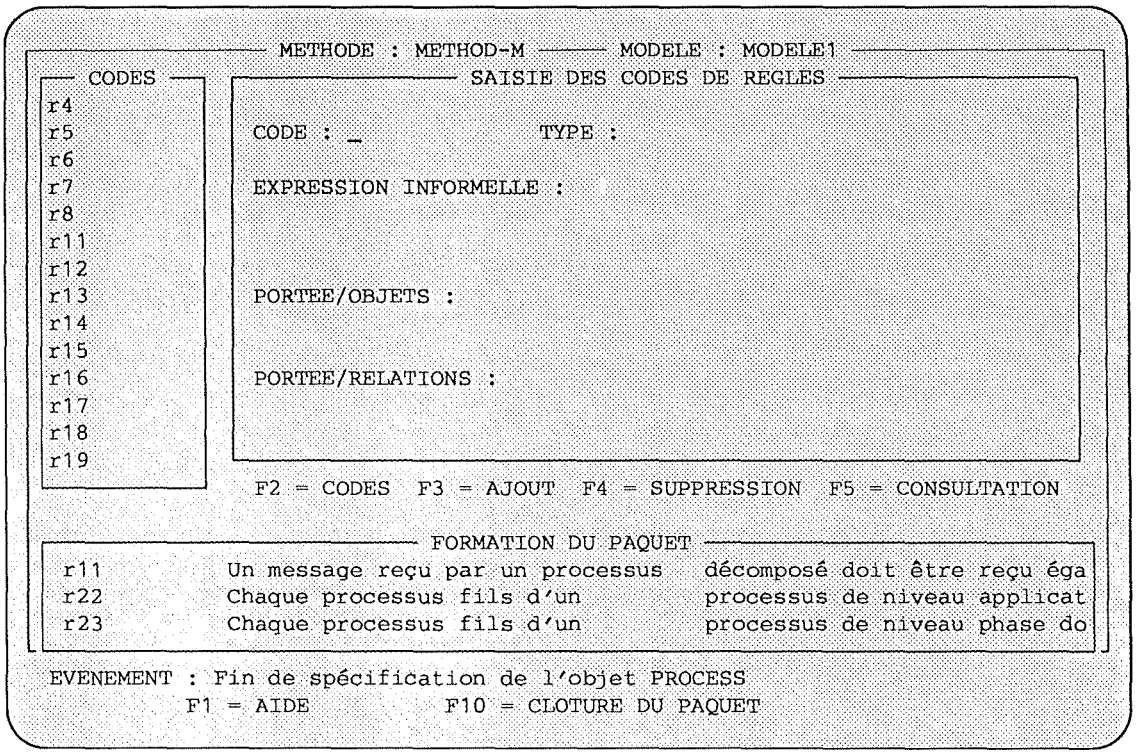
Cet écran permet la saisie des noms de modèles. La limite de 7 modèles maximum est due à une contrainte technique que nous n'avons essayer de franchir bien qu'cela ne présente pas de complexités particulières.

Les écrans suivants présentent l'interface pour la création des paquets de règles. Ces écrans sont identiques excepté qu'ils sont instanciés à l'événement et au modèle concernés (titre et avant dernière ligne de l'écran).

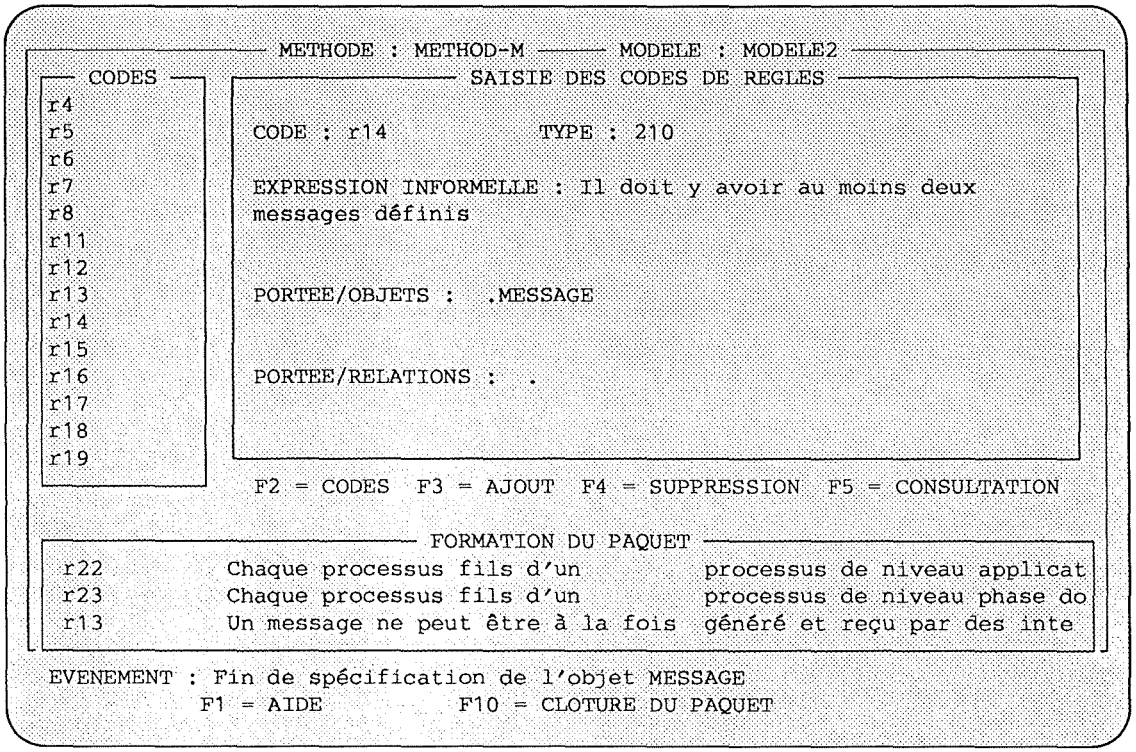


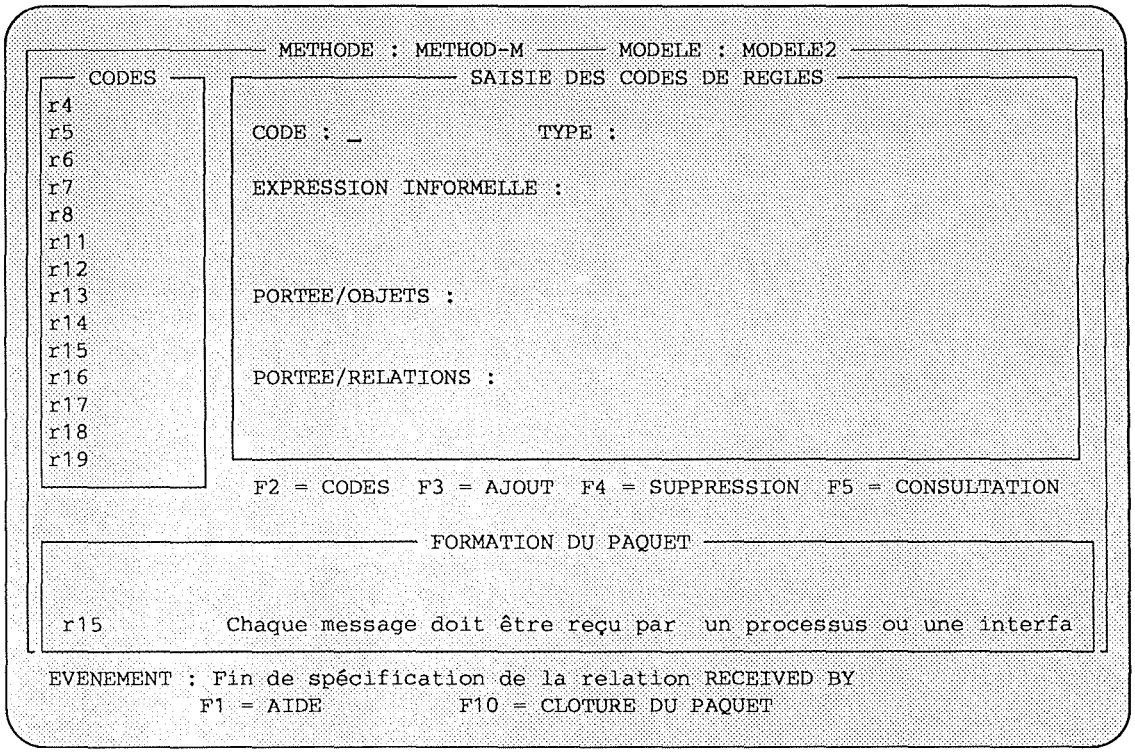
Le paquet de règles créé par l'intermédiaire de cet écran est associé à la méthode de spécification. Cela signifie que quelque soit le modèle en cours lors du travail de spécification, les règles contenues dans ce paquet devront être vérifiées


METHODE : METHOD-M
MODELE : MODELE1
CODES
r13
r14
r15
r16
r17
r18
r19
r20
r22
r23
r24
r25
r26
r27
SAISIE DES CODES DE REGLES
CODE : r26
TYPE : 110
EXPRESSION INFORMELLE : La spécification d'interface est interdite
PORTEE/OBJETS : .INTERFACE
PORTEE/RELATIONS : .
F2 = CODES F3 = AJOUT F4 = SUPPRESSION F5 = CONSULTATION
FORMATION DU PAQUET
r25 La spécification de messages est interdite
EVENEMENT : Début du modèle
F1 = AIDE
F10 = CLOTURE DU PAQUET



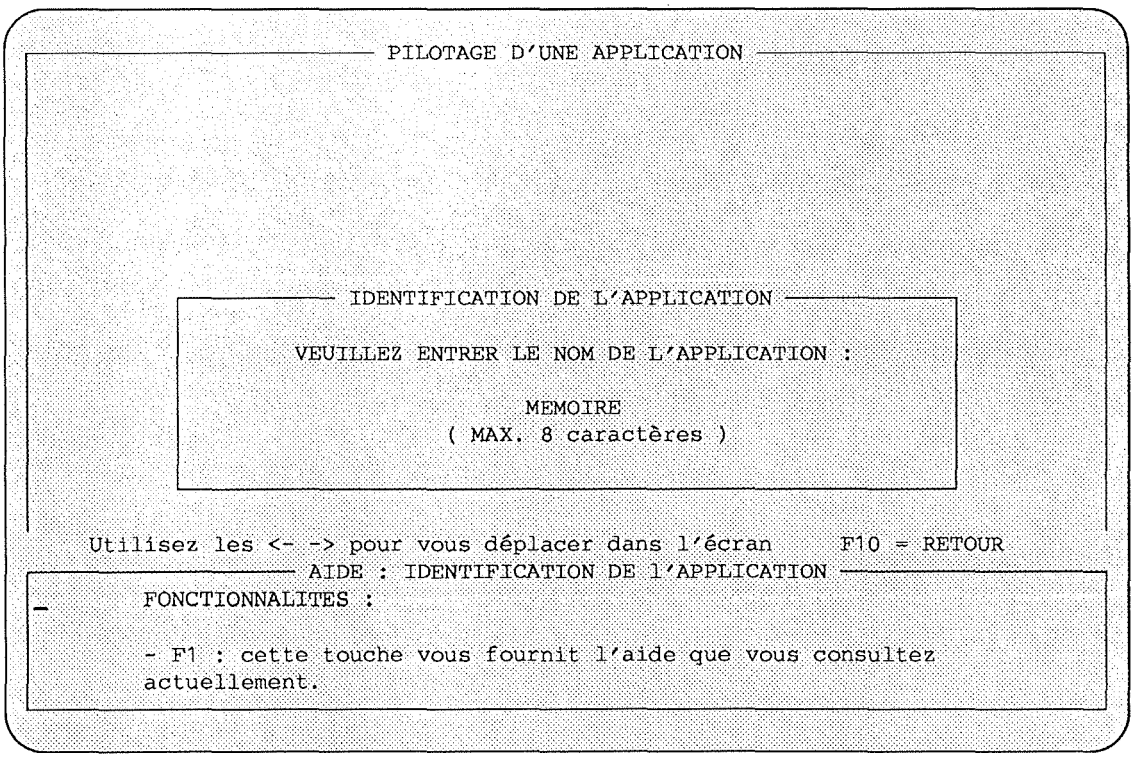
Les écrans précédents concernaient le premier modèle (modèle 1); maintenant nous pouvons remarquer au niveau du titre de l'écran que nous sommes dans le deuxième modèle.


METHODE : METHOD-M
MODELE : MODELE2
CODES
r4
r5
r6
r7
r8
r11
r12
r13
r14
r15
r16
r17
r18
r19
SAISIE DES CODES DE REGLES
CODE : r14 TYPE : 210
EXPRESSION INFORMELLE : Il doit y avoir au moins deux messages définis
PORTEE/OBJETS : .MESSAGE
PORTEE/RELATIONS : .
F2 = CODES F3 = AJOUT F4 = SUPPRESSION F5 = CONSULTATION
FORMATION DU PAQUET
r22 Chaque processus fils d'un processus de niveau applicat
r23 Chaque processus fils d'un processus de niveau phase do
r13 Un message ne peut être à la fois généré et reçu par des inte
EVENEMENT : Fin de spécification de l'objet MESSAGE
F1 = AIDE
F10 = CLOTURE DU PAQUET

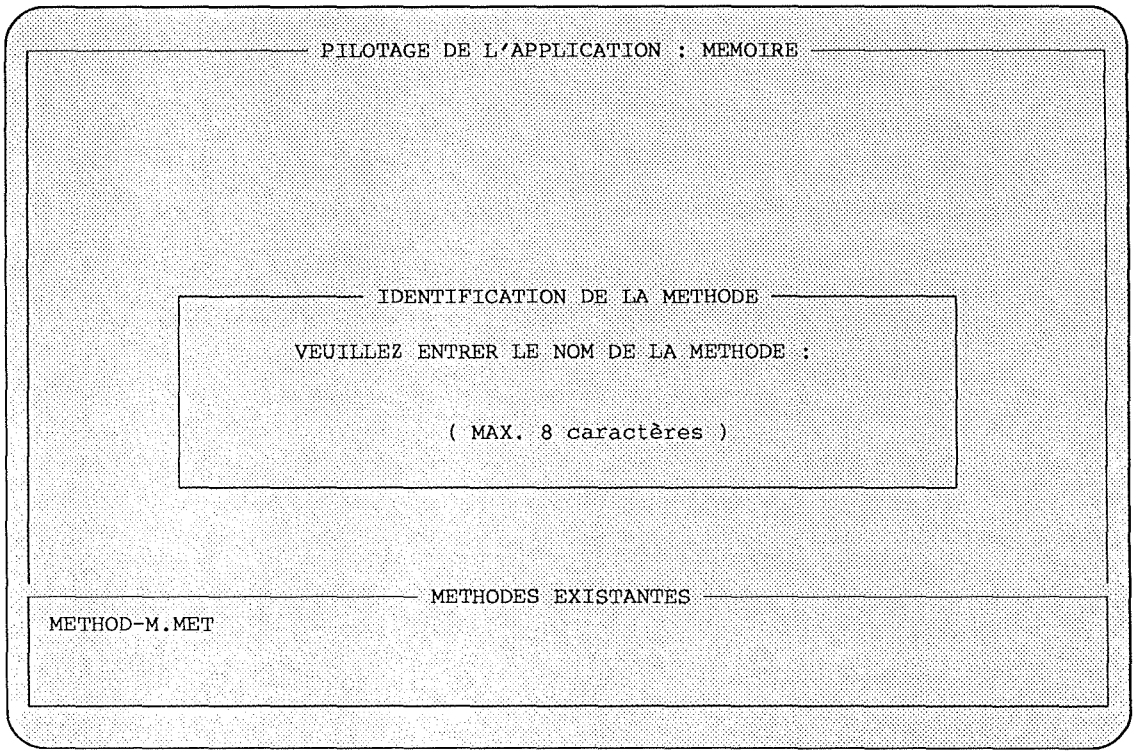

METHODE : METHOD-M
MODELE : MODELE2
CODES
r4
r5
r6
r7
r8
r11
r12
r13
r14
r15
r16
r17
r18
r19
SAISIE DES CODES DE REGLES
CODE : _
TYPE :
EXPRESSION INFORMELLE :
PORTEE/OBJETS :
PORTEE/RELATIONS :
F2 = CODES F3 = AJOUT F4 = SUPPRESSION F5 = CONSULTATION
FORMATION DU PAQUET
r15 Chaque message doit être reçu par un processus ou une interfa
EVENEMENT : Fin de spécification de la relation RECEIVED BY
F1 = AIDE F10 = CLOTURE DU PAQUET

Annexe III Enchainement d'écrans lors du pilotage des spécifications

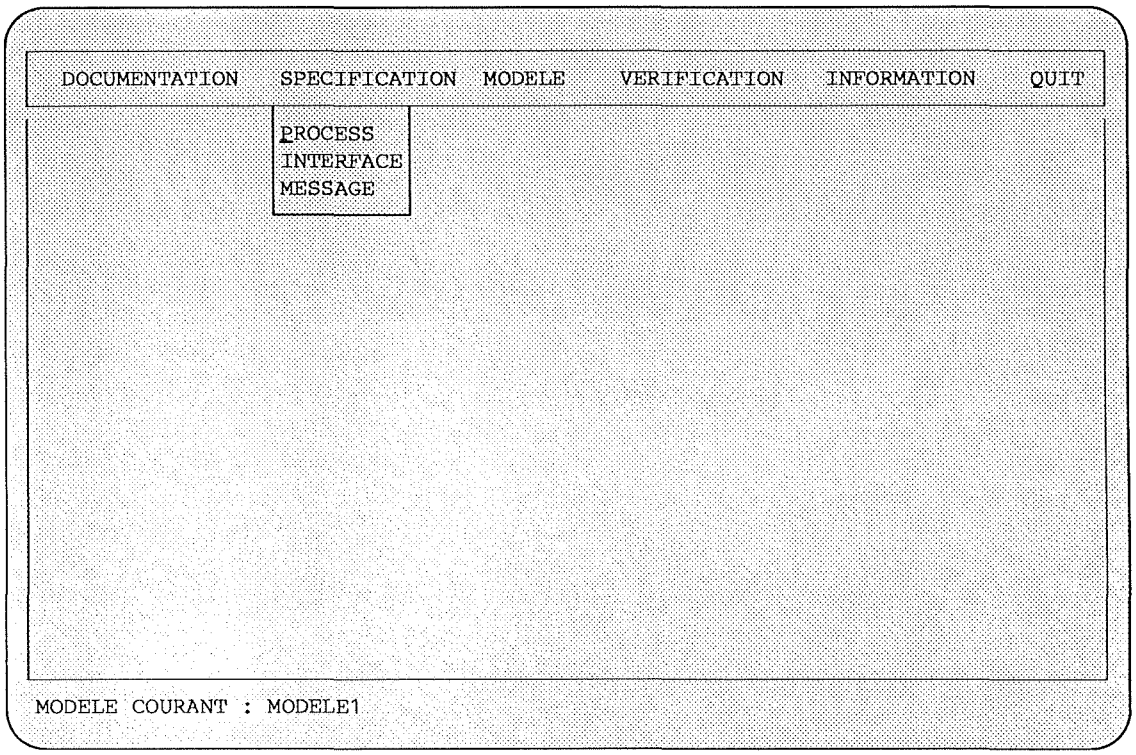
Après avoir passer en revue les exemples d'écrans relatifs à la configuration d'une méthode de spécification, nous allons maintenant visualiser les écrans relatifs au travail de spécification en lui-même.



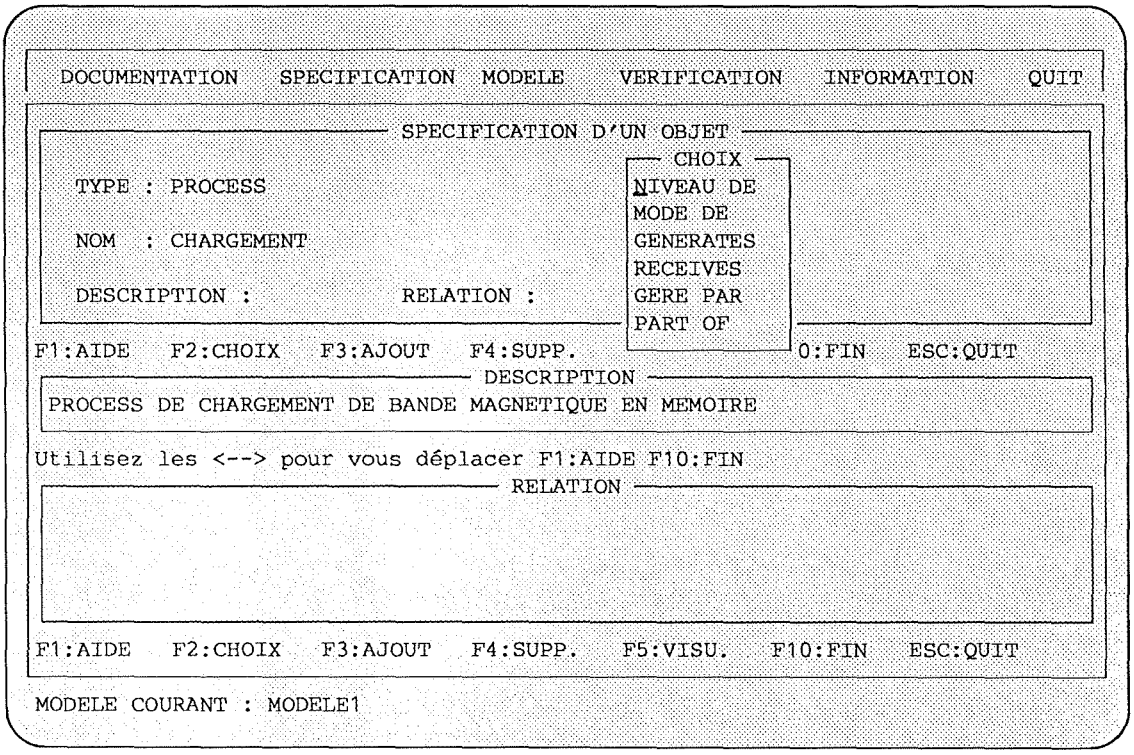
Cet écran permet l'identification du système d'information que l'analyste veut spécifier. Nous voyons également un exemple d'aide contextuelle qui peut être apporté sur le fonctionnement de l'outil.



Cet écran permet la saise de la méthode de spécification que l'analyste désire utiliser pour la spécification du SI identifié dans l'écran précédent. Nous voyons également un exemple d'aide relative à l'accès au méthode existante.

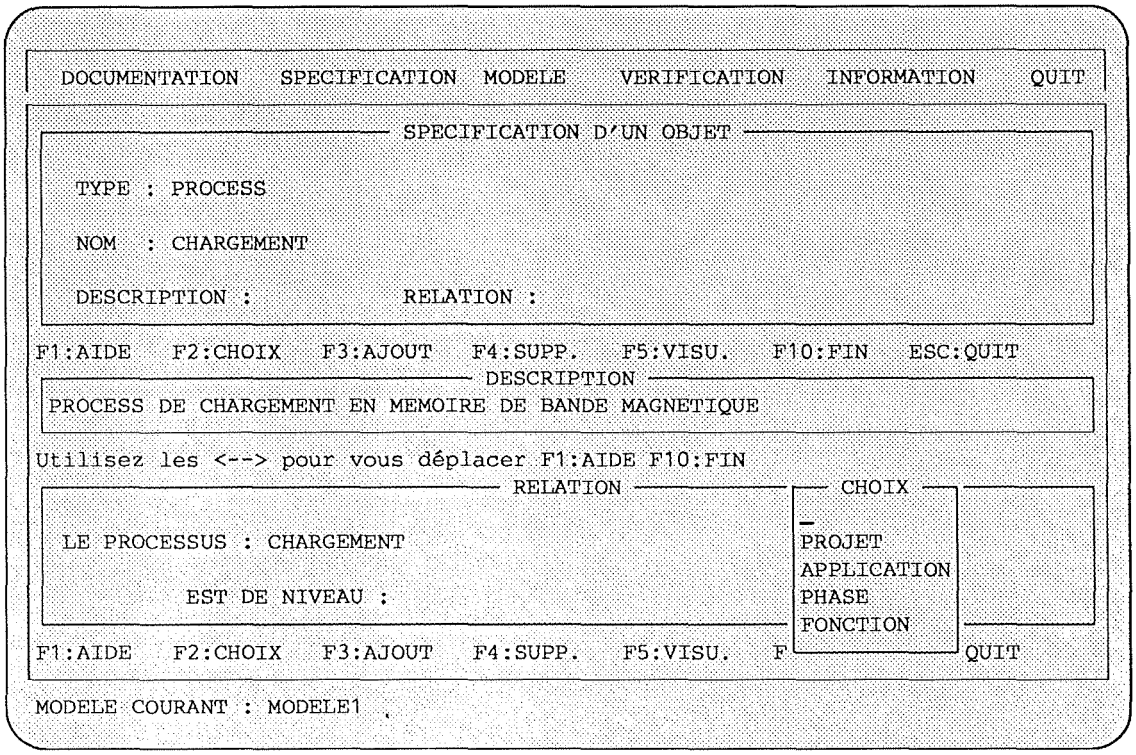


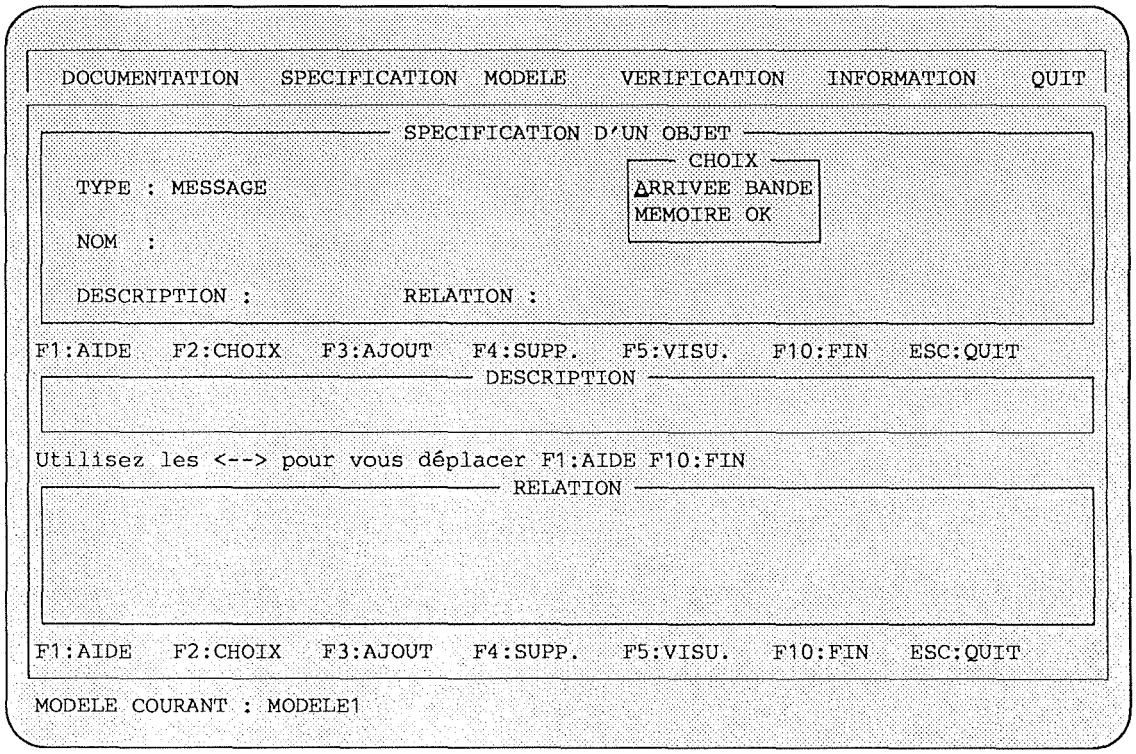
Cet écran montre le menu général de l'outil. Nous voyons que le sous-menu permettant le choix du type d'objet à spécifier est instancié directement Types d'objet définis dans la méthode de spécification. Nous n'avons pas encore prévu une relation dynamique entre cette instanciation et les règles interdisant la spécification d'un type d'objet mais l'établissment de ce lien ne présente pas une difficulté majeure.



Cet écran montre l'interface de saisie des spécifications d'un objet. Le type de l'objet à spécifier est déuit directement du choix effectué dans le menu. Nous voyons également le menu permettant de choisir le type de relation que l'analyste veut spécifier pour cet objet en particulier.

Les écrans suivant nous montre divers exemples de spécification de types d'objet et de types de relation.


DOCUMENTATION SPECIFICATION MODELE VERIFICATION INFORMATION QUIT
SPECIFICATION D'UN OBJET
TYPE : PROCESS
NOM : CHARGEMENT
DESCRIPTION :
RELATION :
F1:AIDE F2:CHOIX F3:AJOUT F4:SUPP. F5:VISU. F10:FIN ESC:QUIT
DESCRIPTION
PROCESS DE CHARGEMENT EN MEMOIRE DE BANDE MAGNETIQUE
Utilisez les <--> pour vous déplacer F1:AIDE F10:FIN
RELATION
LE PROCESSUS : CHARGEMENT
EST DE NIVEAU :
CHOIX
PROJET
APPLICATION
PHASE
FONCTION
F1:AIDE F2:CHOIX F3:AJOUT F4:SUPP. F5:VISU. F QUIT
MODELE COURANT : MODELE1


DOCUMENTATION SPECIFICATION MODELE VERIFICATION INFORMATION QUIT
SPECIFICATION D'UN OBJET
CHOIX
ARRIVEE BANDE
MEMOIRE OK
TYPE : MESSAGE
NOM :
DESCRIPTION :
RELATION :
F1:AIDE F2:CHOIX F3:AJOUT F4:SUPP. F5:VISU. F10:FIN ESC:QUIT
DESCRIPTION
Utilisez les <--> pour vous déplacer F1:AIDE F10:FIN
RELATION
F1:AIDE F2:CHOIX F3:AJOUT F4:SUPP. F5:VISU. F10:FIN ESC:QUIT
MODELE COURANT : MODELE1

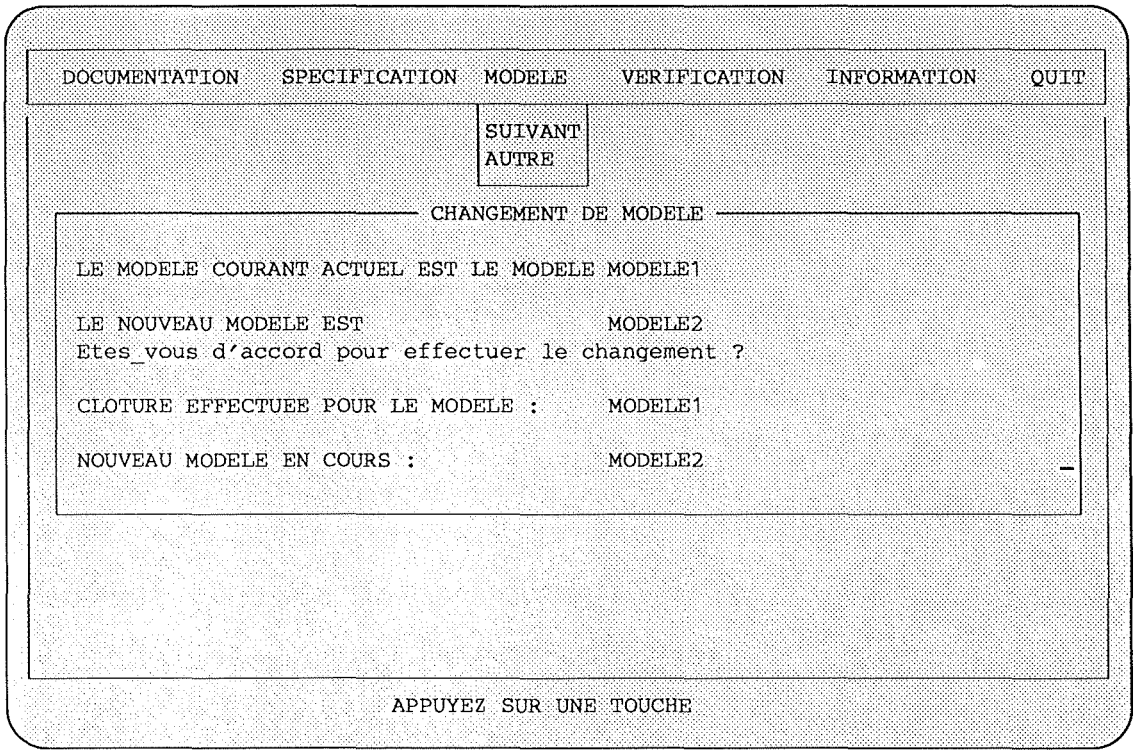

DOCUMENTATION SPECIFICATION MODELE VERIFICATION INFORMATION QUIT
SUIVANT
AUTRE
CHANGEMENT DE MODELE
LE MODELE COURANT ACTUEL EST LE MODELE MODELE1
LE NOUVEAU MODELE EST MODELE2
Etes_vous d'accord pour effectuer le changement ?
CLOTURE EFFECTUEE POUR LE MODELE : MODELE1
NOUVEAU MODELE EN COURS : MODELE2
APPUYEZ SUR UNE TOUCHE

TABLE DES MATIERES

TABLE DES FIGURES