UNIVERSITE DES SCIENCES ET DE LA TECHNOLOGIE
HAOUARI BOUMEDIENE

FACULTE DE GENIE ELECTRIQUE

DEPARTEMENT D'INFORMATIQUE

Mémoire Présenté Par

BENAHCENE DERRADJI-RACHID

Pour L'obtention Du Diplôme De

MAGISTER

Spécialité : INFORMATIQUE

Thème

UNE DEMARCHE DE DEVELOPPEMENT DE SYSTEMES D'INFORMATION PAR FRAGMENTS DE METHODES

Soutenue le : 22 Octobre 2001

Devant le jury :

Président :	**Mr M. AHMED NACER**	maître de conférences (USTHB)
Directrice de thèse :	**Mme Z. ALIMAZIGHI**	maître de conférences (USTHB)
Examinateurs :	**Mme A. AISSANI**	maître de conférences (USTHB)
	Mr N. BADACHE	maître de conférences (USTHB)
	Mme M. BOUKALA	maître de conférences (USTHB)

Ref……/……

Remerciements

Je remercie dieu de m'avoir donné le courage et la force de reprendre mes études et de terminer ce que j'ai commencé.

Je remercie Mme ALIMAZIGHI *de m'avoir faciliter la tâche en me proposant un sujet intéressant et accepté de diriger mon travail, de m'avoir laissé un bon degré de liberté dans l'organisation du travail, tout en me conseillant et me guidant dans la bonne voie et d'avoir mis à ma disposition une bonne partie de la documentation;*
Je tiens à lui exprimer sincèrement toute ma reconnaissance et ma considération.

Le président et les membres de jury d'avoir répondu favorablement à l'appel de faire partie du jury, de s'être intéressé à ce modeste travail et d'avoir accepté de l'évaluer;
je leur exprime mes remerciements et l'appréciation de leurs remarques et conseils.

Le président et les membres du conseil scientifique de l'Institut d'Informatique d'avoir accepté mon inscription en post-graduation et de m'avoir donné la chance de poursuivre mes études.

Tous les enseignants de la post-graduation de l'Institut d'Informatique pour les connaissances qu'ils nous transmises et surtout pour leur gentillesse durant l'année théorique.

Monsieur AOUISSI AHMED *directeur du centre universitaire de LAGHOUAT pour son aide, ses encouragements et son soutien ; je le remercie pour tout ce qu'il a fait pour moi.*

Ma femme et mes enfants qui m'ont supporté, soutenu et ont toujours été à mes cotés pendant les moments les plus difficiles et pour toute leur affection.

Mes remerciements s'étalent à tous ceux qui, de prés ou de loin, m'ont encouragé, soutenu et aidé tout au long de cette période ; ils sont nombreux et présents tous dans mon cœur.

Enfin je ne peux m'empêcher d'avoir une pensée intime à la mémoire des proches et des amis qui nous ont quitté et qui auraient aimé assister à ce jour ; que dieu le tout puissant leur accorde sa sainte miséricorde et les accueille en son vaste paradis.

Le travail est indispensable au bonheur de l'homme;
Il l'élève, il le console;
Et peu importe la nature du travail, pourvu qu'il profite à quelqu'un;
Faire ce qu'on peut, c'est faire ce qu'on doit.

ALEXANDRE DUMAS fils.

SOMMAIRE

DEUXIEME PARTIE

TROISIEME PARTIE

TABLE DES ILLUSTRATIONS

RESUME

Les méthodes de développement de systèmes d'information ont toujours mis l'accent sur l'aspect modèles de données, plutôt que sur l'aspect démarches permettant de les construire. Peu de travaux ont porté leur attention sur l'évolution des processus de développement. Les études faites ont proposé au départ de standardiser le développement par l'application rigide d'une seule méthode qui se veut standard « universelle » pour aboutir de nos jours à une approche plus souple dite « situationnelle » qui propose la définition d'une méthode spécifique à chaque situation. Cette évolution a été possible grâce à l'utilisation des représentations abstraites ou « méta-modèles » pour la description des produits et des processus des méthodes.

Le problème traité dans cette thèse est de considérer qu'il n'existe pas une unification de vues et de méthodes de développement des différents concepteurs et que la transformation des produits peut faire appel à plusieurs méthodes. Le produit intermédiaire deviendrait alors le résultat d'un fragment de méthode ; des problèmes de conformité et de cohérence entre autre viennent alors se poser pour l'intégration des différents produits résultants. Nous nous proposons d'analyser et d'expliciter les problèmes afférents à cette approche et de définir une esquisse de démarche permettant de résoudre une partie de ces problèmes.

mots clés

SYSTEMES D'INFORMATION, INGENIERIE DES PROCESSUS, INGENIERIE DES METHODES, META-MODELISATION, METHODES SITUATIONNELES, FRAGMENTS DE METHODES

ABSTRACT

Information systems development methods have always give more attention on the data models instead of approaches that elaborate them. Few researches pay attention on the evolution of development processes. At first studies have proposed to standardise the development by the application of one none flexible method which was qualified as standard « universal method ». In our days, studies recommend the more flexible approach called « situational methods », which propose the definition of specific method in each situation. This evolution was permitted by the use of the abstract representations or « meta-models », for the methods products and processes description.

This thesis deals with the problem of the none existence of the unified views and development methods of different conceptors, and also that the product transformation needs several methods. The intermediate product becomes than the result of method fragment. Problems like conformity and coherence can obstruct the integration of products resulted. We propose to analyse and to make explicit problem related to this approach, and to define the out line of approach that leads to resolve a part of this problems.

Keywords

INFORMATION SYSTEMS, PROCESS ENGINEERING, METHOD ENGINEERING, META-MODELING, SITUATIONAL METHODS, METHOD FRAGMENTS

INTRODUCTION

i INTRODUCTION GENERALE

Ces dernières années les technologies informatiques sont en évolution permanente : des ordinateurs encore plus puissants en constante amélioration, des instruments de dialogue avec l'homme de plus en plus évolués, des outils logiciels plus sophistiqués et adaptés, contribuant à l'amélioration de la qualité en masquant de plus en plus les techniques de base. Cependant, des domaines qui devraient, en toute logique, être les précurseurs de ces évolutions ne font que s'y adapter, la conception des systèmes d'information en est l'exemple le plus marquant [Pad98].

Avec l'émergence des moyens informatiques, les organes décisionnels des entreprises souhaitent construire et faire vivre un système maîtrisé dans le fonctionnement duquel l'information joue un rôle prépondérant. Ce système dit système d'information est à l'origine de l'automatisation des activités de l'entreprise.

Le processus d'informatisation est plus ou moins lent, complexe et coûteux. C'est là que le développement prend tout son importance pour réduire ces difficultés[Buw92].
Pour résoudre les problèmes connus qui sont associées au développement et à la maintenance des systèmes d'information, le génie logiciel propose des techniques, entre autre les méthodes de développement et les outils supportant ces méthodes.

A travers la classification des méthodes, on constate qu'elles n'ont pas toutes la même couverture du cycle de développement et qu'elles ne s'utilisent pas généralement dans tous les secteurs d'activité. Par ailleurs, certaines sont plus appropriées pour des gros projets, mais trop lourdes pour des projets individuels.

Les outils support des méthodes ont évolué d'un simple outil de modélisation à un environnement de modélisation regroupant un ensemble d'outils simples. Ils s'ouvrent vers une personnalisation des méthodes en proposant d'une part, le support de plusieurs méthodes et d'autre part, en permettent d'adapter les méthodes, voire générer des outils de modélisation[Pad98].

Les méthodes de développement de systèmes d'information ont toujours mis l'accent sur l'aspect modèles de données, plutôt que sur l'aspect démarches permettant de les construire. Peu de travaux ont porté leur attention sur l'évolution des processus de développement, sur l'évolution de la description du processus et plus particulièrement sur l'évolution de la description des produits. La dernière évolution concerne aussi la gestion du processus car toute modification dans la description des objets a un impact sur la manière dont les objets sont vus et manipulés[Jam98].

En fait les évolutions du processus de développement peuvent être classées en deux groupes :

- Le premier concerne les évolutions liées à la mise à jour du modèle décrivant les objets manipulés durant l'exécution des processus logiciels.
- Le deuxième concerne les évolutions liées à la mise à jour du modèle décrivant le processus lui-même.

Des centaines de descriptions de méthodes ont été publiées ; toute fois les recherches n'ont pas réussi à mettre au point la méthode idéale pour mettre en place rapidement un système de qualité[Kuw92]. Les études faites ont proposé au départ de standardiser le développement par l'application rigide d'une seule méthode qui se veut standard «universelle» pour aboutir de nos jours à une approche plus souple dite «situationnelle» qui propose la définition d'une méthode spécifique à chaque situation. Cette évolution a été possible grâce à l'utilisation des représentations abstraites ou «méta-modèles» pour la description des produits et processus des méthodes[Rol96].

Dans les travaux sur les supports informatisés et plus précisément sur les méta-outils, on a pris en considération l'assemblage de fragments de méthodes dans le développement de nouvelles applications, cependant on a négligé la possibilité de changer de stratégie pendant le développement ou de récupérer des applications déjà existantes permettant une éventuelle réutilisation.

Les travaux de l'IFIP(International Fédération for Information Processing) menés par le groupe CESI(Conception et Evaluation des Systèmes d'Information) précisent que : la stratégie la plus probable pour l'avenir consistera à gérer l'évolution.

ii LE CONTEXTE DE LA THESE

La conception des systèmes d'information a suivi différentes transformations dans l'histoire directement reliées au développement de technologies. Le développement des bases de données et leurs systèmes de gestion a contribué activement à celui des systèmes d'information, basé sur une indépendance des données et des traitements y-compris pendant le processus de conception.

le SI «objet artificiel » était :

- Un ensemble de données (*aspect statique du SI*).
- Un ensemble de programmes qui réglaient le comportement du SI (*aspect dynamique du SI*).
- Un ensemble de commandes permettant d'exécuter les programmes sur les données (*aspect fonctionnel du SI*).

Aujourd'hui l'avènement des approches orientées objet, qui remet en cause la dichotomie données-traitements, basées sur l'encapsulation de la structure et du comportement dans un concept unique «l'objet », a fourni une plate-forme de modélisation séduisante pour les concepteurs des systèmes d'information[Ali99].

L'introduction du processus est à l'origine de vouloir maîtriser convenablement la technologie ; l'entreprise n'est plus formée uniquement de deux piliers : personnes et technologie mais en plus d'un troisième pilier à savoir le processus[Iat93] qui est défini comme le chemin suivi permettant de faire aboutir les besoins de l'utilisateur au SI visé. Sa formalisation vise à décrire de façon détaillée, à l'aide d'un formalisme exécutable, le mode de production d'un type de logiciel[Ost87].

L'introduction de la technologie du «processus» devrait accroître la productivité des équipes de développement et améliorer la qualité des produits finaux, mais elle nécessite en revanche beaucoup de changements tant au niveau global(stratégie de développement de l'entreprise, gestion de ses projets), qu'au niveau Individuel(les habitudes des développeurs). Il est donc indispensable que cette nouvelle technologie soit introduite de façon progressive au moyen d'outils de guidage de processus[Jam98].

Ce mémoire s'inscrit dans le cadre du développement des systèmes d'information où le processus sous-jacent à toute méthodologie, peut être vu comme un processus de génie logiciel, c'est à dire comme une succession de transformations de produits, jusqu'à l'obtention du produit final qui est le système d'information automatisé.

iii PROBLEMATIQUE ET OBJECTIFS

La variété des méthodes et leur évolution dans le temps, en plus de la non-existence d'une méthode valable pour tous les projets ont suscité l'intérêt de connaître la capacité d'une méthode à systématiser les activités de développement afin de faciliter l'appréciation des similitudes et des différences dans le but de rapprocher, standardiser et harmoniser les méthodes.

Certains travaux se sont intéresses à la question : *« doit-on jeter aux orties tous les efforts de structuration de la pensée qui ont conduit aux méthodes actuelles et essayer de trouver une méthode standard ? »*[Buw92]. La majorité de ces travaux sont d'accord sur le fait que cette attitude est aussi irresponsable que celui qui réinvente la roue à chaque fois qu'il en a besoin. La réalité est que la plupart des méthodes actuelles ont introduit des concepts porteurs originaux mais aucune d'elles n'est dépourvue de défauts. De ce fait on peut dire que la méthode idéale n'existe pas encore et quelle n'existera peut être jamais.

De leur côté, les supports de modélisation ont subi des évolutions dues d'une part, à l'acquisition du savoir-faire et d'autre part, aux évolutions de la technologie. Ces supports sont formés d'un ensemble d'outils simples connus sous le nom d'outils CASE *(Computer Aided Software Engineering)* dont les majeurs bénéfices sont : La documentation, la vérification des spécifications, l'intégration des divers modèles et occasionnellement la génération automatique des nouvelles spécifications. D'un autre côté, les supports de modélisation actuels n'ont pas considéré la possibilité de passage d'une stratégie à une autre au cours du développement ou lors de la maintenance d'une application, ce qui s'avère encore difficile à réaliser voir même impossible à réaliser dans certains cas.

Si dans le même projet, les développeurs maîtrisent des formalismes différents, il serait impossible de prendre en compte les habitudes de travail de ces développeurs et ces derniers seront contraints de s'adapter à des formalismes différents, ce qui réduirait leur efficacité et leur productivité.

Théoriquement une méthode doit être choisie selon son adéquation au problème posé, les outils de conception sont ensuite sélectionnés parmi les outils dédiés à la méthode choisie selon la situation, or les méthodes sont inégalement outillées. Depuis les années 80 et avec la tendance de diversification de méthodes constatée, on observe une évolution des supports de modélisation, jusque là dédiés à une seule méthode, vers des supports multi-méthodes, dans lesquels on dispose d'un niveau d'abstraction additionnel dit : « méta-niveau » qui permet de modéliser les méthodes et offre les outils appropriés permettant de faciliter l'application de la méthode appropriée à une situation donnée[Jam98].

Récemment un nouveau sujet de recherche a émergé pour consolider la notion d'approche situationnelle dont le principal objectif est de définir des outils permettant de proposer une stratégie de développement pour chaque type de projet. Ces outils sont connus sous le nom d'outils CAME*(Computer Aided Methodology Engineering)*.

Le problème traité dans cette thèse est de considérer qu'il n'existe pas une unification de vues et de méthodes de développement des différents concepteurs et que la transformation des produits peut faire appel à plusieurs méthodes. Le produit intermédiaire deviendrait alors le résultat d'un fragment de méthode ; des problèmes de conformité et de cohérence entre autres viennent alors se poser pour l'intégration des différents produits résultants.

Nos nous proposons :

- D'analyser et d'expliciter les problèmes afférents à cette approche.
- De Définir une esquisse de démarche permettant de résoudre une partie de ces problèmes.

iv LE PLAN DU MEMOIRE

Le plan proposé pour développer ce mémoire est composé de trois grandes parties :

1- Une partie introductive comprenant les définitions et les notions fondamentales sur le développement et la modélisation des systèmes d'information ainsi que l'évolution historique des approches de développement et des supports de modélisation.

2- une deuxième partie comprenant l'état de l'art composée :

- D'un cadre de référence pour l'ingénierie des processus impliquant quatre mondes en interaction :
  - Le monde du sujet.
  - Le monde du système.
  - Le monde du développement.
  - Le monde de l'usage.

chacun de ces mondes est associé à un aspect particulier de leur problématique, permettant la compréhension, l'évaluation et l'amélioration des processus.

- Une introduction à l'ingénierie des méthodes qui est définie comme l'approche systématique pour le développement, la mise en œuvre, la maintenance et l'abandon des méthodes. Trois alternatives y prédominent actuellement :
  - Le développement d'une méthode universelle.
  - Le développement de méthodes par type d'application.
  - La méta-modélisation permettant d'instancier plusieurs modèles de méthodes à partir d'un méta-modèle.
- La présentation du méta-modèle NATURE.

3- une troisième partie comprenant une esquisse d'une démarche de développement de systèmes d'information par fragments de méthodes permettant l'activité de transformation et la navigation entre les méthodes et composée de quatre principales étapes :

- La caractérisation du projet en fonction des facteurs de contingence.
- La sélection des fragments de méthodes.
- L'assemblage des fragments de méthodes.
- L'amélioration de la méthode au cours du développement.

Et enfin une conclusion générale faisant le bilan du travail et les perspectives pour le futur.

Chapitre I

LE DEVELOPPEMENT DES SYSTEMES D'INFORMATION

Les systèmes d'information sont devenus au fil des années des outils stratégiques pour les organisations, l'importance de tels systèmes a suscité l'intérêt de les comprendre. C'est ainsi que COLETTE ROLLAND les définit comme étant un *«artefact, un objet artificiel, greffé sur un objet naturel qui peut être une organisation ».*

Ils sont conçus pour mémoriser un ensemble d'images de l'objet réel à différents moments de sa vie ; Ces images doivent être accessibles par les partenaires qui s'en servent pour décider des actions à entreprendre dans les meilleures conditions [Rfb88]. De tels systèmes ont pour objectif d'informer sur l'ensemble des phénomènes réels qui sont constatés dans l'organisation. Ils apparaissent pour l'organisation, comme des outils nécessaires pour l'amélioration de la prise de décision qui mérite d'être informatisée[Bub86].

1.1 DEFINITIONS

Un système d'information d'une organisation est un ensemble formé :

- De collections de données, représentation partielles, en parties 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 S.I, ces règles doivent être connues des acteurs qui utilisent le S.I. 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.

Cet artefact supporte un réseau de flux d'informations nécessaires pour organiser, mettre en œuvre, gérer et maintenir les activités d'une organisation[Rol88].

Le système d'information est à la fois passif en tant que base d'information de l'entreprise et actif par les règles de gestion qu'il leur applique. Sa composante informatique s'enrichit d'un ensemble de procédures d'acquisition , de stockage, de transformations et de recherche de données pertinentes, représentatives de la réalité[Pad98].

1.2 LE CYCLE DE VIE D'UN SYSTÈME D'INFORMATION

Comme tout logiciel, le cycle de vie d'un système d'information comporte trois niveaux de construction[Fig 1.1].

1. **NIVEAU CONCEPTUEL**

Consiste à décrire dans un formalisme compréhensible par l'informaticien les objectifs réels et précis du projet, analyser les besoins et établir un schéma conceptuel du futur S.I pour conclure sur une évaluation de la faisabilité fonctionnelle et technique du S.I.

Connue aussi sous le nom d'ingénierie des besoins, couramment utilisée pour décrire le processus de représentation des connaissances indépendamment de toute considération technique. Elle est réalisée par les utilisateurs du S.I est menée en collaboration avec les concepteurs.

2. **NIVEAU LOGIQUE**

constitué de deux étapes :

- la première consiste à représenter le système dans sa globalité et rendre le schéma conceptuel définitif avec une documentation détaillée, elle décrit ce que le système doit faire en faisant complète abstraction de la façon de le faire.
- La deuxième a pour objectif de constituer un cahier de charges de réalisation interne permettant la production de logiciel ou la réutilisation de logiciels existants.

Connue aussi sous le nom d'ingénierie des systèmes, utilisée pour décrire le processus d'élaboration du schéma interne du S.I à partir du schéma conceptuel.

3. **NIVEAU INERNE**

Ce niveau est composé de trois phases :

- **Le codage :** qui consiste à traduire les spécifications du cahier de charges en programmes informatiques. Cette étape est suivie par des tests unitaires qui permettent de vérifier la logique des programmes développés.

- **La validation :** cette phase très importante, a pour objet de vérifier la cohérence du logiciel produit par rapport aux besoins spécifiés lors de la première phase.

- **La maintenance :** elle permet de gérer les perturbations localisées dans le fonctionnement du système.

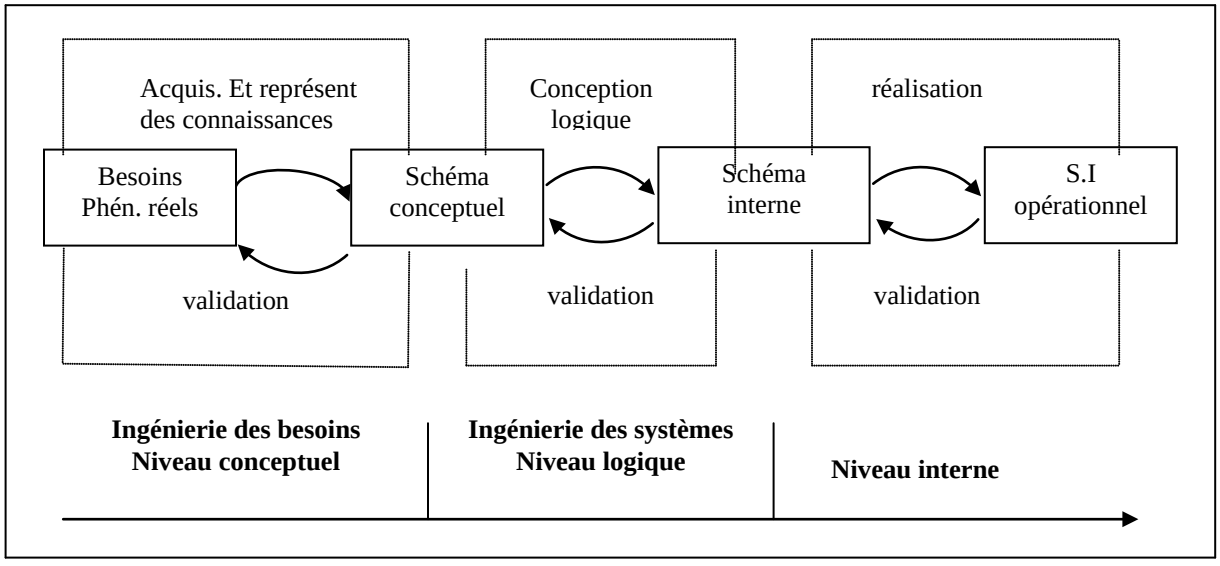


Figure 1.1 : Processus de développement d'un Système d'information.

1.3 LES METHODES DE DEVELOPPEMENT DES SYSTEMES D'INFORMATION

Les systèmes d'informations deviennent de nos jours des systèmes stratégiques dans les entreprises et leur qualité devient également un facteur critique. Le développement de bonnes démarches de conception est pour cette raison devenu un sujet de recherche à part entière. De nombreuses définitions de méthodes ont été proposées dans la littérature, sans qu'un véritable consensus ne soit établi. Toutefois, elles s'accordent sur le fait qu'une méthode met en œuvre deux composants principaux[Ben98].

- Un ensemble de modèles qui interprètent explicitement la compréhension d'une situation et définissent les concepts permettant de décrire les schémas conceptuels appelés aussi produits.

- Une démarche prescrivant plus ou moins complètement le processus opératoire à suivre pour obtenir le schéma conceptuel à partir de l'énoncé des besoins et grâce auquel s'effectue le travail de modélisation et de description du S.I.

Une méthode ne décrit pas une solution mais comment aboutir au résultat, elle se concrétise par un modèle de réflexion exprimant comment procéder pour spécifier et concevoir. Les premières définitions de méthodes étaient concentrées sur l'aspect produit d'une méthode. Depuis les années 90, les définitions de méthodes intègrent en plus l'aspect processus comme composant dans la définition d'une méthode.

Deux autres composants peuvent se greffer :

- Les concepts et le formalisme associés doivent être sémantiquement puissants. Ils doivent se présenter de façon structurée afin d'être appréhendés et facilement mis en œuvre au cours du cycle de vie pour permettre de décrire le produit de conception.

- La méthode doit être dotée d'outils permettant de prévoir et de réaliser les extrapolations des phases de développement et doit se montrer suffisamment souple et adaptable pour pouvoir être utilisée dans de diverses circonstances.

Les caractéristiques qui appuient la décision de retenir ou non une méthode sont diverses et leur importance diffère d'une application à une autre ou d'une entreprise à une autre. Toutefois chaque méthode doit rendre compte des considérations suivantes[Chr95]:

Une méthode doit permettre d'obtenir des produits de qualité conformes aux besoins des utilisateurs, comportant un faible nombre d'erreurs faciles à détecter dans les délais fixés et afficher clairement ses limites d'utilisation qui sont le plus souvent le fruit de l'expérience des utilisateurs de la méthode et doit s'appuyer sur des outils qui permettent de formaliser les résultats de l'analyse et de conception. La force et l'espérance de vie d'une méthode sont étroitement liées aux outils disponibles pour sa mise en œuvre. Ce point doit être particulièrement pris en compte lors du choix d'une méthode.

1.3.1 LES PRODUITS D'UNE METHODE

Le produit désigne le résultat d'une étape quelconque du processus, il peut être un modèle d'analyse ou de conception, un document, un code ou un résultat intermédiaire.
Un modèle sert à décrire visuellement un système qui deviendra réalité après sa conception. Quatre formalismes sont généralement utiles dans le développement des systèmes d'information, chaque catégorie a ses caractéristiques, ses concepts et des terminologies spécifiques au langage utilisé[Jam98].

- **Les modèles libres**

Ces modèles sont limités dans leur structure par l'imagination de leurs utilisateurs ; ils sont peu précis mais très pratiques. Cependant, ces modèles ne peuvent être vérifiés automatiquement car ils n'utilisent pas de concepts formalisés. Des exemples typiques de ce type de modèle sont les modèles verbaux et d'illustration.

- **Les modèles structurés**

Ces modèles sont généralement représentés par des diagrammes, des tables, des matrices ou des textes structurés; ils possèdent des types et des concepts prédéfinis, de plus leur utilisation doit être conforme à une certaine syntaxe qui régit l'utilisation des différents concepts. Ces modèles peuvent être analysés et contrôlés.

- **Les modèles mathématiques**

Ces modèles sont définis par l'emploi des langages basés sur les constructeurs mathématiques. Les spécifications formelles représentent l'exemple typique de ce genre de modèles.

- **Les modèles dynamiques**

Ces modèles peuvent être des modèles de simulation ou de prototypage. Ils offrent une facilité d'expérimentation et peuvent être utilisés pour analyser le comportement dynamique de certaines Propriétés du système.

1.3.2 LE PROCESSUS

le processus logiciel ou tout simplement processus est défini comme un ensemble intégré d'activités requises pour transformer les besoins de l'utilisateur en un logiciel fonctionnel y compris les activités nécessaires à la maintenance de ce logiciel [Feh91]. Sa formalisation vise à décrire de façon détaillée, à l'aide d'un formalisme exécutable, le mode de production d'un type de logiciel[Ost87].

Cette définition utile, mais peu être trop spécifique, des processus de développement de logiciel, n'est pas complètement acceptée par la communauté des systèmes d'information. De plus, le terme « *processus* » est également utilisé dans d'autres contextes qui ne prennent pas à l'effort de recherche commun engagé pour aboutir à une terminologie standard. Néanmoins, on peut affirmer que le terme processus est employé dans la majorité des cas dans le sens : « *d'un ensemble d'activités inter-reliées et menées dans le but de définir un produit* »[Fra91] .

Le processus est modélisé par une succession d'étapes enrichissant de manière formelle la nature et le contenu des informations issues de l'expression des besoins pour aboutir à une représentation proche de l'informatique[Pad98]. Chaque étape du processus s'appuie sur :

- L'abstraction qui fabrique des vues externes exprimant le quoi faire.
- La traduction qui aboutit à des vues internes exprimant le comment cela sera fait.
- Le contrôle, la validation et la décision.

l'idée que le processus de développement de systèmes est un processus de prise de décision est devenue aujourd'hui une alternative au paradigme classique du processus vu comme une suite d'activités.

1.4 EVOLUTION DES APPROCHES DE DEVELOPPEMENT

Les méthodes de développement sont diverses ; l'approche couramment considérée pour la comparaison de méthodes a été de trouver les propriétés et les caractéristiques communes dans les méthodes. Les approches de développement que nous présentons sont le fruit de l'évolution historique de la technologie, des outils et des langages utilisés. elles tiennent compte de trois aspects du système d'information.

- **Les traitements** : qui sont utilisés pour la consultation et la transformation des données
- **Les données** : qui matérialisent les entités statiques
- **Les comportements** : qui indiquent les conditions de déclenchement et enchaînement des traitements.

1.4.1 LES APPROCHES CARTESIENNES (Années 70)

les méthodes cartésiennes ou analytiques mettent l'accent sur la démarche de conception qui décrit la manière de conduire et de dérouler le processus de conception, elles se basent sur une stratégie de décomposition hiérarchique du problème à résoudre en fonctions (approche fonctionnelle) et en modules où chacun est un raffinage du précédent (approche descendante). Le rôle de cette catégorie de méthodes est d'aider le concepteur à passer d'une vision fonctionnelle globale à la description détaillée du processus de traitement qu'elle constitue. Parmi ces approches on cite :

- La méthode **SADT** : Structured Analysis and Design Technique [ROSS 77]
- La méthode **SA/SD** : Structured Analysis/Structured Design [YOURDON 79]
- La méthode **JSD/JSP** : Jackson Structured Design/Jackson Structured Program [Jackson 75]

Le S.I est abordé par les fonctions qu'il doit assurer, plutôt que par les données qu'il doit gérer. La décomposition fonctionnelle garantit la conformité de la conception aux besoins des utilisateurs et la démarche naturelle utilisée pour aborder les problèmes et rend ces méthodes très simples. Toutefois, les règles de décomposition des fonctions sont non explicites et produisent des hiérarchies de décomposition selon les analystes et les fonctions sont volatiles et susceptibles de changer et d'évoluer selon les besoins des utilisateurs.

1.4.2 LES APPROCHES SYSTEMIQUES (Années 80)

La stratégie de ces approches considère le système d'information comme un système complexe doté d'une autonomie de fonctionnement et interagissant avec son environnement. La conception globale du S.I. est basée sur la recherche d'éléments pertinents de l'organisation et leurs relations. La principale caractéristique de ces approches est la séparation nette entre les données et les traitements avec des modèles séparés pour décrire les deux aspects.

Dans les méthodes systémiques le S.I est abordé à travers l'organisation des systèmes constituants de l'entreprise dont il fait lui même partie, celui-ci est donc à la fois une image de la réalité et une abstraite collection de données et de traitements[Pad98]. Les méthodes les plus connues sont :

- La méthode **MERISE** [TARDIEU 83]
- La méthode **IDA** [BODART 83]
- La méthode **REMORA** [ROLLAND 88]
- La méthode **NIAM** [HABRIAS 88]

La perception globale de ces approches permet une vue moins redondante et plus cohérente des informations et une capacité de décrire des systèmes plus complexes , mais la dichotomie données–traitements conduit à une impossibilité de validation mutuelle des deux modèles correspondants.

On peut admettre que cette séparation était fondée en ce qui concerne les phases basses du développement(analyse organique, programmation et test), mais ce principe a dûment contaminé les phases hautes qui se doivent pourtant de rester au plus près du discours de l'utilisateur. Ainsi si on met l'accent sur les données, on préserve une bonne part d'expression disponible dans le discours de l'utilisateur mais on s'oblige à traiter des opérations de façon globale et détachée des données. Si on met l'accent sur les traitements, on obtient une solution fonctionnelle claire mais au prix de l'abstraction des données[Chr95].

1.4.3 LES APPROCHES OBJET (Années 90)

l'apparition des langages dits « *orientés objets* » a nécessité une autre manière de concevoir les systèmes d'information. Le paradigme objet hérite légitimement du paradigme systémique, ce qui lui permet de fournir de précieux outils pour aborder la modélisation et combler le manque des méthodes en

ce qui concerne les aspects globaux du système d'information. l'objet est plus que la donnée ou l'entité du modèle entité/association ; il est un modèle exact et complet d'un objet du monde réel[Vau93].

Les approches orientées objets se basent sur une spécification détaillée du S.I. centrée sur la notion d'objet regroupant structure de données et traitements et sur une spécification globale centrée sur les liaisons statiques et dynamiques inter-objets. Cette catégorie est une synthèse des méthodes cartésiennes et systémiques, Il ne s'agit plus d'élaborer des algorithmes en séparant les données et les traitements mais de distinguer des ensembles qui regroupent les actions et les objets sur lesquels elles s'appliquent. Les méthodes les plus connues et les plus utilisées sont :

- La méthode **OOD** : Object Oriented Design [BOOCH 86]
- La méthode **OMT** : Object Modelling Techniques [RUMBAUGH 91]
- La méthode **OOA** : Object Oriented Analysis [SHLAER-MELLOR 93]
- La méthode **OOSE** : Object Oriented Software Engineering [JACOBSON 92]

La puissance de la démarche orientée objet réside dans la linéarité et la continuité de l'implémentation. contrairement aux démarches traditionnelles, à aucun moment le développement ne rencontre de rupture, aucun changement de niveau n'oblige à remodeler la physionomie du système. Ces approches possèdent une grande capacité à modéliser des objets complexes et d'exprimer de façon intégrée leur dynamique. Celles-ci réutilisent largement les techniques, concepts et parties de démarches d'autres méthodes.

1.5 EVOLUTION DES SUPPORTS DE MODELISATION

La plupart des méthodes de conception de systèmes d'information ont été renforcées par des supports informatisés de modélisation afin de faciliter la tâche des développeurs durant les projets informatiques. Un support de modélisation est un ensemble de composants logiciels, apportant une assistance aux concepteurs, organisés pour couvrir tout ou partie du cycle de production et d'évolution des logiciels[Fug93].

Plusieurs classifications de supports de modélisation ont été faites. Nous retenons celle qui cadre avec l'évolution des approches de développement.

1.5.1 LES SUPPORTS DE PREMIERE GENERATION

Les supports de cette génération offrent des éditeurs de modèles structurés pour la saisie et la mise à jour des modèles et proposent des interfaces graphiques pour la manipulation de diagrammes basées sur des modèles traditionnels tels que entité-association ou flux de données. On retrouve parfois certaines fonctionnalités qui enrichissent ces supports :

- La validation des spécifications pour satisfaire les règles de contrôle spécifiques au modèle utilisé.
- L'amélioration de la représentation des modèles.
- Et la transformation automatique des spécifications d'une application en spécifications exécutables.

Le mode de représentation des produits logiciels(le schémas de base) est propre à chaque support, il est défini en fonction des besoins de chacun des composants logiciels disponibles. De ce fait, les supports de cette première génération apparaissent comme fermés et non adaptables aux utilisateurs qui se voient contraints d'abandonner leurs méthodes de travail au profit de celles imposées par l'outil.

1.5.2 LES SUPPORTS DE DEUXIEME GENERATION

Les supports de cette génération sont basés sur un cadre flexible qui fournit un mécanisme d'intégration efficace d'outils, facilite l'échange d'information et permet de mieux s'adapter à la variété des méthodes.

Cette génération est marquée par l'introduction de la méta-modélisation. Les méta-outils sont des CASE génériques qui permettent de décrire différents formalismes en utilisant un même méta-formalisme. Leur adoption apparaît comme le meilleur chemin à suivre particulièrement pour les grandes organisations qui sont dans le besoin de diversifier les techniques de modélisation au cours du développement des projets. Au niveau exploitation des modèles ces supports restent faibles, ils sont qualifiés de « *cahiers de brouillon* » sophistiqués bénéficiant de l'apport de l'informatique pour la manipulation et le stockage des information[Rol92].

1.5.3 LES SUPPORTS DE TROISIEME GENERATION

Face au besoin d'automatiser de plus en plus les activités de modélisation et afin d'améliorer l'exploitation des modèles; deux principales fonctionnalités sont attendues de cette nouvelle génération :

- **L'indépendance aux méthodes**

Les supports de cette génération doivent montrer une plus grande adaptabilité à la flexibilité des méthodes de développement et leur évolution vers les approches situationnelles.

- **Le guidage des processus**

La troisième génération va être marquée par l'introduction de l'intelligence artificielle, et on en attend une meilleure efficacité et gestion du processus de développement, un bon degré de réutilisation, une plus grande facilité dans le travail des développeurs et une meilleure qualité des produits logiciels.

On parle alors de supports « *centrés processus* » qui se proposent de piloter et d'assister les développeurs en mémorisant et exploitant, sous forme de modèles de processus, les informations et les règles qui définissent les méthodes[Col91].

Chapitre II

L'INGENIERIE DES PROCESSUS

Les méthodologies traditionnelles utilisées dans les systèmes d'information ont été élaborées avec hasard. Les modèles de processus sous-jacents sont l'expression de l'expérience des développeurs; il n'est donc pas possible de savoir comment les modèles de processus ont été générés.

L'accroissement de la productivité des équipes de développement et l'amélioration de la qualité des systèmes développés ne sont possibles que grâce à l'amélioration des processus de développement. La communauté de l'ingénierie des systèmes d'information s'est depuis peu de temps penchée sur ce problème et a adopté une approche d'ingénierie des méthodes dans laquelle le modèle de processus est considéré comme un produit de l'ingénierie des processus. L'idée majeure est celle de la configuration d'une méthode à partir de fragments de méthodes prédéfinis et validés. Cette nouvelle façon de faire est justifiée par l'étude des nouvelles méthodes et notamment les méthodes orientées objet qui réutilisent largement les techniques, concepts et parties de démarches d'autres méthodes.

L'ingénierie des processus comprend par définition toutes les activités permettant l'amélioration des processus et par conséquent, pour optimiser ces processus, l'intérêt devrait se déplacer vers la compréhension, l'évaluation et l'amélioration des différents aspects de cette ingénierie. il est donc nécessaire de se doter d'un cadre de référence qui permette de comprendre leur problématique et d'apprécier les solutions proposées. Le cadre de référence proposé est inspiré de celui du projet DAIDA [Jar92] pour l'ingénierie des systèmes d'information et de son adaptation pour l'ingénierie des besoins dans le projet NATURE [Jar93]. Il comporte quatre mondes en interaction[Fig2.1] :

- le monde du sujet
- le monde du système
- le monde du développement
- le monde de l'usage

Chacun de ces mondes est associé à un aspect particulier de cette problématique[Rol96].

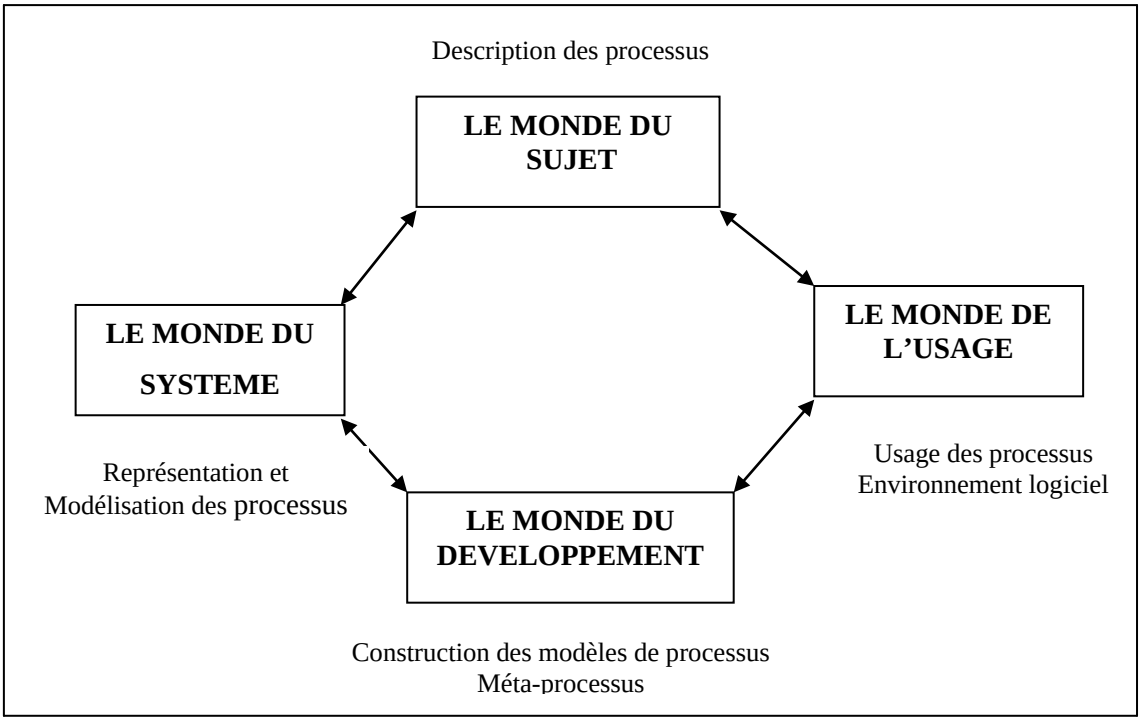


Figure 2.1 : Cadre de référence pour l'ingénierie des processus.

2.1 LE MONDE DU SUJET

Ce monde comporte la terminologie ainsi que les différents niveaux de description des processus. Il correspond à la définition des processus de développement de systèmes et de quelle manière on peut les décrire. Il existe trois niveaux d'abstraction : le niveau des instances (les processus), le niveau des types (modèles de processus) et celui des méta-types (le méta-modèle de processus).

2.1.1 LE PROCESSUS

Il se situe au niveau instance et est composé d'instances d'actions permettant de modifier les instances de produit.

Exemple : Séquence d'actions pour une application de prêt de livres dans une bibliothèque.

1 – Créer l'entité-type « ABONNE ».

2 - Créer l'entité-type « EXEMPLAIRE ».

3 – Créer la relation-type « EMPRUNTE ».

Cette séquence mentionne les actions qui sont menées et l'ordre dans lequel elles sont menées.

2.1.2 LE MODELE DE PROCESSUS

Il se situe au niveau type. Il fait référence à des actions type représentant chacune un ensemble d'instances.

Exemple : Séquence d'actions-type servant de modèle de processus pour la séquence précédente.

1 – Créer les Entités–Type.

2 - Créer les Relations-Type.

en conséquence, un processus est donc une instanciation d'un modèle de processus qui est exécuté.

2.1.3 LE META-MODELE DE PROCESSUS

Il se situe au niveau méta-type. Il décrit un ensemble de modèles de processus et permet la représentation de concepts génériques.

Exemple : Séquence d'actions génériques servant de méta-modèle de processus pour les séquences précédentes.

1 - Identifier Actions.

2 - Ordonner Actions.

Cette séquence correspond au méta-modèle de processus et aux processus décrits respectivement dans les deux exemples précédents.

2.2 LE MONDE DU SYSTEME

Le monde du système concerne les aspects de représentation et de modélisation des processus. La compréhension et l'amélioration des processus passent par la mise au point de modèles de représentation. De nombreux travaux ont porté sur l'amélioration des modèles existants (démarches génériques) ; mais la limitation du niveau de granularité dans la spécification des processus de développement et la pauvreté d 'expression des enchaînements de tâches rendent ces démarches inadaptées à la modélisation fine des processus.

Pour satisfaire les objectifs récents de l'ingénierie des processus, il faut disposer de modèles ayant un pouvoir d'expression plus grand. On assiste à la définition de modèles génériques, voire de méta-modèles. Ces modèles ont initialement cherché à traduire la caractéristique essentielle des processus à savoir les activités qui les composent et leur enchaînement. Ensuite on a voulu modéliser l'évolution du produit et de coupler les états du produit aux activités qui génèrent ces états. Enfin on s'est penché sur les intentions motivant l'exécution des activités, leur enchaînement et les états du produit qu'elles génèrent.

Un modèle de processus est la description d'un type de processus. Il est utilisé, le plus souvent, dans un but prescriptif pour prescrire « *comment les choses doivent, devraient, pourraient être faîtes* ». Contrairement au processus lui-même qui décrit exactement ce qui se passe « une instanciation d'un modèle de processus ». Un modèle de processus est une anticipation plus ou moins approximative de ce à quoi ressemblera le processus[Rol97].

Plusieurs modèles de processus connus sous le nom de modèles de cycle de vie ont été proposés pour fournir des traits généraux. Ces modèles permettent d'identifier les activités d'un projet et de décrire leurs interactions. Ils ont montré l'importance d'une définition explicite des processus logiciel. Cependant, ils présentent une forte limitation inhérente à leur généralité ; ils ne précisent pas les détails du processus logiciels, qui sont pourtant d'une importance fondamentale pour une bonne conduite des projets logiciels.

On peut classer les approches en deux groupes apparus chronologiquement :
- Celles qui ont prévalu jusqu'au milieu des années 80, visant à prescrire l'organisation en étapes du projet du cycle de développement d'un système et qui ont conduit aux démarches génériques qui se limitent à des recommandations d'organisation globale du cycle de développement.
- Les approches plus récentes, sous-jacentes aux nouveaux modèles de processus qui ont des objectifs plus ambitieux tels que la capture d'heuristiques de développement, le contrôle et /ou le guidage intelligent et interactif des développeurs, la trace des activités menées et de leur justifications.

2.2.1 LES DEMARCHES GENERIQUES

Sont des démarches globales spécifiant un enchaînement linéaire de tâches à gros niveau de granularité, qui ne précisent pas dans le détail toutes les activités réalisées dans un processus [Ali99b]. De plus les enchaînements de tâches qu'elles expriment sont simples, limités à de simples successions, voire dans certains cas à des relations de parallélisme.

Le projet est décomposé en une série d'étapes selon une démarche qui part du besoin pour aboutir au produit fini qu'est le S.I. la notion de cycle représente l'itération des projets décrivant de nouveaux besoins engendrés le système développé. La présentation retenue fait apparaître trois aspects de ces démarches :

- des propositions d'organisation du cycle de vie du système
- le rôle du prototypage
- le principe transformationnel

2.2.1.1 LA DEMARCHE EN CASCADE

Introduite par W.W.ROYE en1970 qui reconnaît la validité du découpage des processus en six étapes où chacune d'elles est suivie d'une documentation détaillée[Fig 2.2].

1. L'étude des besoins du système (system requirements)
2. L'étude des besoins du logiciel (software requirements)
3. L'analyse (analysis)
4. La conception des programmes (program design)
5. La programmation (coding)
6. Le test (testing)

Mais il soutient que la mise en œuvre de ces étapes est souvent source d'itérations longues et coûteuses. Pour remédier à ce problème, il suggère d'insérer une étape de conception préliminaire (preliminary design) entre l'étape d'études des besoins du logiciel et celle de l'analyse dans le but de mettre en avant les contraintes techniques pouvant influencer l'analyse.

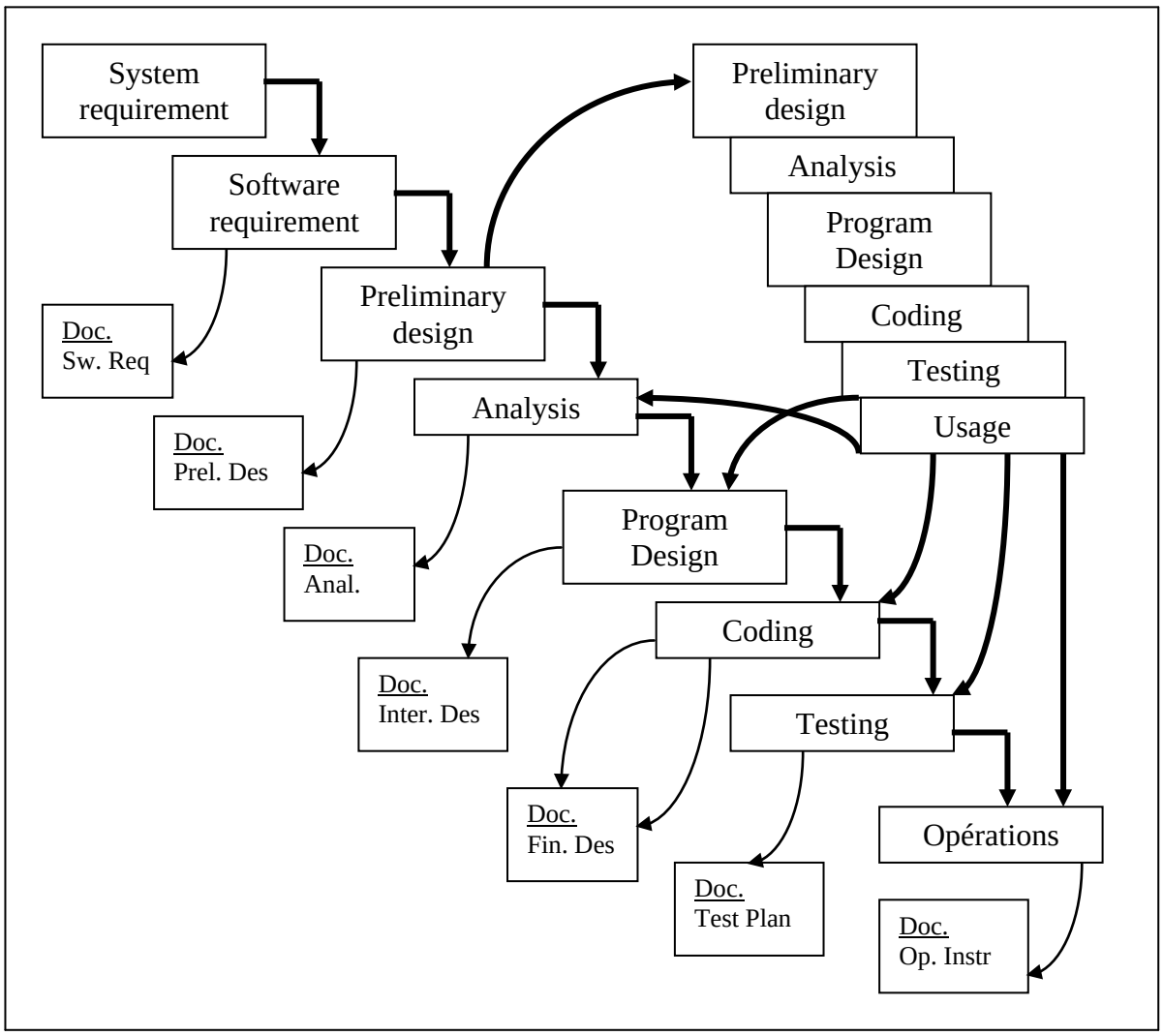


Figure 2.2 : la démarche en Cascade

2.2.1.2 LA DEMARCHE PAR PROTOTYPAGE

Le prototypage est une technique qui permet d'obtenir assez rapidement une première version implémentable d'un système à partir de spécifications d'un sous-système représentatif du système projeté. En procédant à des essais sur le prototype, le développeur peut, en détectant les erreurs de fonctionnement, les réajuster. Il peut aussi le compléter de différentes manières, en fonction de l'extension du sous-système représentatif qui a été pris pour cible, jusqu'à obtenir le système complet.

Le prototypage a été fréquemment utilisé dans les ateliers de génie logiciel et en bases de données. Il a séduit les concepteurs surtout comme moyen de validation dans la mise en œuvre de systèmes complexes. Les essais sur le prototype permettent d'étendre la compréhension des besoins.

Certains auteurs définissent le sous système représentatif comme la partie de l'énoncé des besoins des utilisateurs la moins comprise, d'autres auteurs définissent le prototype sur la partie la mieux comprise des besoins des utilisateurs. C'est le prototype qui est complété au fur et à mesure que la compréhension des besoins des utilisateurs augmente.

2.2.1.3 LA DEMARCHE EN SPIRALE

En 1988 BOHEM définit la démarche en spirale en s'inspirant en même temps de la démarche en cascade et du prototypage. Cette démarche représente le processus de développement comme une succession de boucles[Fig 2.3], chacune donnant lieu à l'exécution successive de cinq activités de bases :

1. Diagnostic des résultats de la boucle précédente
2. Identification des objectifs, moyens et contraintes
3. Evaluation des alternatives avec analyse de leur implication sur le développement et le comportement du système.
4. Développement de la meilleure alternative
5. Planification de la boucle suivante.

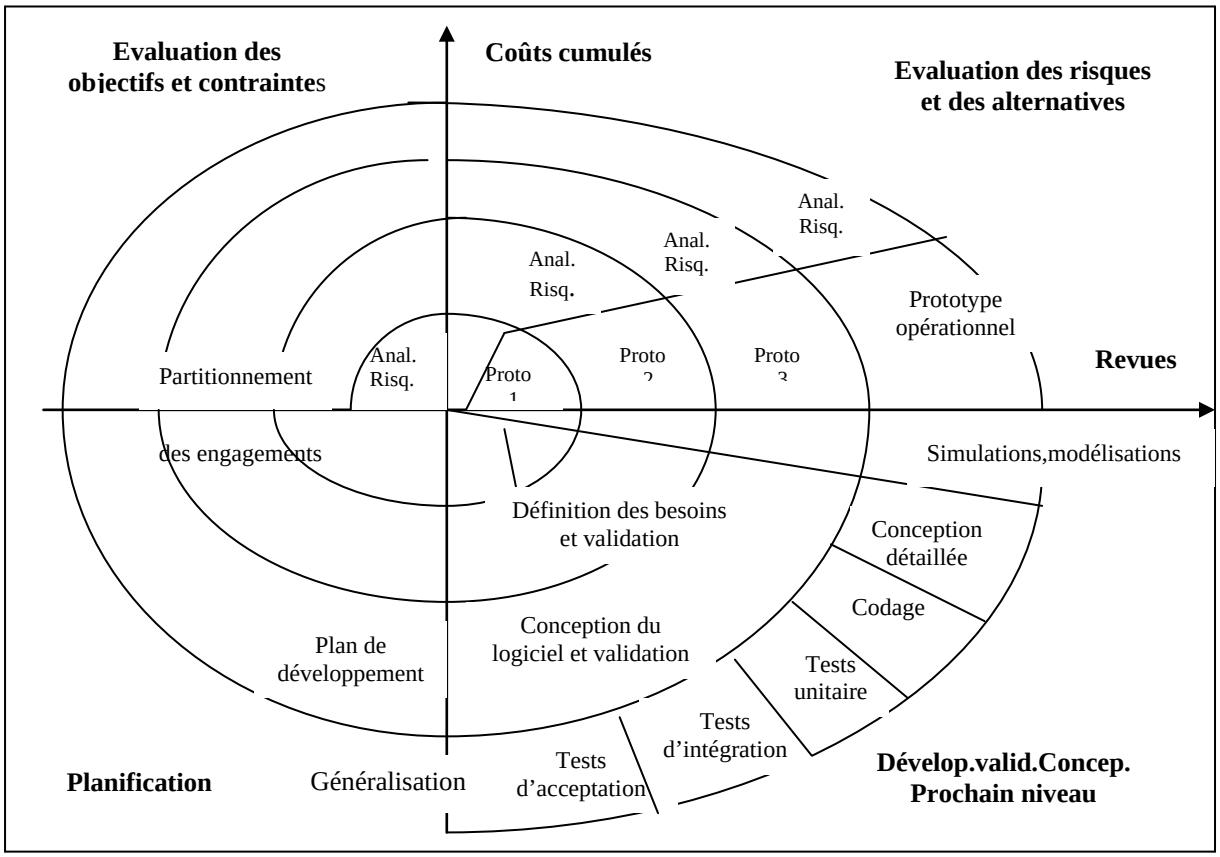


Figure 2.3 : la démarche en Spirale

Le prototypage est fait à la fin de chaque boucle afin de valider le sous système, jusqu'à la construction du système complet. La dimension radiale de la spirale représente la progression globale de développement et la dimension angulaire représente l'état d'avancement de chaque boucle dans le processus.

2.2.1.4 LA DEMARCHE EN FONTAINE

La démarche en la fontaine a été définie pour les environnements de développement orientés objet. Elle consiste en un processus incrémental qui permet de représenter non seulement le chevauchement des étapes de développement mais aussi les itérations entre les différentes étapes de développement[Fig 2.4].

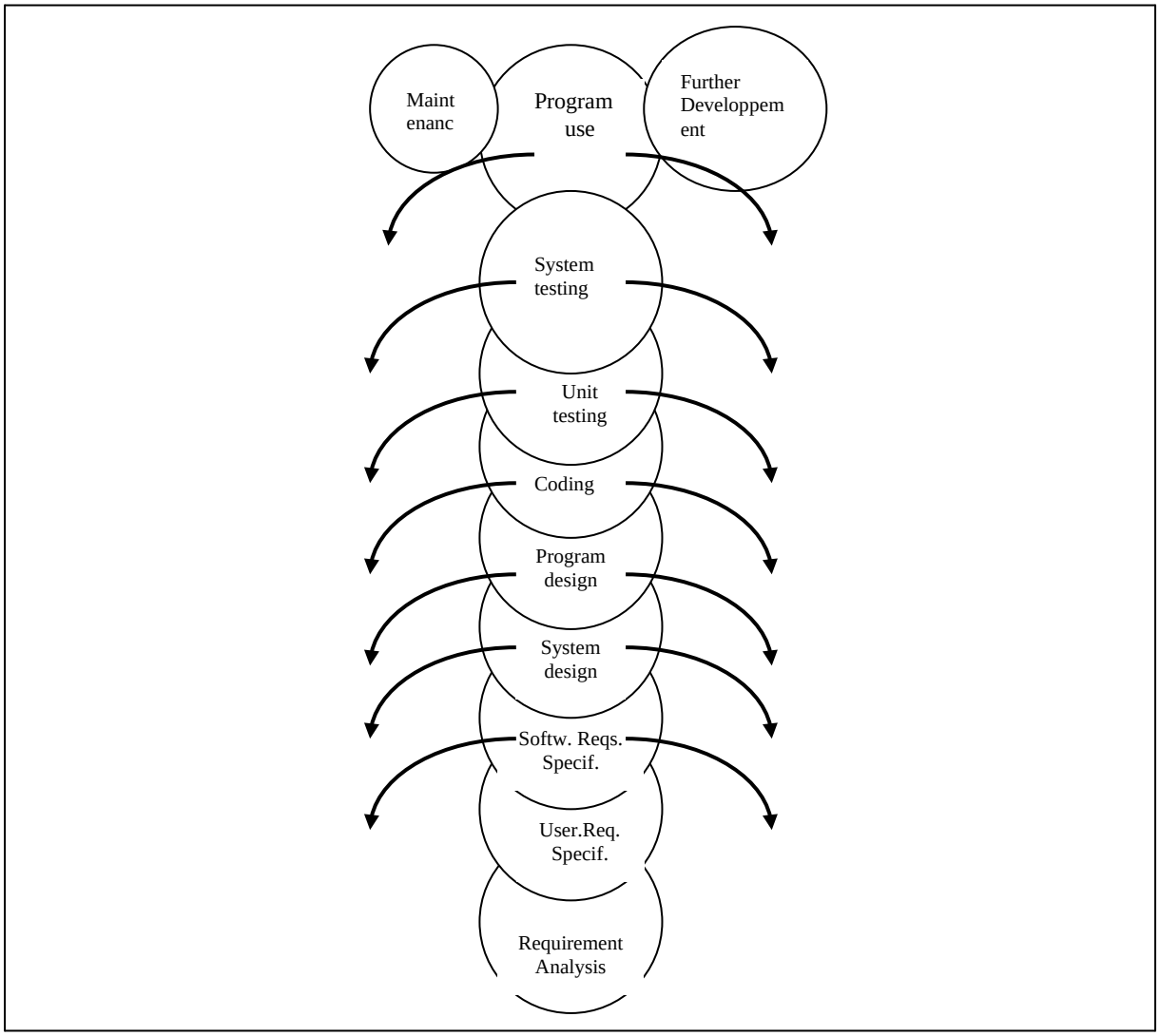


Figure 2.3 : La démarche en Fontaine

Le processus débute par une phase d'analyse des besoins (requirements analysis) et se poursuit par l'exécution des même phases que celle définies dans la démarche « en cascade ». Ce processus est répété jusqu'à ce que toutes les parties du système en cours de développement aient été développées. En fait dans cette démarche, sont combinées en même temps, la technique d'itération et celle du parallélisme.

2.2.2 LES MODELES DE PROCESSUS

En 1987 Dowson propose de classer les modèles de processus en trois catégories selon l'aspect prédominant dans la compréhension des processus [Dow 87].

- Les modèles orientés–activité.
- Les modèles orientés–produit.
- Les modèles orientés–decision.

En effet, les activités et les produits qu'elles manipulent et/ou génèrent sont indissociables ; de plus les activités réalisées dans un processus sont toujours motivées par une intention et la connaissance s'enrichit par la prise en compte des décisions. Les activités, les produits et les décisions sont donc bien trois notions fondamentales dans la modélisation des démarches.

2.2.2.1 LES MODELES ORIENTES-ACTIVITE

Ils correspondent à la manière intuitive de résoudre un problème qui consiste à établir un plan d'actions et à l'appliquer en exécutant des actions selon l'ordonnancement prescrit par le plan. Ces modèles voient le processus de développement comme des ensembles d'activités et de conditions régissant leur ordonnancement, ils se diffèrent les uns des autres par la variété des conditions qu'ils permettent d'exprimer ainsi que par les langages associés.

Ces modèles s'inspirent des démarches génériques (et notamment de la démarche en CASCADE), le paradigme sous-jacent de ces modèles est celui de la décomposition hiérarchique des activités. Dans la pratique, un grand nombre de démarches méthodologiques sont décrites avec cette approche. La méthode OMT par exemple, propose la séquence d'activités suivante pour accomplir la construction du produit d'analyse.

1) Faire une description initiale du problème
2) Construire un modèle statique
3) Développer un modèle dynamique
4) Construire un modèle fonctionnel

Le méta-modèle correspondant à cette classe est basé sur deux concepts : « *les activités* » et les « *conditions d'enchaînement des activités* ». Les modèles les plus représentatifs sont :

- Le modèle d'OSTERWEIL [OST 87]
- Le modèle SPM [WIL 88]
- Le modèle HFSP [KAT 89]
- Le modèle MASP [DEI 89]

2.2.2.2 LES MODELES ORIENTES-PRODUIT

Ces modèles se basent sur la notion d'état du produit et celle d'activité de transformation du produit qui peut en changer l'état. De plus c'est sur l'état du produit que sont spécifiées les conditions d'exécution des activités. Ainsi l'évolution du produit est représentée à l'aide de diagrammes de transitions d'état, ces transitions sont déclenchées par des événements survenant sur les différents produits. Ces modèles offrent l'avantage d'établir un lien explicite entre les activités et le produit résultant en ne mettant plus en avant les activités du processus, mais le résultat de ces activités sur le produit. Les deux modèles les plus connus sont :

- Le modèle EPM [HUM 89]
- Le modèle VIEW POINT [FIN 90]

Le modèle EPM considère les processus de développement comme des successions d'états des éléments du produit appelés entités. A un instant donné du processus, une entité est dans un état unique, par exemple un module de code peut être dans l'état (non existant , initial, en cours de développement, test, transfert ou final). Les transitions d'état sont déclenchées par des événements conditionnés par des expressions booléennes.

Le méta-modèle sous-jacent est basé sur les trois concepts : « *l'état du produit* », « *l'activité* » et « *transition d'état* ».

Ces modèles sont utiles pour tracer les transformations du produit, cependant il est difficile de construire un diagramme de transition d'états réaliste et capable de décrire, d'une façon convenable, tous les éléments du produit et permettre une estimation réelle du degré d'avancement du processus.

2.2.2.3 LES MODELES ORIENTES-DECISION

Selon ces modèles, les démarches ne spécifient plus uniquement les enchaînements d'activités ou d'états du produit, mais elles spécifient également les intentions motivant l'exécution des activités et leur enchaînement. Ainsi dans ce type de modèles, l'activité n'est pas ignorée mais elle est mise en arrière plan alors que la décision est mise en premier plan.

La décision comporte deux aspects : l'intention et l'approche: L'intention exprime un objectif que l'on souhaite atteindre c'est à dire l'expression d'un but, l'approche caractérise la manière d'atteindre le but. Par exemple, lors de la définition d'un système E/R, la création de l'entité-type « CLIENT » devient secondaire alors que l'intention de représenter les clients devient prédominante.

Le besoin d'informations sur les clients et la nécessité de les identifier sont des arguments en faveur de la prise de décision. Les modèles les plus connus sont :

- Le modèle de POTTS [POT 88]
- Le modèle PSD [DES 90]
- Le modèle DRL [LEE 91]
- Le modèle REMAP [RAM 92]

Le méta-modèle sous-jacent à ce paradigme est l'exemple du méta-modèle IBIS qui repose sur trois concepts : « *Le problème* »: défini comme une question qui nécessite d'être débattue avant d'être résolue, « *L'alternative* »: définie comme une solution répondant à la question posée par le problème et « *L'argument* » : qui défend ou s'oppose à l'alternative.

Les modèles orientés–decision permettent non seulement d'expliquer comment le processus se déroule mais aussi pourquoi les actions sont menées et les décisions prises ; Ils sont donc plus aptes à guider les processus que les autres types de modèles.

2.3 LE MONDE DU DEVELOPPEMENT

La plupart des travaux menés sur le thème des processus dans le domaine du génie logiciel, ce sont concentrés sur le développement et l'expérimentation de modèles de processus ; mais peu d'efforts ont étaient faits pour résoudre les problèmes d'un développement systématique de nouveaux modèles de processus[Dow93].

Le monde du développement correspond à l'ensemble des acteurs et des activités aboutissant à la construction de modèles de processus, c'est à dire des processus permettant de construire, d'améliorer et de faire évoluer les modèles de processus. La communauté de l'ingénierie des systèmes d'information a adopté une approche d'ingénierie des méthodes dans laquelle le processus est considéré comme un produit de l'ingénierie des processus.

Le chapitre suivant sera consacré à l'ingénierie des méthodes et fera le point sur les principales approches prédominantes actuellement en ingénierie des méthodes.

2.4 LE MONDE DE L'USAGE

Ce monde concerne les ateliers logiciels dédiés à l'exécution guidée des processus et de construction de méthodes. Il s'agit en premier lieu de guider l'exécution des processus de développement de systèmes et en second lieu, guider la construction des modèles de processus.

2.4.1 L'EXECUTION DES PROCESSUS

Les environnements centrés processus cherchent avant tout à contrôler l'usage des outils intervenant dans le développement et la maintenance et guider les ingénieurs logiciels dans la sélection des bonnes séquences d'outils. Dowson [DOW 93] propose d'articuler l'exécution du processus autour de trois domaines interagissant les uns avec les autres.

- **Le domaine de la modélisation des processus** : contient les modèles de processus, méthodes et fragments de démarches qui prescrivent l'exécution des processus.
- **Le domaine du développement** : est celui des activités réelles conduites par des personnes ou des machines pendant le projet.
- **Le domaine d'exécution des processus** : contient les occurrences des modèles et fragments de modèles de processus qui sont exécutés.

2.4.2 LA CONSTRUCTION DES MODELES DE PROCESSUS

Plusieurs prototypes d'environnement CAME (*Computer Aided Methodology Engineering*) ont été développés. A titre illustratif on présente l'environnement MENTOR[Fig 2.5], développé dans le cadre du projet NATURE, qui a été construit pour aider à la construction des modèles de processus, guider les activités intellectuelles et créatives de l'analyse et faire évoluer les modèles de processus à partir des expériences. Quand à l'architecture, elle comporte trois composants :

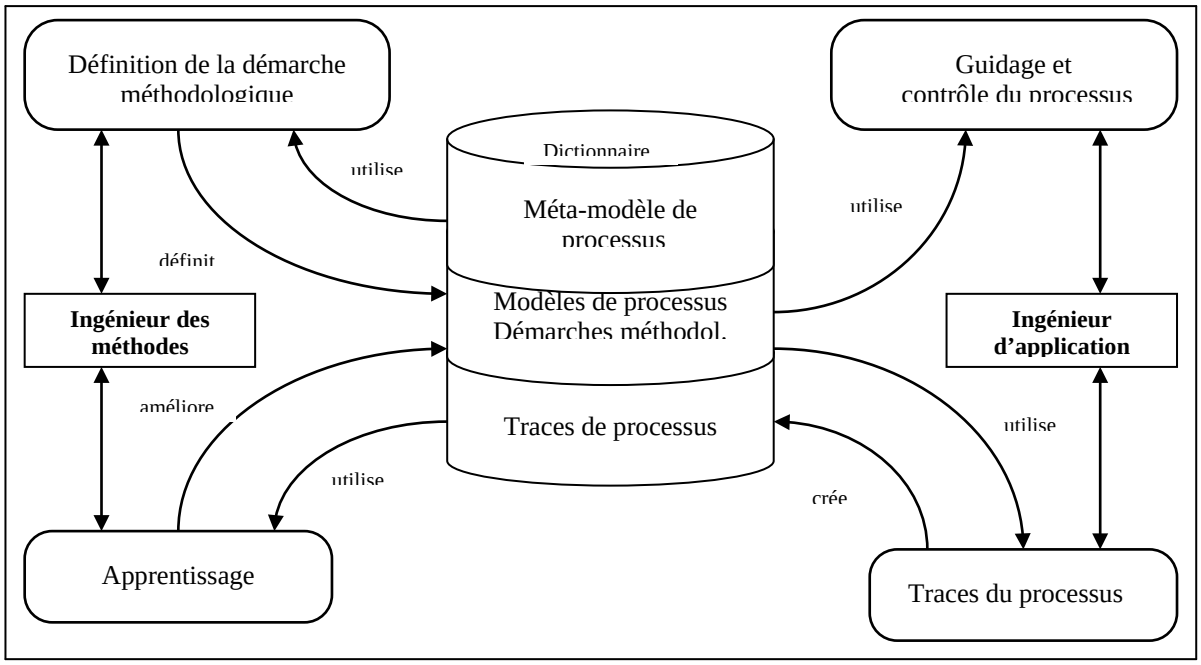


Figure 2.5 : Architecture générale de l'environnement MENTOR

Quand à l'architecture, elle comporte trois composants :

- **Un environnement pour l'ingénieur d'application** :

dans lequel le processus est guidé, exécuté et tracé.

- **Un environnement pour l'ingénieur des méthodes** :

dans lequel le processus est défini et enrichi.

- **Un dictionnaire** :

qui relie les deux environnements précédents et dont le contenu est organisé en trois niveaux : traces de processus, modèles de processus et méta-modèle de processus ; chaque niveau contient de l'information sur le processus et sur le produit qu'il conduit à construire.

Chapitre III

L'INGENIERIE DES METHODES

Vu le nombre de méthodes de développement, leur évolution continuelle, l'élargissement de leur champ d'action et leur inconvenance pour tous les types de projets, il est important de connaître la capacité d'une méthode à offrir des produits de qualité et sa considération des diverses situations caractérisant le développement d'un projet pour la choisir.

Certaines comparaisons de méthodes ont été faites dans l'objectif de disposer d'éléments d'appréciation sur les méthodes tels que les concepts, l'objectif, la portée… D'autres ont considéré l'évolution des méthodes d'une part, vers une plus grande considération de la traçabilité et d'autre part, vers une plus grande flexibilité qui correspond à l'adaptation de la démarche aux diverses situations, tel est le cas de F.HARMSEN qui présent l'évolution des approches par un spectre de flexibilité montrant que l'évolution est passée de l'utilisation d'une méthode rigide vers l'assemblage modulaire des méthodes[Har94].

Par analogie au terme ingénierie des(systèmes, processus), l'ingénierie des méthodes est l'approche systématique pour le développement, la mise en œuvre, la maintenance et l'abandon des méthodes[Sho91]. Elle propose que les approches méthodologiques doivent être développées de la même manière qu'on développe les systèmes d'information.

De nos jours, un consensus semble se dégager sur le fait que l'élément essentiel de la prochaine

génération des environnements de développement serait une plus grande aptitude à incorporer les méthodes. Cette nouvelle génération doit aller d'une part, dans le sens d'une plus grande adaptabilité à la méthode et d'autre part, elle ne doit pas imposer des contraintes aux développeurs mais s'adapter à leurs exigences en leur offrant la possibilité de passer d'une méthode à une autre ou d'un formalisme à un autre.

La prolifération des méthodologies de développement de systèmes d'information et la multiplicité des concepts sous-jacents à ces méthodologies n'ont fait qu'accroître l'embarras des ingénieurs d'application et font que les chercheurs se trouvent devant trois alternatives :

- Développer une méthode universelle, résultat d'une intégration de plusieurs méthodologies, qui sera utilisée pour tous les projets.
- Développer des méthodologies par type d'application. Cela implique l'application en chaque lieu, de la méthode la plus adaptée au projet. On les qualifie de méthodes situationnelles.
- Développer une ingénierie des méthodes qui permettrait de construire une méthode à partir d'un méta-modèle de processus qui peut instancier plusieurs modèles de processus.

3.1 LA METHODE UNIVERSELLE

La tendance de standardisation a eu lieu suite à la constatation qu'il y a des points de convergence entre les méthodes. Il est donc tout à fait normal de s'interroger sur l'existence de tant de méthodes et la question qui se pose est « *pourquoi ne se dégage-t-il pas un standard* ». S'il existait une méthode qui s'applique de façon universelle à tous les cas d'informatisation, qui prend en charge n'importe quel type de problème, quel que soit le contexte organisationnel, tout le monde l'utilisera. En réalité, le nombre de méthodes existantes et leur diversité sont significatifs de la complexité des processus d'informatisation.

Le besoin d'avoir des méthodes de plus en plus performantes pour confronter les divers types de projets, suite à l'évolution rapide des techniques (telles que : approches orientées objet et architectures client/serveur et multimédia), fait que la standardisation de méthodes trouve rapidement ses limites. Bien que cette standardisation ne présente que des avantages pour les praticiens ; l'ouverture des méthodes est une chose plus acceptée. En revanche, plusieurs chercheurs sont optimistes sur le fait que cette standardisation doit concerner les outils dans l'espoir de rendre cette dernière plus transparente. Les essais de standardisation des méthodes restent rares. Le principal travail de renommée internationale qui mérite d'être cité est UML (Unified Modeling Language) proposé par le groupe OMG (Object Management

Group) qui a pour objet d'unifier les méthodes objets.

3.1.1 PRECURSEURS D'UML

UML est un langage graphique de modélisation objet résultant des tentatives de l'unification des méthodes :

- **OOD** (Object Orienred Design) de G. BOOCH
- **OMT** (Object Modeling Technique) de J. RUMBAUGH
- **OOSE** (Object Oriented Software Engineering) de I. JACKOBSON

Le rapprochement en premier lieu d'OMT et de OOD a permis :

- De garder la puissance d'expression de la sémantique d'OMT à travers les concepts d'association entre objets, et l'expression de leur comportement à travers la notion de diagramme d'états-transition,
- De sélectionner de OOD les concepts de modules (sous-systèmes) et la notion de flots de messages.

Le rapprochement en second lieu avec OOSE a permis d'utiliser « les cas d'utilisation » comme moyen d'expression des besoins des utilisateurs dans la phase d'analyse.

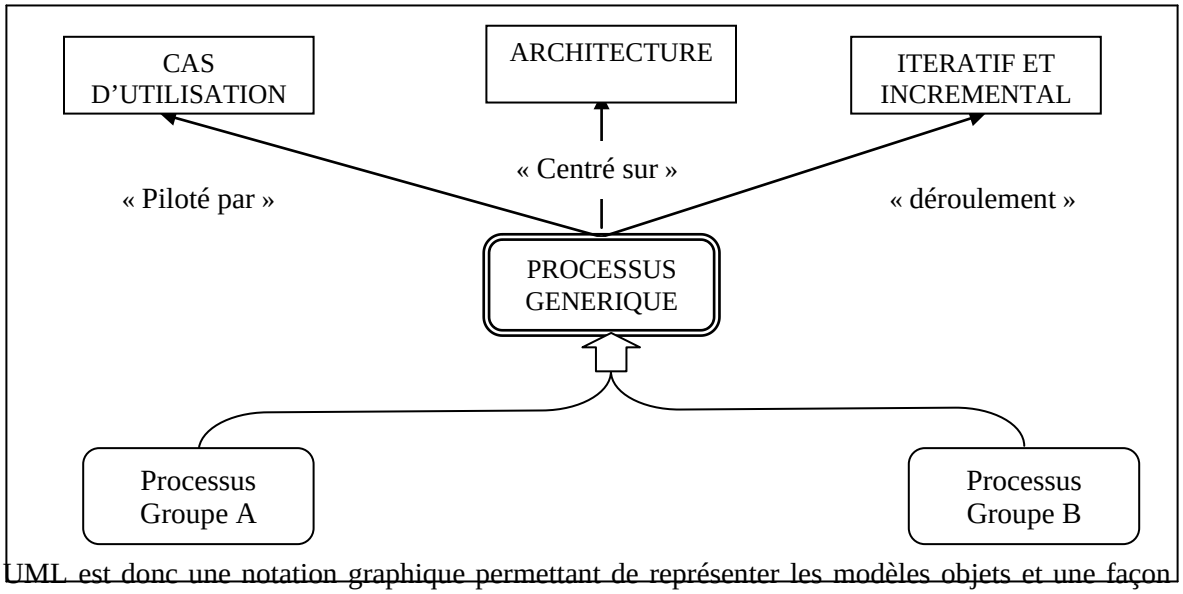


Figure 3.1 : modélisation du processus à l'aide de la syntaxe UML

UML est donc une notation graphique permettant de représenter les modèles objets et une façon standardisée pendant les auteurs donnent quelques consignes à suivre pour le processus de développement :

- Le processus doit être piloté par les cas d'utilisation.
- Le processus doit être centré sur l'architecture.
- Le processus doit se dérouler d'une façon incrémentale et itérative.

3.1.2 CONCEPTS D'UML

UML est constitué de neuf (09) diagrammes :

1. **Les diagrammes d'objets**

Permettent de présenter l'aspect statique des objets et les liens entre ces objets. Les objets sont des instances de classes et les liens sont des instances de relations entre classes.

2. **Les diagrammes de classes**

Décrivent les types d'objets (classes) et leurs relations.

3. **Les diagrammes de cas d'utilisation**

Permettent de mieux enregistrer les besoins des utilisateurs et apportent une aide tout au long du processus de développement. Le modèle de cas d'utilisation comprend les acteurs, le système et le cas d'utilisation lui même.

4. **Les diagrammes de composants**

Décrivent les éléments physiques. Chaque classe est réalisée à l'aide de deux composants : la spécification qui contient l'interface de la classe et le corps qui contient l'implémentation.

5. **Les diagrammes de déploiement**

Ils montrent la disposition physique des différents matériels, ainsi que la répartition des programmes exécutables sur ces matériels.

6. **Les diagrammes d'activité**

Ils expriment le comportement interne des méthodes. Ils sont basés sur le concept d'activité qui représente une étape dans l'exécution de la méthode et celui de la synchronisation qui exprime le déclenchement de la transaction.

7. **Les diagrammes d'états – transition**

Décrivent le comportement exhaustif de chaque objet de manière informelle. Le comportement d'un objet peut être décrit en termes d'états et d'événements ou chaque état est caractérisé par un ensemble de valeurs de l'objet à un instant donné, l'événement est un déclencheur de la transaction d'un état à un autre

8. **Les diagrammes de collaboration**

Ils sont associés aux diagrammes d'objets et représentent les interactions entre objets. Une

interaction est réalisée par un groupe d'objets qui collaborent en échangeant des messages.

9. **Les diagrammes de séquence**

Ils représentent l'interaction entre les objets en insistant sur la chronologie des envois de messages.

3.1.3 LES APPORTS D'UML

Les avantages apportés par UML tiennent du concept objet et du développement à base de composants :

- **Réactivité**

Dans un système objet, la prise en compte d'une évolution fonctionnelle est plus rapide et les éléments concernés sont vite identifiés. La problématique ne réside pas dans le choix du langage mais plutôt dans une modélisation du sujet de l'étude ; Il doit être possible de modifier les modèles et leur ajouter des fonction nouvelles.

- **Plus grande évolution des applications**

Les systèmes développés avec UML bénéficient d'une forte structuration du logiciel, d'une grande modularité et des mécanismes d'abstraction et d'encapsulation qui fournissent les bases idéales pour la réutilisation.

- **Gains de productivité en développement et en maintenance**

Une fois la problématique de l'objet et d'UML comprise, la productivité des équipes s'améliore au fur et à mesure que les investissements se mettent en place en vue de la réutilisation en recherchant ce qui est disponible avant de réinventer.

- **Meilleure communication avec les utilisateurs**

L'approche par les cas d'utilisation remet l'utilisateur au centre de l'analyse et de la conception. Le prototypage est le moyen de faire le pont entre la difficulté de l'utilisateur à spécifier ses besoins et celle de l'analyste à les comprendre.

- **Interfaces Homme-Machine plus sophistiquées**

Il existe de nombreuses bibliothèques de composants de type IHM disponibles sur le marché et de nombreux logiciels de construction des IHM utilisant cette approche.

3.2 LES METHODES SITUATIONNELLES

L'analyse de la pratique des méthodes montre que celles-ci ne sont jamais suivies à la lettre mais au contraire, adaptées à la situation de chaque projet. Toutes sortes de facteurs relatifs au projet, à la technologie, à l'expertise de l'équipe et aux caractéristiques du domaine d'application induisent une adaptation de la méthode au profil du projet.

Une méthode situationnelle est construite en fonction de la situation particulière d'un projet par agencement de fragments de méthodes prédéfinis, validés et accessibles dans une base de méthodes [Rol 96]. Cette attitude préserve les acquis et apporte la flexibilité en fournissant les moyens d'adapter une méthode aux besoins spécifiques d'un projet.

LES TYPES DE FRAGMENTS 3.2.1

Comme chaque méthode est définie par ses produits et son processus, les fragments de méthodes se divisent en deux catégories[Fig 3.2] :

Les fragments de produits : - qui représentent les produits et les sous-produits dérivables par une méthode.

- **Les fragments de processus :**

qui représentent les étapes ou les activités à effectuer pour réaliser un fragment de produit.

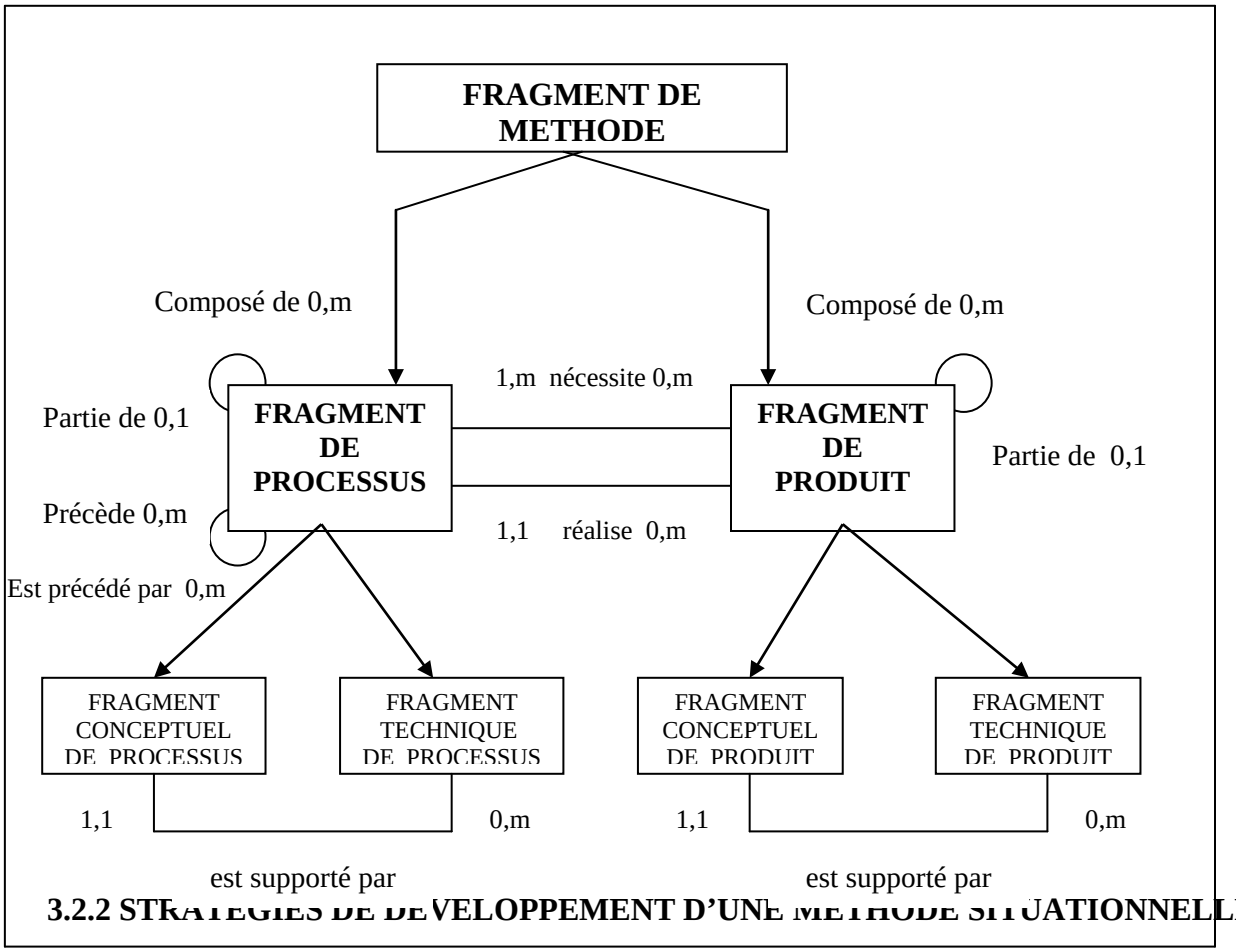
Dans une méthode situationnelle, les fragments de méthodes peuvent être considérés de niveau conceptuel, comme il peut s'agir de simples descriptions d'outils ou de parties d'outils ; les premiers sont appelés des fragments conceptuels, les seconds des fragments techniques[Fig 3.2].

- **Les fragments conceptuels :**

représentent les méthodes et les parties de méthodes tels que les descriptions de modèles et activités ; les fragments techniques contiennent les composants des outils CASE tels que les éditeurs de diagrammes.

- **Les fragments conceptuels :**

de produits doivent être supportés par des fragments techniques de produits et les fragments conceptuels de processus doivent être supportés par des fragments techniques de processus.



3.2.2 STRATEGIES DE DEVELOPPEMENT D'UNE METHODE SITUATIONNELLE

Figure 3.2 : les fragments de méthodes

une certaine flexibilité autour d'une même méthode, la deuxième associe des fragments de différentes méthodes. F.HARMSEN propose un spectre de six stratégies de développement de méthode situationnelle correspondant à un degré croissant de flexibilité passant de l'utilisation d'une méthode d'une manière rigide vers l'assemblage modulaire de fragments de méthodes[Har94].

1. **Utilisation d'une méthode rigide**

C'est l'usage traditionnel d'une méthode qui ne prévoit pas de mécanismes de son adaptation à des situations particulières. La même méthode est utilisée pour tous les projets d'une organisation.

2. **Sélection d'une méthode rigide**

Il s'agit de sélectionner, parmi plusieurs méthodes, une seule pour un projet spécifique. Cela permet une meilleure adéquation de la méthode au projet.

3. **Sélection d'un chemin dans une méthode**

Il s'agit de sélectionner un chemin méthodologique parmi les multiples que prévoit une méthode.

4. **Sélection et adaptation d'une méthode**

C'est une extension de la précédente. La méthode est enrichie par de nouvelles techniques et pour chaque projet on sélectionne le chemin de développement à suivre.

5. **Approches multi-vues**

Propose une combinaison prédéfinie de méthodes, chacune adaptée à l'un des aspects du développement du système.

6. **Construction modulaire**

Il s'agit de configurer une méthode à partir de fragments provenant de plusieurs méthodes et qui sont sélectionnés en fonction des valeurs des facteurs de contingence caractérisant le projet.

3.3 LA META-MODELISATION

Etant donné que l'activité de modélisation devient de plus en plus complexe et vu que la considération d'une situation donnée pour développer un projet s'impose, le nombre de modèles et de types de modèles ne cessent d'augmenter et les liens entre eux se sont multipliés. Ceci nécessite donc la gestion de modèles multiples et fait apparaître la nécessité de relier les spécifications décrites moyennant divers formalismes à travers une correspondance entre ces divers formalismes. Ceci est rendu possible grâce à la méta-modélisation qui se présente comme une technique puissante pour la description des différents niveaux d'abstraction et les liens de correspondance entre eux.

Par analogie à la modélisation, qui est considérée comme étant la classification des instances par un modèle, la méta-modélisation peut être définie d'une manière récursive, comme étant la modélisation des formalismes d'une méthode par un méta-modèle afin d'élaborer une classe de formalismes. Un méta-modèle sert de référence à la représentation des différents modèles spécifiques et facilite leur évaluation et leur comparaison. Elle est appliquée aussi bien aux produits qu'au processus d'une méthode, D'où la reconnaissance de divers niveaux d'abstraction de produit et de processus et la relation entre les niveaux respectifs.

La modélisation utilise des modèles pour percevoir des systèmes, la méta-modélisation utilise des méthodes pour percevoir une modélisation. le méta-modèle définit les informations relatives aux concepts manipulés par la méthode, aux formes de représentation de ces concepts, à la manière de les relier pour former le produit et enfin à la manière d'utiliser la méthode (le processus)[Fig 3.3].

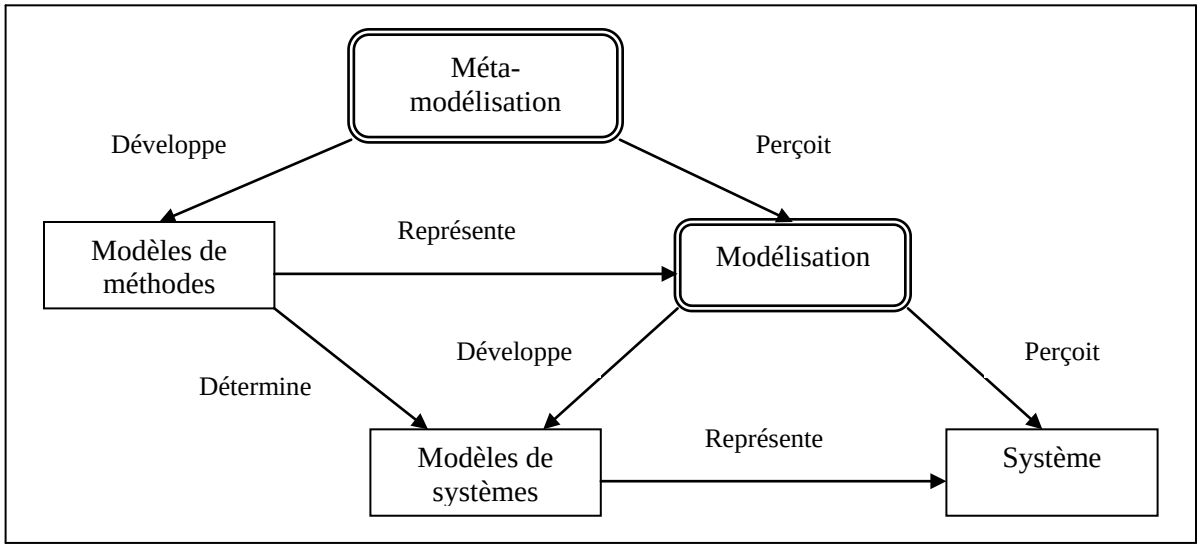


Figure 3.3 : Méta-modélisation et Modélisation

3.3.1.1 LE PRODUIT

Il est le résultat d'une étape quelconque du processus. Il est généralement représenté par des modèles graphiques structurés qui peuvent être vérifiés, documentés et convertibles. Il est caractérisé par quatre niveaux d'abstraction :

- **l'application**

  Elle représente le niveau le plus bas, c'est une instance d'un modèle d'une méthode appliquée à un projet donné.

  Exemple : diagrammes d'instances de classes et diagrammes de scénarios (dans la méthode de OMT).

- **Le schéma**

  Il représente un modèle d'une méthode appliquée à un projet donné, c'est le schéma de l'application.

  Exemple : diagrammes de classes et types de scénarios.

- **Le modèle du produit**

Définit les concepts pour modéliser une application, il est constitué des différents modèles d'une méthode.

Exemple : modèle statique, modèle dynamique et modèle fonctionnel.

- **Le méta-modèle du produit**

Il définit les formalismes qu'on peut utiliser pour modéliser une application et permet d'instancier les modèles graphiques d'une méthode.

Exemple : représentation de modèles d'OMT à l'aide du concept de classes et relation entre classes.

3.3.1.2 LE PROCESSUS

Le processus est l'ensemble intégré d'étapes ou d'activités requises pour transformer les différents produits de développement de l'état initial à l'état final.

Il est caractérisé par trois niveaux d'abstraction :

- **L'instance**

Consiste en un ensemble d'étapes parallèlement ordonnées pour répondre à un besoin spécifique.

Exemple : Action de créer l'objet classe « CLIENT » pour une application particulière de gestion de commandes.

- **Le modèle du processus**

Il représente l'activité de modélisation d'un ensemble d'actions de même type, il décrit un type de processus.

Exemple : Action de « CREER LES OBJETS CLASSES » afin de les instancier.

- **Le méta-modèle du processus**

Il permet de représenter les concepts génériques utilisés pour décrire les modèles de processus.

Exemple : Action génériques «IDENTIFIER ACTIONS » et « ORDONNER ACTIONS ».

3.3.1.3 RELATIONS ENTRE LES ABSTRACTIONS DE PRODUIT ET DE PROCESSUS

- Le processus conserve la trace de la manière dont le produit a été construit.

Exemple : si le produit contient les classes « CLIENT » et « COMMANDE » , le processus contient les éléments décrivant la création de ces classes.

- Le modèle de processus fera référence aux concepts du modèle de produit.

Exemple : Le modèle de processus « CREER LES CLASSES OBJETS » fait référence au modèle de produit d'OMT « MODELE STATIQUE ».

- Les méta-types du méta-modèle de processus font référence aux méta-types du méta-modèle de produit.

Exemple : le méta-type « IDENTIFIER ACTIONS » du méta-modèle de processus fait référence au méta-type « CONCEPTS » du méta-modèle de produit.

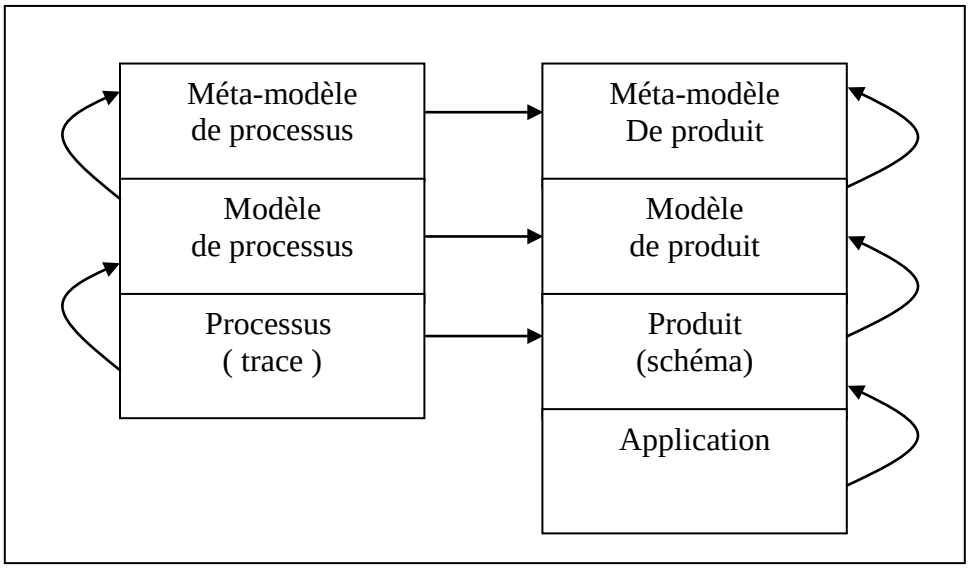


Figure 3.4 : Relations entre niveaux d'abstraction des produits et ceux des processus

3.3.2 LES META-FORMALISMES

Le problème qui se pose lorsqu'on aborde la méta-modélisation est celui de choisir le méta-formalisme adéquat qui permet de représenter le méta-modèle. Le choix d'un méta-formalisme repose sur la réponse à deux questions :

- La première «quels *types de formalismes peut-on représenter* ? »

- La deuxième «quels *sont les concepts de base que doit offrir ce méta-formalisme* ? » [Ald91].

La première question nous ramène à considérer deux types de méta-formalismes, l'un pour les produits l'autre pour les processus.
La deuxième est liée à la qualité du méta-formalisme.

3.3.2.1 QUALITES D'UN META-FORMALISME

Les qualités attendues d'un méta-formalisme sont déterminantes pour le choix de ce dernier, les plus couramment citées sont :

- **Généralité**

la qualité essentielle d'un méta-formalisme est d'être général pour constituer un modèle de référence permettant de définir le plus grand nombre de modèles potentiels.

- **Complétude**

le modèle de référence que constitue le méta-formalisme doit permettre de décrire complètement toutes les caractéristiques inclues dans le modèles.

- **Précision et unicité**

le méta-formalisme doit permettre de donner une interprétation précise et unique à chaque présentation.

- **Concision et puissance**

le méta-formalisme doit être suffisamment concis tout en gardant sa puissance de représentation ou de description.

3.3.2.2 DIFFERENTES CATEGORIES DE META-FORMALISMES

- **Les méta-formalismes génériques**

ils incluent une large gamme de concepts à combiner pour constituer méta-modèle d'une méthode, leur combinaison repose sur l'agrégation et/ou la spécialisation [Pot89]. L'avantage de cette catégorie

réside dans le choix d'un ensemble de concepts à utiliser, tandis que l'inconvénient de cette dernière est la difficulté d'intégration des concepts du méta- modèle car la relation entre ces concepts est syntaxique.

- **Les méta-formalismes hiérarchiques**

Ils reposent sur l'utilisation de liens de type occurrence ou spécialisation « is-a ». Leur avantage est la facilité de description des méta-modèles, cependant il peuvent générer des hiérarchies longues.

- **Les méta-formalismes entité-association**

Ils proposent l'entité type (d'entité) comme concept de base de la méta-modélisation. Les Entités sont reliées entre elles par des associations (type d'association) caractérisées par des propriétés(ou attributs). Pour accroître leur pouvoir d'expression, ces méta-formalismes sont étendus et enrichis par d'autres concepts complémentaires tels que l'hiérarchie et la partition [Sah95].

Chapitre IV

LE META-MODELE NATURE

Le méta-modèle nature (Novel Aproach Toward Underlying Requriement Engineering) décrit le processus de développement d'un système d'information comme des portions de démarches permettant d'assister le concepteur au cours de la progression du processus de développement. Dans NATURE le déroulement du processus est vu comme étant une succession destinée à construire un schéma conceptuel de système d'information à partir de spécifications initiales, d'intentions du concepteur sur des situations et appliquées pour la réalisation d'actions de transformation de spécifications[Ali99b].

Au début du processus, la situation initiale est réduite à une collection de besoins exprimés de manière formelle. Au fur et à mesure de l'évolution du cycle de développement, les situations deviennent des éléments de produit.

4.1 LE MODELE DE DEMARCHE NATURE

Le méta-modèle NATURE repose sur un paradigme contextuel définit par les concepts suivants :

- **La situation :** est toute partie sur laquelle est définie une intention.
- **La décision :** est la précision d'un objectif du concepteur destinée à faire progresser l'analyse vers l'obtention du produit final.
- **L'action :** est une transformation du produit.
- **Le contexte :** est l'association d'une situation et d'une intention que le concepteur a sur cette situation.

Pour assurer le lien entre le processus et le produit, il s'agit d'expliciter d'une part, comment la situation est construite sur le produit et d'autre part, comment une action modifie le produit. La situation doit spécifier un ensemble minimum de parties du produit-type auquel peut être associée au moins une décision de démarche, ainsi ces deux concepts établissent une relation de référence entre le processus et les produits à élaborer.

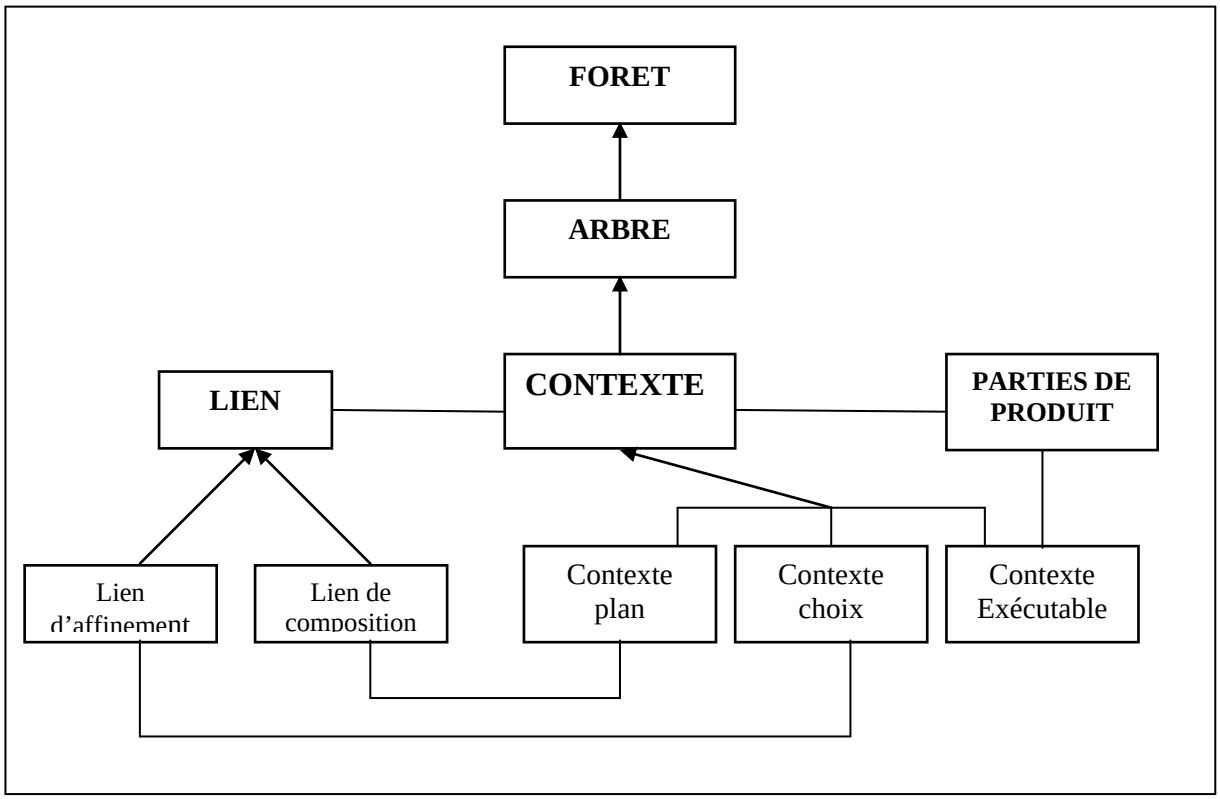


Figure 4.1: le modèle de démarche NATURE

La description du processus devient une arborescence de contextes complexes (hiérarchie de composition pour les contextes plan et les différentes alternatives pour les contextes choix)et de contextes simples (contextes exécutables). Les différents nœuds de l'arborescence sont des contextes complexes, les feuilles sont des contextes exécutables[Fig 4.1].

L'arbre est un ensemble de contextes associés par les liens d'affinement et de composition.

- **Un lien de composition :** matérialise la relation entre un contexte plan et ses composants.
- **Un lien d'affinement :** représente l'association existant entre un contexte choix et ses alternatives.

Plusieurs arbres, sont nécessaires pour décrire le processus d'une méthode, participant à la définition d'un modèle de processus, cette collection est appelée forêt[Fig 4.1].

4.2 LE META-MODELE DE PRODUIT

Le concept central est celui de partie de produit qui se spécialise en produit, élément de produit ou association de parties de produits.

- **Les parties de produits :** désignent les fragments de produits de complexité variable.
- **un produit :** est le résultat de l'analyse qui peut se décomposer en éléments de produits.
- **Les associations de parties de produits :** sont des liens entre produits distincts.
- **Les éléments de produits :** sont des composants de produits structurés en hiérarchie et classifiés en deux catégories suivant leur complexité et leur type: élément de produit atomique et élément de produit composé[Fig4.2].

Deux types d'élément de produit sont définis :

- **élément de produit non lien :** représente les éléments de produits qui ont une existence propre
- **lien inter-éléments de produit :** traduit les relations qui peuvent exister entre deux éléments de produits.

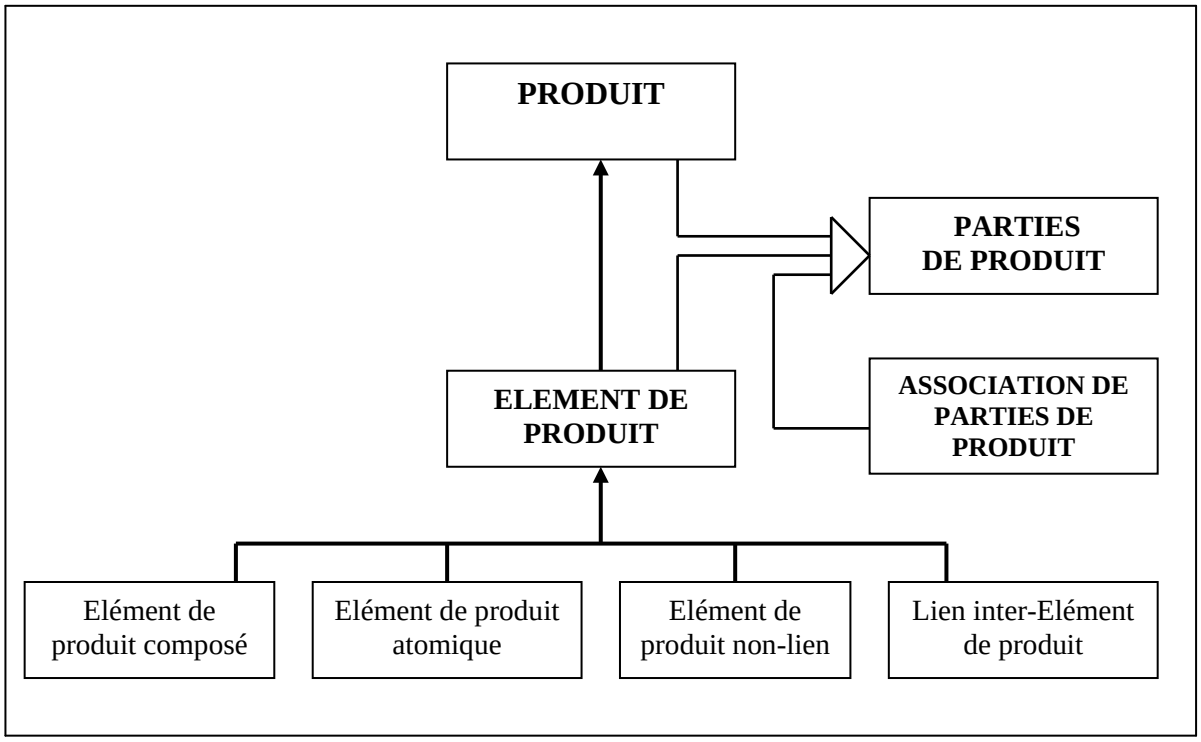


Figure 4.2 : le méta-modèle de produit NATURE

4.3 LE META-MODELE DE PROCESSUS

Le méta-modèle de processus nature s'appuie sur cinq concept :

- **La situation** définie sur des parties de produits expliquant le contexte dans lequel existe la décision.
- **La décision** qui guide le processus.
- **Le couplage** situation-décision qui détermine le contexte.
- **L'action** qui met en œuvre les transformations du produit.
- **Les parties de produits** modifiées par les actions et sur lesquelles sont constatées les situations.

La mise en avant du contexte est très significative car durant le processus de développement, le concepteur agit et prend des décisions de façon contextuelle. Le contexte permet de représenter, à un niveau de granularité choisi, un pas particulier du processus. Il est affiné en trois types où chacun intervient différemment dans le déroulement du processus.

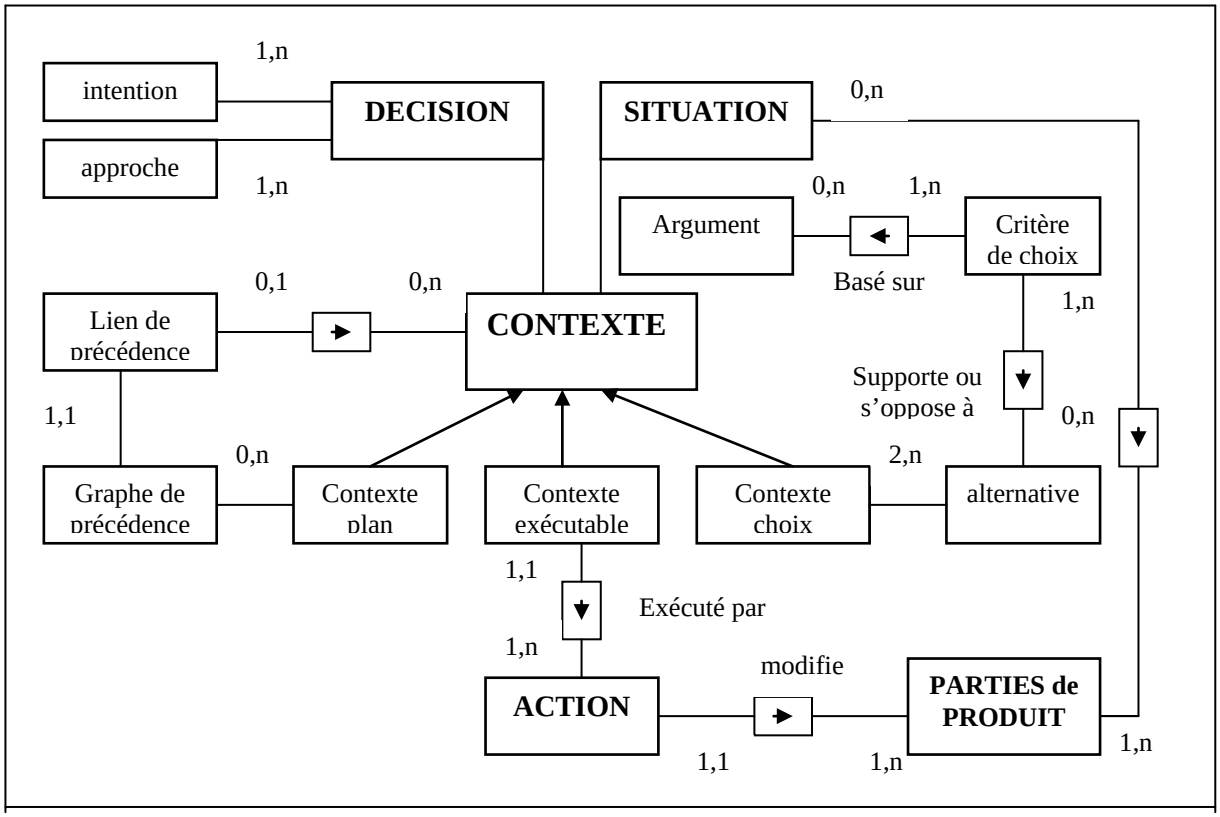


Figure 4.3 : le méta-modèle de processus NATURE

4.3.1 LES CONTEXTES PLAN

Le contexte plan permet de décomposer un problème complexe en un ensemble de sous-problèmes, a chaque problème correspond une sous-décision prise sur une sous-situation de la situation du problème initial, un contexte dont l'exécution n'est spécifiée que par des relations de décomposition vers d'autres contextes (liens de composition).

Les contextes composants d'un contexte plan peuvent être exécutés d'une manière séquentielle, itérative et/ou parallèle, ceci est présenté par un graphe de précédence où les nœuds représentent les sous-contextes et les arcs représentent les liens de précédence qui définissent les transitions ordonnées des contextes ou leur exécution parallèle.

La figure suivante nous montre un exemple simplifié de contexte plan et ses contextes composants relatifs à l'élaboration d'un schéma Entité-Association selon une approche descendante, ainsi que le graphe de précédence qui y est associé.

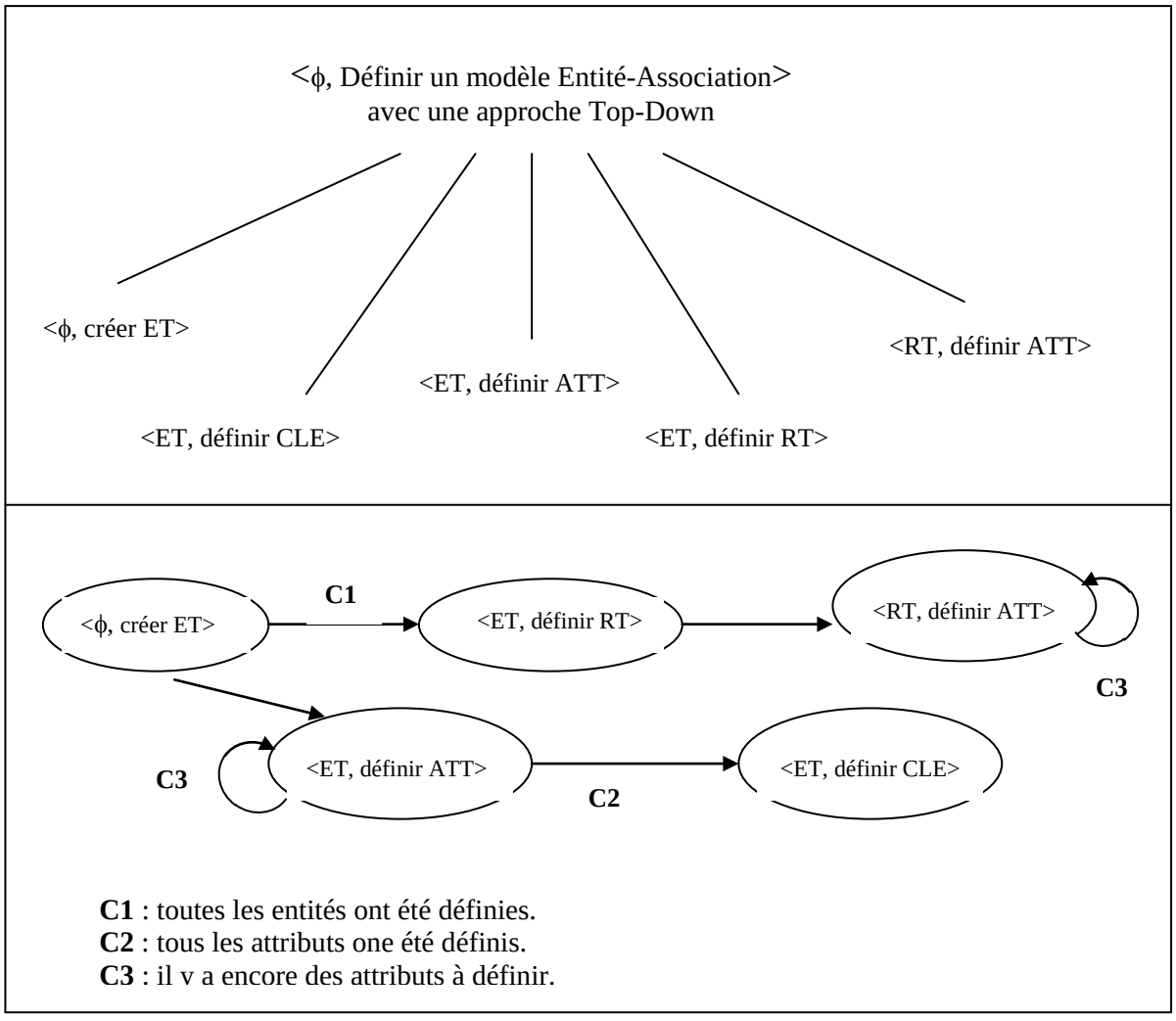


Figure 4.4 : Exemple d'un contexte plan

4.3.2 LES CONTEXTES CHOIX

Un contexte choix correspond à une situation qui nécessite l'exploration de différentes alternatives lors de la prise de décision et choisir, selon certains critères, la plus appropriée. Chaque alternative est caractérisée par un critère de choix composé d'une combinaison d'arguments en faveur ou s'opposant à la décision dans une situation donnée.

L'alternative est elle-même un contexte qui peut être contexte exécutable, plan ou choix. Cette hiérarchisation de contextes peut être interprétée comme un moyen de raffinement des décisions de haut niveau en décisions plus fines.

La figure suivante montre un exemple de contexte choix qui consiste à élaborer un modèle Entité-Association selon trois alternatives :

- soit une approche descendante,
- soit une approche ascendente,
- soit une approche mixte.

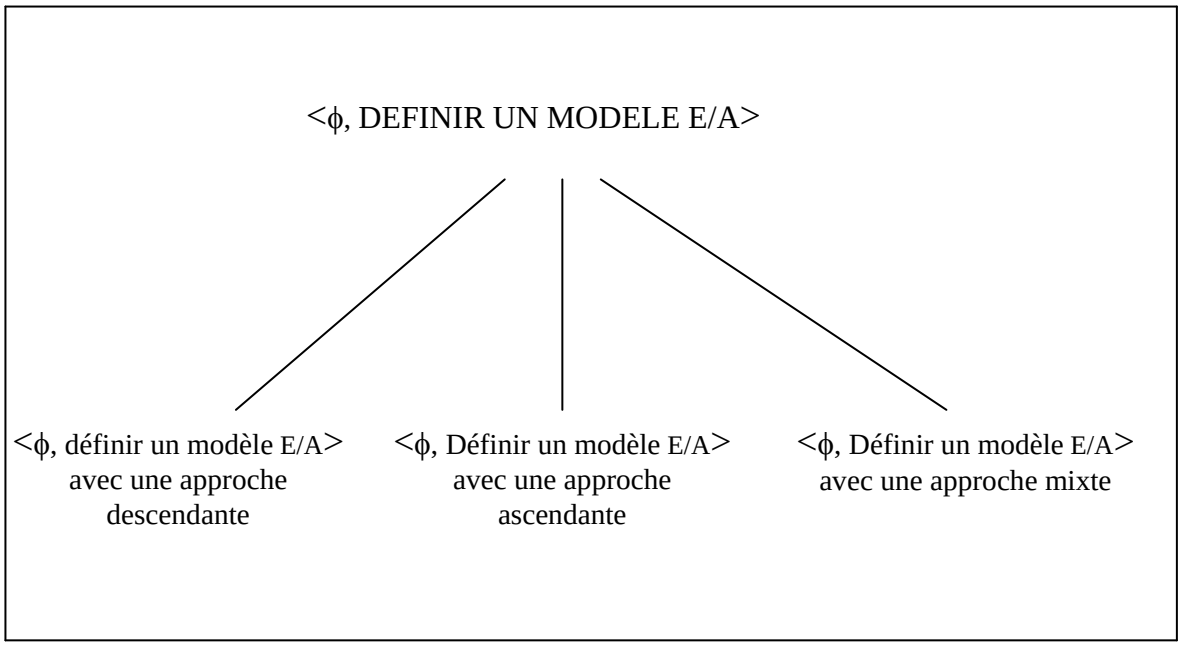


Figure 4.5 : exemple d'un contexte choix

4.3.3 LES CONTEXTES EXECUTABLES

A son niveau de granularité le plus bas, le processus de conception est une suite d'actions élémentaires qui transforment le produit d'une forme vers une autre en créant de nouveaux éléments de produit et/ou transformant ceux existants. Il est associé à une action qui porte sur le produit en cours de développement ; celle-ci peut être atomique ou composée selon qu'elle modifie une ou plusieurs parties du produit.

Un deuxième groupe d'actions peut être envisagé. Ce sont les actions de sélection de contextes, celles-ci fournissent le guidage de flux (flow guidance) .Elles aident dans la sélection de l'intention suivante qui doit être satisfaite (prochain contexte à exécuter) afin de faire avancer le processus [Gna99] .

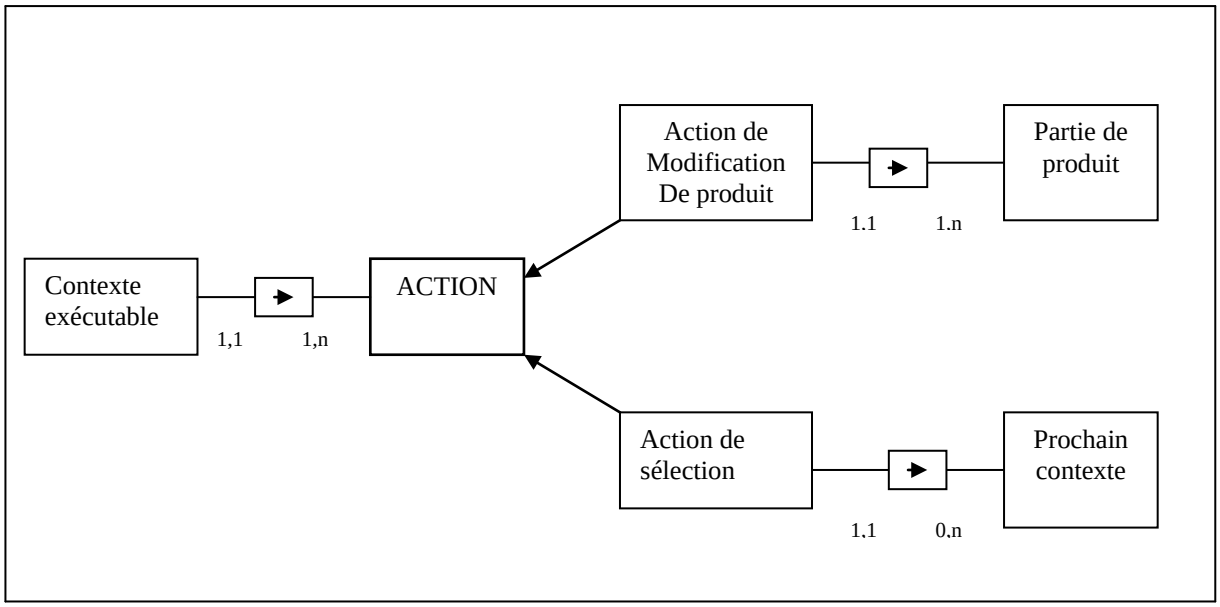


Figure 4.6 : structure détaillée d'un contexte exécutable

L'exécution d'une action change le produit et peut générer de nouvelles situations donnant lieu à la prise d'autres décisions. **<ϕ, créer ET>** est un contexte exécutable générant l'action de créer une entité dans un schéma E/A selon une approche descendante.

Chapitre V

PRESENTATION DE LA DEMARCHE

Du fait que la durée de réalisation des projets est longue et l'évaluation continuelle des techniques parait comme étant une caractéristique indissociable aux approches de développement de systèmes informatiques, le souci de considérer les nouvelles situations et donc la réactivité au changement au cours du développement de systèmes parait aussi nécessaire. C'est pourquoi les approches de développement ont évolué vers des méthodes situationnelles , cette évolution est marquée par une augmentation de leur degré de flexibilité pour la considération des différentes situations [Har94]

D'un autre côté , étant donné que l'activité de modélisation devient de plus complexe et la considération d'une situation donnée pour développer un projet s'impose , le nombre de modèles et de types de modèles ne cessent d'augmenter et les liens entre eux se multiplier, leur gestion est rendue possible grâce à la méta-modélisation qui est une technique puissante pour la description des différents niveaux d'abstraction et des liens de correspondance entre eux .

Cependant la possibilité de passage d'un modèle à un autre ou l'utilisation d'une approche situationnelle nécessite le recours à l'activité de transformation et la navigation entre les méthodes [Jam98].

En ce qui concerne la transformation on distingue deux cas :

- La transformation dans un même modèle (niveau d'abstraction).
- La transformation d'un modèle vers un autre (conversion).

En ce qui concerne la navigation entre les méthodes, il faudrait d'une part formaliser les transformations locales et d'autre part définir un processus de transformation global défini par un ensemble d'étapes décrites à l'aide de règles de transformation.

Quant aux environnements de développement, ils sont considérés comme des AGLs intégrant une collection d'outils supportant toutes les activités de développement pour la couverture de tout le cycle de vie. Plusieurs travaux ont été menés ; certains concernent la prise en compte de la méta-modélisation, d'autres se sont intéressés à l'aspect multi-méthodes, les plus récents sont plus étendus et proposent une stratégie de développement pour chaque type de projet[Hhg97].

5.1 LA CONSTRUCTION

La démarche proposée est inspirée de [Har 94] , elle repose principalement sur quatre Phases successives et indispensables[Fig 5.1].

La première phase est la caractérisation du projet dont le point de départ est l'environnement qui peut imposer une cohérence de l'ensemble des produits de conception dans des contextes différents. Elle implique également l'existant, les utilisateurs, la culture et l'expérience méthodologique des concepteurs et les outils disponibles ou utilisés. Elle est réalisée par l'évaluation d'un certain nombre de facteurs décisifs pour le projet.

Cette caractérisation sera prise en compte lors de la phase de sélection des fragments de méthodes à partir d'une base de méthodes conformément à un méta-modèle.

Les fragments sont insérés, actualisés, modifiés et supprimés de la base par une fonction d'administration de méthodes et ce après avoir été sélectionnés et représentés.

La phase d'assemblage consiste au montage des différents fragments sélectionnés en effectuant un certain nombre d'opérations de transformation selon des règles précises, ensuite les assembler en les triant, vérifiant et ordonnant en une seule méthode.

La méthode résultante est appliquée à un projet qui peut nécessiter des révisions dues soit à un changement dans l'environnement, soit à la clarification de certaines caractéristiques du projet et par voie de conséquence la méthode doit être améliorée au cours du développement, adaptée et validée pour l'application étudiée.

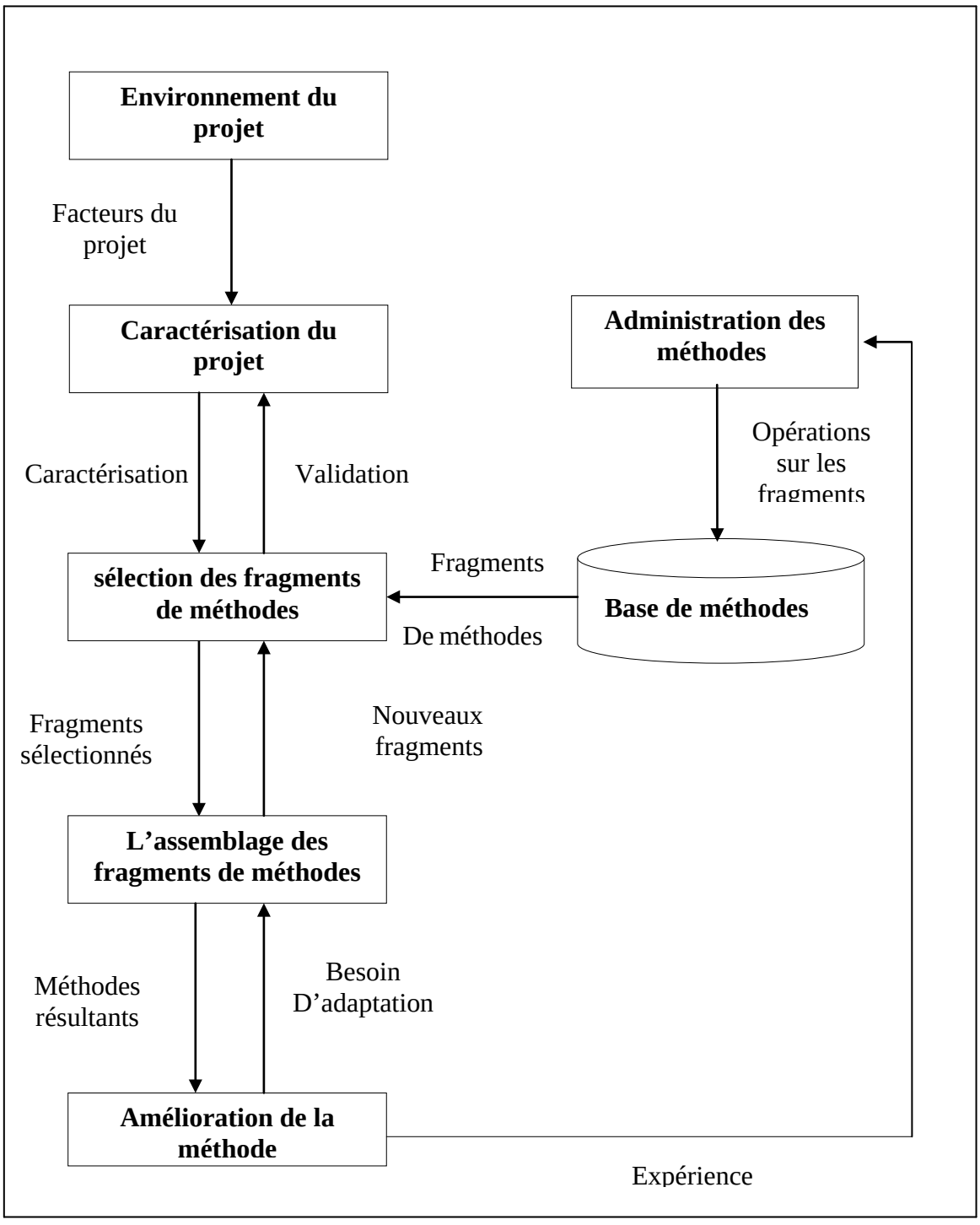


Figure 5.1 : Processus de construction de la démarche.

5.2 LES FACTEURS DE CARACTERISATION DU PROJET

Un projet est décomposé en une série d'étapes selon une démarche qui part du besoin pour aboutir au produit fini qu'est le système informatique. Il induit des aspects organisationnels, humains, techniques, spacio-temporels et de qualité qui agissent et interagissent tout au long de son déroulement.

Quel que soit le type de modèle utilisé, la phase de développement d'un projet s'inscrit dans un milieu caractérisé par des facteurs qui auront une incidence sur la stratégie de conception à mettre en place [Pad98]. En fonction des valeurs de ces facteurs, le projet est considéré à risque ou non, pertinent ou non.

L'ENVIRONNEMENT -

L'environnement peut être mal structuré dans les domaines n'ayant fait l'objet d'aucune Informatisation, le projet passe alors par un schéma directeur qui impose une démarche structurée s'appuyant sur une méthodologie permettant d'extraire les données essentielles du futur S.I et les règles de gestion importantes en les organisant par domaines informationnels.

IMPORTANCE DU S.I -

Elle influe sur le contenu de la démarche mise en œuvre pour la conception et se décline en fonction de deux facteurs :

- le volume des objets mis en jeu
- l'impact du projet concerné sur les résultats de l'entreprise.

Les moyens mis en œuvre, les produits résultants demandés, l'approche qualité sont, par exemple, des éléments variables dans les différents projets au sein d'une même entreprise. Dans un petit projet, il est permis que certains éléments de l'activité de conception soient supprimés ou que certains produits résultants soient simplifiés.

EXPERIENCE DES CONCEPTEURS -

Cette expérience est liée à la technicité mise en œuvre pour la conception, mais également à la connaissance du concepteur de l'entreprise et en particulier du S.I existant. Celle ci peut l'amener à effectuer des choix implicites ou à éviter des voies culturellement incompatibles avec l'entreprise. Le métier du concepteur intègre également une composante moins technique liée à la gestion de la démarche qu'il suit et des ressources qu'il gère.

COUTS ET DELAIS -

Les délais sont souvent un facteur prédominant, leur influence s'accroît avec le taux d'avancement du projet et donc avec les coûts réellement engagés. Ce type de facteur contribue à une simplification de la démarche de conception qui peut s'avérer fatale pour le projet.

LE TYPE DE PROJET -

Les systèmes transactionnels, statistiques, d'aide à la décision, de production, etc… mettent en jeu des techniques différentes qui nécessitent des moyens et des ressources variées.

LES OUTILS -

Les outils disponibles sur le marché sont de nivaux de sophistication variables, ils tentent de faciliter les représentations graphiques ou textuelles et automatisent certaines tâches. Cependant aucun n'apporte une solution universelle.
Les outils sont liés à un ou plusieurs cycles de développement, à une ou plusieurs méthodes.
C'est au concepteur de définir et d'y introduire des spécificités contextuelles généralement liées aux normes et standards retenus pour l'entreprise.

Les justifications des choix et des tendances restent encore des notions purement intellectuelles et les outils n'apportent pas d'aide véritable dans la définition de la stratégie informatique de l'entreprise.

L'EXISTANT -

Les outils aidant à la construction du S.I deviennent une contrainte dès lors qu'ils sont déjà utilisés. Les choix d'outils faits à un instant donné deviendront naturellement une contrainte de l'existant dans le futur. La stratégie de la conception doit donc intégrer des contraintes à long terme et il est souvent plus pertinent de retarder le choix d'un outil, plutôt que d'en imposer un qui aura une durée de vie limitée.

L'existant se mesure également en terme de taux d'activités déjà automatisées. En fonction de son importance, le processus de conception le prend en compte ou non de manière à aboutir soit à une évolution, soit à une refonte.

5.3 LA SELECTION DES FRAGMENTS DE METHODES

Un fragment de méthode décrit les directives relatives à l'usage des modèles de celle-ci, il suggère comment progresser d'un point à un autre du processus pour satisfaire un objectif partiel de conception que l'Ingénieur se fixe. Il pourrait être également considéré comme un module structuré de connaissances pour aider à la prise de décision dans le processus [Gna99]. Il est important de formaliser les méthodes de manière à pouvoir identifier les fragments de méthodes et caractériser un fragments pour faciliter son usage dans un contexte situationnel.

On en distingue deux types : les fragments de produits et les fragments de processus.

5.3.1 LES FRAGMENTS DE PRODUITS

les fragments de produits sont tous les produits et sous produits dérivables d'une méthode tels que documents, modèles et diagrammes. Leurs descriptions sont souvent données par des méta-modèles de données représentant ces derniers par des diagrammes Entité/Relation [Fig 5.2].

les fragments de produits mettent en jeu trois concepts :

- **l'entité** : qui permet d'identifier le fragment par rapport à d'autres.
- **l'attribut** : qui contient la valeur d'une propriété de l'entité.
- **la relation** : qui permet de lier deux entités différentes.

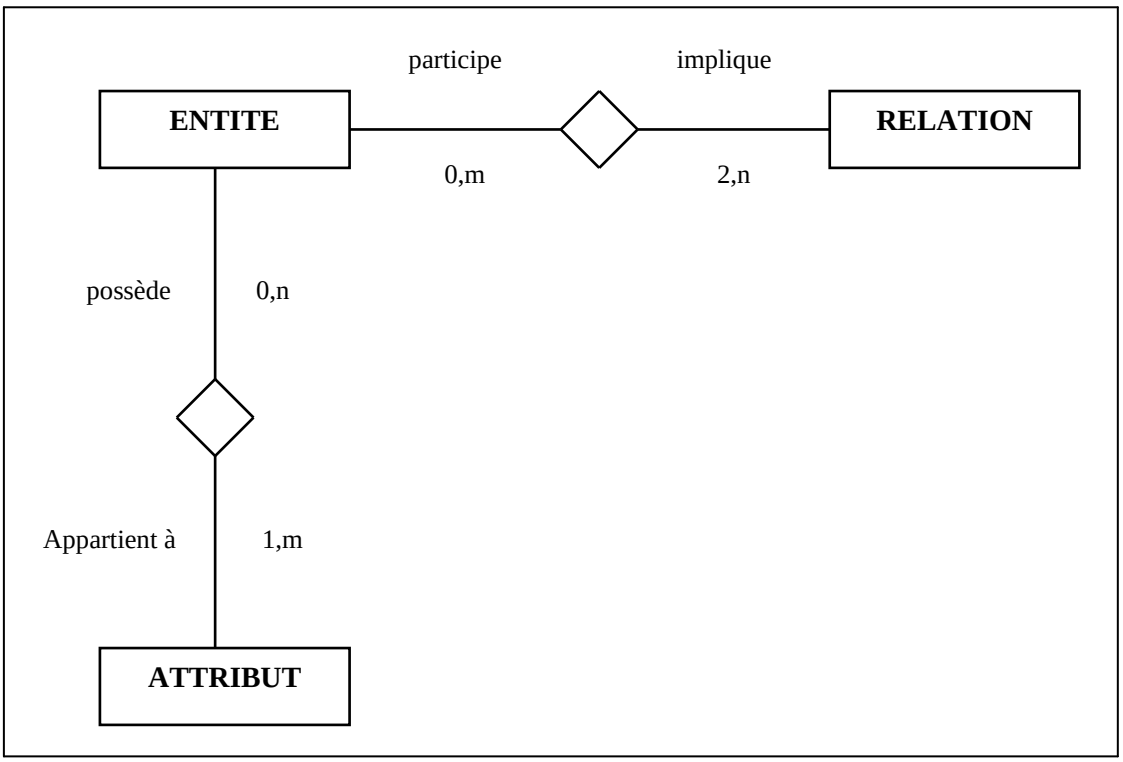


Figure 5.2 : Description d'un fragment de méthode

Une entité peut être impliquée dans plusieurs relations et peut contenir plusieurs attributs en utilisant le modèle de représentation suivant :

Entité : (réf.E , dés.E , typeE) -

Attribut : (réf.A , dés.A) -

Relation : (réf.R , dés.R , inv.R) -

Implique : (réf.E , réf.R , card.R) -

Appartient A : (réf.A , réf.E) -

5.3.2 LES FRAGMENTS DE PROCESSUS

les fragments de processus représentent toutes les étapes, les activités et les tâches qu'il faut effectuer pour réaliser un fragment de produit. Leurs descriptions sont également données par des méta-modèles de processus [Fig 5.3] où les rectangles représentent les tâches, les cercles représentent les décisions et les flèches représentent les déclencheurs[Har94].

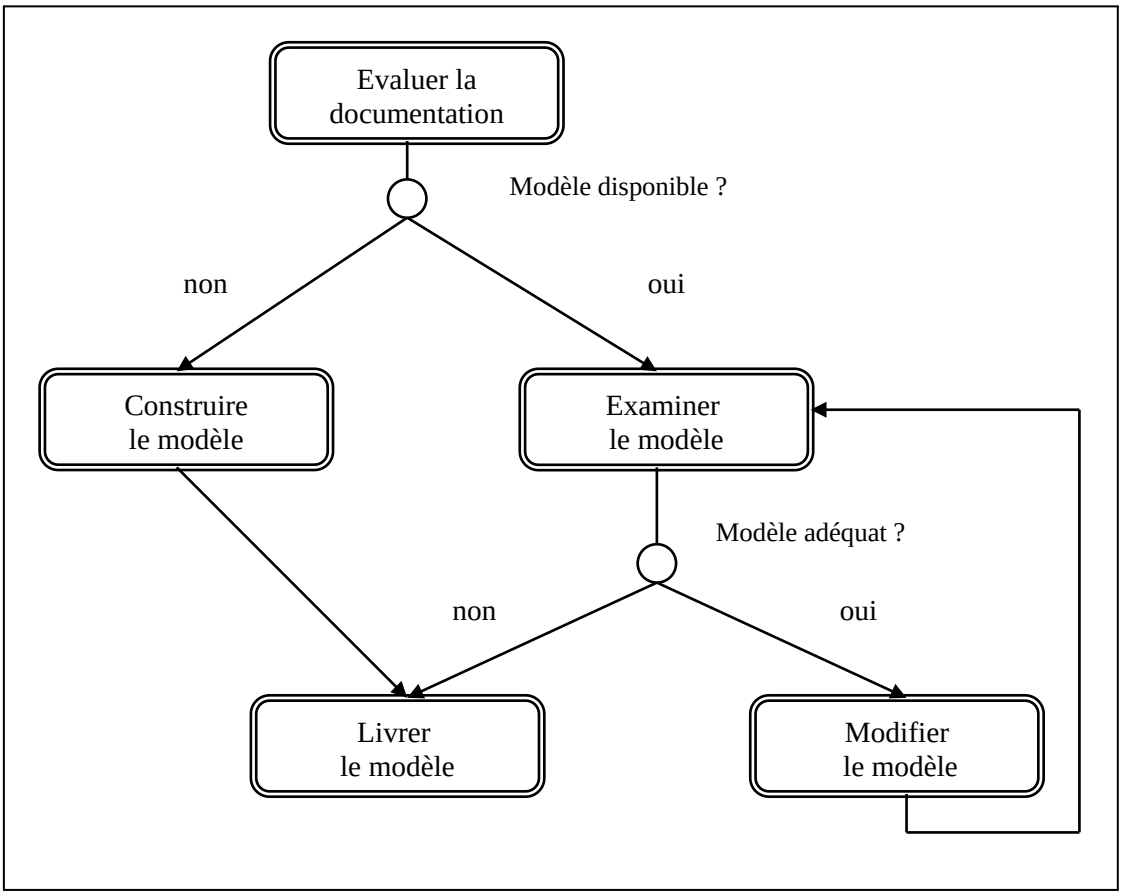


Figure 5.3 : Description d'un fragment de processus

Le processus commence par l'évaluation de la documentation et la recherche du modèle de données : En cas où celui ci existe, il sera examiné et vérifié après quoi :

- il sera livré s'il est valide (adéquat à la situation),
- ou modifié et adapté s'il ne l'est pas.

Dans le cas contraire un nouveau modèle doit être conçu ou construit.

5.3.3 LES RELATIONS ENTRE FRAGMENT DE METHODES

5.3.3.1 RELATIONS ENTRE FRAGMENTS DE MEME TYPE

- liens de granularité

ils se traduisent par la relation : « fragment de méthode **consiste en** fragment de méthode »

cette relation met en jeu deux fragments de même type et elle est valable pour les deux types de fragments(produits et processus). Ces liens expriment le fait qu'un fragment de méthode peut être décrit à plusieurs niveaux de granularité.

- Liens de précédence

Ces liens sont définis uniquement entre les fragments de processus et consistent à exprimer l'enchaînement des tâches et l'ordre de leur exécution, mais peuvent être étendus à certains fragments de produits et se traduisent par la relation : « fragment de processus **précède** fragment de processus ».

5.3.3.2 RELATIONS ENTRE FRAGMENTS DE TYPES DIFFERENTS

Liens de support -

Ces liens expriment, comme leur nom l'indique, le fait que les fragments conceptuels sont ou doivent être supportés par des fragments techniques et se traduisent par la relation : « fragment technique **supporte** fragment conceptuel ».

cette relation concerne aussi bien les fragments de produits que ceux des processus.

Liens d'entrées/sorties -

Ils décrivent les données d'un fragment en entrée ou en sortie et expriment le fait que les fragments de produits sont nécessaires pour le déclenchement de certains fragments de processus, qui eux mêmes, réalisent d'autres fragments de produits. Ils se traduisent par les relations :

« fragment de processus **nécessite** fragment de produit »

« fragment de processus **réalise** fragment de produit »

5.4 L'ASSEMBLAGE DES FRAGMENTS DE METHODES

5.4.1 LES PRINCIPES

Comme mentionné dans l'état de l'art , les approches de développement évoluent de l'utilisation d'une méthode rigide vers les méthodes situationnelles. Cette évolution est marquée par une augmentation de leur degré de flexibilité pour la considération des différentes situations et pour que cette flexibilité ne réduise pas les qualités attendues de ces approches, l'assemblage de fragments de méthodes doit être fait selon trois principes [Kan97] à savoir :

5.4.1.1 LA CONSTRUCTABILITE

Ce principe concerne d'une part, l'assemblage des fragments de processus et d'autre part, l'assemblage des fragments de produits d'une méthodes. L'assemblage des fragments de processus offre l'enchaînement des activités et des étapes de la méthode sans se soucier des techniques et des outils. L'assemblage des fragments de produits détermine les fragments de produits à utiliser dans une approche de développement ce qui consiste à pouvoir assembler, à partir de différents fragments de produits, un nouveau produit. Ainsi dans une activité de méta-modélisation , il y a une nécessite de disposer d'une meilleure considération de la réutilisation des fragments de méthodes.

5.4.1.2 L'ADAPTABILITE

Ce principe consiste à relier les fragments à partir d'autres déjà existants. Il peut également être vu selon deux aspects : Adaptabilité de fragments de produits et adaptabilité de fragments de processus. En effet, en ce qui concerne la flexibilité, il est possible d'enrichir le processus afin de guider les developpeurs dans le processus de transformation et éventuellement de faire retour en arrière pour retrouver la meilleure transformation afin d'établir une relation bijective entre les différents concepts.

5.4.1.3 L'UTILISATION DES SUPPORTS INFORMATISES

Ce principe consiste à utiliser des supports composés d'outils informatisés capables de supporter les fragments de méthodes et connus sous le nom de méta-outils, et d'outils informatisés servant à assembler ces fragments. Ces environnements multi-méthodes favorisent la réactivité aux changements et permettent aux constructeurs de décrire les méthodes et les liens entre elles puis de générer les outils correspondants, et aux développeurs d'utiliser les outils existant ou générés par l'environnement et de passer d'une méthode à une autre.

5.4.2 L'ACTIVITE DE TRANSFORMATION

Globalement on distingue deux catégories de transformation de modèles :

- La première consiste à effectuer des transformations sur un modèle en restant dans le même formalisme et ce pour des raisons d'optimisation, de normalisation ou de fusion.
- La deuxième consiste à créer un nouveau modèle décrit dans un autre formalisme , ce sont ces transformations qui nous intéressent et plus précisément celles qui relèvent des approches situationnelles à savoir les deux cas suivants :
  - Transformer pour passer d'un niveau d'abstraction à un autre.
  - Transformer pour convertir.

Ces types de transformations sont généralement difficiles du fait qu'elles doivent minimiser les pertes d'informations voire même les préserver si possible.

Dans pas mal de cas, les travaux sur les transformations proposent l'établissement d'une série de correspondances entre les concepts du modèle initial et ceux du modèle cible. Il est parfois difficile de faire des mises en correspondances directes ce qui a conduit à l'élaboration de stratégies de transformation.

5.4.2.1 LE PASSAGE D'UN NIVEAU D'ABSTRACTION A UN AUTRE

Il s'agit de décrire les modèles dans un niveau d'abstraction (conceptuel), puis de les transformer en des modèles dans un autre niveau d'abstraction (logique ou physique). Ces modèles possèdent beaucoup de correspondances structurelles qui rendent la transformation possible. La stratégie employée consiste à extraire des informations du modèle source et à leur faire correspondre une information dans le modèle destination. Ce type de transformation est facilité grâce à la traçabilité définie dans les méthodes. En effet, chaque méthode propose des règles de passage d'un niveau d'abstraction à un autre .

5.4.2.2 LA CONVERSION

Dans le développement avec des approches situationnelles, il arrive souvent d'avoir plusieurs modèles partiels à fusionner. La conversion des modèles se traduit par une série de transformations de tous les modèles partiels vers un modèle unique. Ce cas de transformation est donc utilisé pour passer d'une méthode à une autre qu'elles soient de même approche ou d'approches différentes.

La majorité des études comparatives donnent des correspondances directes d'un concept dans le modèle source en un autre concept du modèle cible et affirment qu'il y a des stratégies de passage d'une méthode à une autre [Jam98].

Deux solutions sont envisageables :

- L'intervention du concepteur pour effectuer la transformation
- L'introduction d'heuristiques permettant de faire ce choix

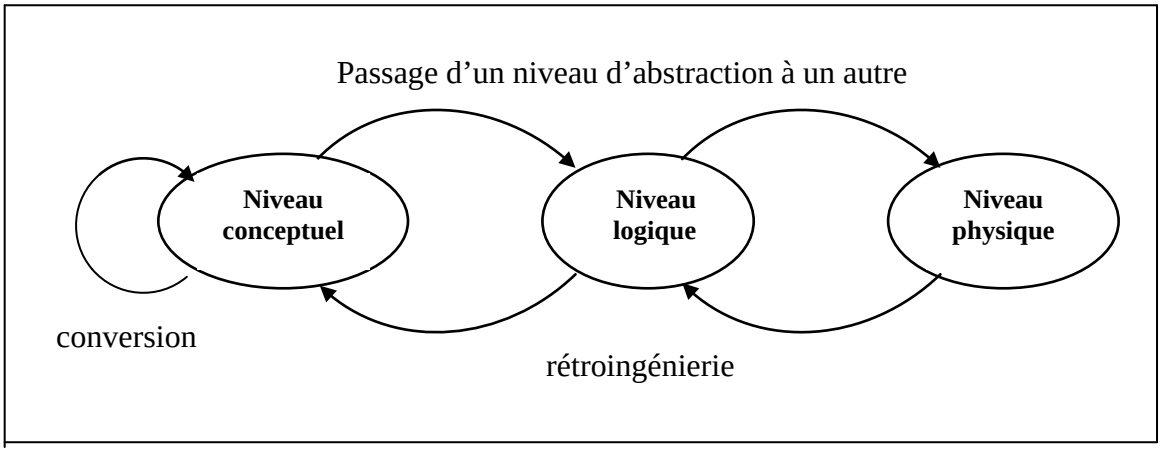


Figure 5.4 : les types de transformations

5.4.3 CORRESPONDANCES ENTRE LES METHODES

Pour ce qui est du passage d'une méthode orientée objet à une autre, il faut introduire les liens de correspondance entre les diverses notations des méthodes orientées objet et établir les liens d'héritage entre les concepts de méta-modélisation. Plusieurs études comparatives ont été faites, la plus importante est celle réalisée dans le cadre du projet COMMA (Common Object Methodology Metamodel Architecture), élaborée par Brian Henderson-Sellers et son équipe au sein du COTAR (Center for Object Technology Applications and Research) à l'université de SYDNEY [Heb 96].

Quatorze (14)méthodes orientées objet ont été concernées par cette étude, qui a porté sur l'aspect Statistique et dynamique des méthodes. Elle a permis d'une part , de comprendre les origines des méthodes et d'autre part, de situer les méthodes les unes par rapport aux autres et de souligner les liens qui existent entre elles.

Elle a montré également que certains modèles de méthodes ne diffèrent que par la forme de leur représentation et qu'il existe différentes techniques utilisées dans les méthodes qu'on ne peut intégrer en une seule, mais il est possible de juxtaposer nombreuses méthodologies par les différentes techniques manipulées.

La table suivante établit une correspondance entre les concepts des méthodes orientées objet les plus connues[Jac93].

OOSE	**OOA**	**OOD**	**HOOD**	**OMT**
Classe	Classe	Classe	(Objet classe)	Classe
Objet	Objet	Objet	Objet	Objet
Hérite	Structure Gen.Spéc.	Hérite	-	Généralisation
Accointance	Instance Conn. tout/partie	(Relation utilise)	(relation inclus)	Lien
Communication	Connexion message	(Relation utilise/instancie)	(Relation utilise)	(Flot de données)
Stimuli	Message	Message	(Stimuli)	Evénement
Opération	Service	Operation	Opération	Opération
Attribut	Attribut	Champ	(Type)	Attribut
Acteur	(Utilisateur)	-	(Objet environant)	-
Cas d'utilisation	(Fils d'exécution)	(Mécanismes)	(Flot de contrôle)	Scénario
Sous-système	(Sujets)	Catég.de classes sous-systèmes	Objet (racine)	Sous-système
Paquet de services	-	-	-	-
Bloc	-	Modules/catég. classe	Objet	Modules
Module Objet	-	Classe	Objet terminal	Classes
Module Obj.publ	-	(Visibilité de cat. de classes)	Interface Offerte	(Service)

Figure 5.5 : Table de correspondance des méthodes orientées objet

5.4.4 LES OPERATIONS DE MANIPULATION DES FRAGMENTS DE METHODES

Les opérations de manipulation des fragments de méthodes font partie des outils CAME. Elles sont nécessaires pour la construction des méthodes situationnelles et sont composées d'opérateurs simples facilitant l'utilisation des fragments depuis leur saisie jusqu'à leur suppression définitive de la base. En on distingue trois principaux types :

- **Les opérations d'administration**

Elles sont composées des opérateurs de base utilisés pour ajouter, modifier ou supprimer des fragments de la base.

- **Les opérations de sélection**

Elles sont utilisées pour restituer ,selon des critères définis, les fragments de la base et comprennent les opérateurs de comparaison, de sélection et d'extraction.

- **Les opérations d'assemblage**

Elles permettent de configurer les fragments sélectionnés dans une nouvelle méthode résultante et comprennent les opérateurs de conversion, de transformation, d'enchaînement et d'adaptation.

Lors de l'assemblage, plusieurs points de conformité ou de cohérence doivent être vérifiés et contrôlés afin de valider la consistance et l'enchaînement des différents fragments dans la méthode. Ces points concernent principalement les quatre types de relations entre fragments de méthodes (*Paragraphe 7.3.3*) et peuvent être représentés par les règles du formalismes suivant[Har94].

5.4.5 LA FORMALISATION DES FRAGMENTS DE METHODES

Soient :

- P : l'ensemble des fragments de processus
- D : l'ensemble des fragments de produits
- M : l'ensemble des fragments de méthodes avec $M = P \cup D$ et $P \cap D = \varnothing$
- Pc : l'ensemble des fragments conceptuels de processus
- Dc : l'ensemble des fragments conceptuels de produits
- Mc : l'ensemble des fragments conceptuels avec $Mc = Pc \cup Dc$ et $Pc \cap Dc = \varnothing$
- Pt : l'ensemble des fragments techniques des processus
- Dt : l'ensemble des fragments techniques de produits
- Mt : l'ensemble des fragments techniques avec $Mt = Pt \cup Dt$ et $Pt \cap Dt = \varnothing$
- g : la fonction but définie de M dans G(l'ensemble des buts)
- v : la fonction niveau de granularité définie de M dans N(l'ensemble des entiers)
- Consiste en(m1,m2) : la relation de granularité.

- Précède(m1,m2) : la relation de précédence.
- Les relations d'entrée/sortie.
  - Nécessite(p,d) :
  - Réalise(p,d) :
- Supporte(t,c) : la relation de support.

- Règle 1

Elle permet de vérifier que, dans une relation de granularité, les deux fragments sont de même type.

$\forall m1,m2 \in M$: [**Consiste en(m1,m2)** $\rightarrow (m1 \in P \ \wedge \ m2 \in P) \ \vee \ (m1 \in D \wedge \ m2 \in D)$].

- Règle 2

Cette règle permet de vérifier que le niveau de granularité d'un fragment est plus élevé que celui de ces composants

$\forall m1,m2 \in M$: [**Consiste en(m1,m2)** $\rightarrow \ v(m1) > v(m2)$].

- Règle 3

Est une heuristique qui indique que la différence entre les niveaux de granularité de deux fragments consécutifs dans une méthode ne doit pas dépasser 2

$\forall m1,m2 \in M$: [**Précède(m1,m2)** $\rightarrow \ |v(m1) - v(m2)| \leq 2$].

- Règle 4

Est une règle de conformité de précédence qui exprime le fait que si un fragment de processus nécessite un fragment de produit, celui-ci doit être réalisé auparavant par un autre fragment de processus

$\forall p2 \in P \ \forall d \in D \ \exists p1 \in P$ / [**Nécessite(p2,d)** $\rightarrow$ **Réalise(p1,d)** $\wedge$ **Précède(p1,p2)**].

- Règle 5

Est une règle de conformité de supports, elle qu'un fragment conceptuel doit être supporté par un fragment technique et impose que les deux fragments doivent avoir le même but

$\forall c \in Mc \ \forall t \in Mt$: [**Supporte(t,c)** $\rightarrow g(c) = g(t)$].

- Règle 6

Est une règle de conformité d'entrée/sortie, elle exprime la nécessité d'un fragment de processus pour la réalisation de chaque fragment de produit

$\forall d \in D \ \exists p \in P$ / [**Réalise(p,d)**].

5.5 L'AMELIORATION DE LA METHODE

L'utilisation de la démarche proposée pour la définition de la solution appropriée pour l'ingénierie des processus permet d'abord de se situer par rapport à cette situation et ensuite de mettre en œuvre une politique d'amélioration des processus qui s'adapte à son contexte. Cette politique est généralement influencée par les objectifs de l'organisation ainsi que par les problèmes et les contraintes susceptibles d'influencer la nature de la solution à mettre en œuvre.

Il s'agit donc de configurer des solutions d'amélioration spécifiques aux contextes des organisations considérées ; ceci nous amène à envisager trois scénarios d'amélioration : Le changement radical, le changement progressif et le rejet du changement. Ce dernier s'applique lorsque l'organisation est soumise à des contraintes d'ordre financier, technique ou humain ne lui permettant pas de changer sa solution.

5.5.1 SCENARIO DE CHANGEMENT RADICAL

Cette solution est préconisée si l'organisation considérée se trouve dans l'obligation de fonder sa propre politique de gestion des processus de développement pour faire face aux changements qui s'imposent et améliorer les pratiques d'ingénierie des processus. Cette solution ne se borne pas à la proposition d'une séquence d'actions d'amélioration, mais elle consiste plutôt à mettre en œuvre les pratiques de l'ingénierie des processus capables de satisfaire aux besoins spécifiques des projets et des processus.

5.5.2 SCENARIO DE CHANGEMENT PROGRESSIF

Ce Scénario s'applique si l'organisation dispose déjà d'une certaine pratique de l'ingénierie des processus et qu'elle souhaite l'améliorer. Elle opte pour un changement progressif en commençant par définir la solution qui s'adapte à son contexte spécifique ensuite gérer la résistance au changement qui pourrait en résulter. Pour vaincre cette résistance on doit opter pour la motivation et la formation du personnel sur ces nouvelles pratiques ce qui permettra d'estimer leurs apports dans l'amélioration de la qualité du développement.

La démarche proposée permet d'assister les producteurs de logiciels dans la définition de la solution technique appropriée en leur permettant d'analyser les besoins en matière d'environnement de développement et de déterminer la solution en fonction de ces besoins. Cette solution consiste à suivre les phases suivantes pour chaque modèle de processus à mettre en œuvre[Lam99].

- **La délimitation de la couverture**

La couverture du modèle de processus est déterminée selon la nature des processus (tactique, stratégique ou d'implémentation) et les propos qui leur sont assignés (prescriptifs ou descriptifs).

- **La détermination de la granularité**

La granularité d'un modèle de processus, qui peut être large ou fine, est fixée selon le niveau de détail requis lors du guidage et du traçage des processus.

- **Le choix des styles de représentation**

Chaque modèle de processus doit utiliser le style de représentation adéquat(programme, script, hypertexte) qui dépend également du propos qui lui est assigné.

- **Le choix du modèle de processus**

Les Spécifications techniques des modèles de processus requis varient généralement d'un processus à un autre. Il s'agit de choisir entre plusieurs méthodes celles dont les modèles de processus répondent aux spécifications préalablement fixées et sélectionner celles qui s y adaptent le plus.

- **La mise en œuvre du mécanisme d'exécution**

Il permet de déterminer les interactions entre les agents qui exécutent le processus afin de contrôler leur mise en œuvre et d'assurer le guidage et le traçage requis. Ce mécanisme est influencé par la forme du modèle de processus.

- **Le suivi des modèles de processus**

Le mécanisme d'exécution permet de guider le processus en cours conformément au modèle de processus, cependant l'exécution réelle peut diverger de ce qui est prévu. Le changement s'opère soit dynamiquement au cours de l'exécution du modèle de processus ou statiquement à la fin du projet.

- **La définition de la modularité**

A partir des propos, on peut décider de la nécessité en terme de modularité des modèles de processus correspondant. Lorsque plusieurs propos sont assignés à un même modèle de processus, c'est celui qui définit l'exécution qui sera retenu pour déterminer sa modularité.

VI CONCLUSION GENERALE

6.1 BILAN

La liberté de passage d'une méthode à une autre au cours du développement ou lors de la maintenance d'une application s'avère difficile. De plus si dans le même projet, les développeurs maîtrisent des formalismes différents; il serait difficile voir impossible de prendre en compte les habitudes de travail des développeurs. Or des travaux ont montré que la qualité des produits et la productivité des équipes de développement sont étroitement liées à la compétence des développeurs et au degré de maîtrise des techniques de modélisation qu'ils emploient[Har94]. on pense alors qu'il est utile de laisser cette liberté aux concepteurs en leur offrant la possibilité de naviguer dans un environnement de développement englobant plusieurs méthodes.

La préoccupation principale de ce travail a été l'étude de la considération du passage entre méthodes au cours du développement. on a étudié la nécessité de passage d'un formalisme (méthode,modèle) à un autre, les stratégies et les moyens informatisés supportant ce passage.

La première partie a présenté le cadre du sujet à savoir les concepts et les pratiques de développement des systèmes d'information ainsi que l'évolution historique des différentes approches de développement et des supports de modélisation des processus.

La deuxième partie a présenté l'état de l'art en ingénierie des systèmes d'information composé :

- D'un cadre de référence pour l'ingénierie des processus permettant la compréhension, l'évaluation et l'amélioration des processus.

- Des principales alternatives prédominantes aujourd'hui en ingénierie des méthodes pour le développement, la mise en œuvre et la maintenance des méthodes.

- Le méta-modèle NATURE qui décrit les notions de produits et de processus dans un concept contextuel, permettant ainsi leur compréhension et la facilité de leur modélisation.

La troisième partie a présenté les éléments et les phases nécessaires d'une démarche de développement de systèmes d'information par fragments de méthodes permettant l'activité de transformation et la navigation entre les méthodes en passant par les étapes suivantes :

- La sélection, dans un premier temps, des fragments(produits, processus) de différentes méthodes, décrits dans une structure bien définie et administrés dans une base de méthodes.

- Dans un deuxième temps, l'assemblage selon diverses techniques, des fragments sélectionnés pour aboutir à une méthode résultante.

- Cette dernière peut bien sûr être améliorée au cours du développement. Cette politique est généralement influencée par des objectifs de l'organisation ainsi que par les problèmes et les contraintes susceptibles d'influencer la nature de la solution à mettre en œuvre.

- Tout ceci ne doit être fait qu'après la caractérisation du projet en fonction des valeurs de certains facteurs situationnels qui agissent et interagissent tout au long du déroulement du processus de développement et ayant une incidence sur la stratégie de conception à mettre en place.

6.2 PERSPECTIVES

les méta-outils ont, dans une certaine mesure, permis d'augmenter la productivité des équipes de développement et la qualité des produits logiciels. L'un des avantages de l'utilisation des méta-outils est que ces derniers sont capables de produire des outils supports de méthodes dans des intervalles de temps très courts en comparaison avec le temps nécessaire à leurs développement. En effet, en général le temps de production des outils est réduit au temps d'introduction, dans le méta-outil, des informations relatives à la méthode.

Cependant de nombreuses raisons ont fait qu'au niveau de la pratique, certains de ces méta-outils ne sont que des boites à outils qui ne permettent que le support de certaines activités et ne fournissent aucune aide à la réalisation de ces dernières. De plus ils sont difficilement adaptables à l'évolution des approches de développement et n'offrent malheureusement pas suffisamment de souplesse ni pour le passage entre méthodes, ni pour le maintien de cohérence de la base des modèles[Jam98].

Pour résoudre ces problèmes, il faut envisager de regrouper la description des méthodes et les liens entre elles dans une base d'information méthodologique et d'intégrer celle ci dans un environnement de développement multi-méthodes.

L'environnement multi-méthodes intégrant la base de méthodes doit regrouper les différents outils, supportant les méthodes agrées par une organisation de développement, et communiquant à travers un dictionnaire d'informations sur la base de description des différentes méthodes et les liens assurant leur rapprochement. Ceci offre la possibilité au concepteurs de méthodes de décrire les méthodes par rapport à un dénominateur commun et au développeurs de stocker leurs produits dans un formalisme et de les restituer dans un autre.

Et enfin, Il faut que le support de modélisation prenne en charge d'assurer que l'ensemble des modèles forme une base cohérente, soit en avertissant le concepteur après une modification, soit en propageant lui même les conséquences d'une modification apportée par le concepteur sur les autres modèles.

Annexe : PRESENTATION DES METHODES ORIENTEES OBJET

INTRODUCTION :

L'Approche objet est devenue incontournable pour le développement des systèmes. L'échelle du développement de la technologie objet a augmenté progressivement ces dernières années car elle offre en environnement sophistiqué qui maintient une pratique saine de l'ingénierie du logiciel. Devant cette admiration pour les techniques objets, il est naturel de s'attendre à une augmentation du nombre de méthodes. Ces méthodes ont une approche commune basée sur une triple perception du système d'information. Une description complète du système nécessite la prise en considération de trois dimensions [Ali 99b] :

- Une dimension statique :
qui représente la description de la structure des objets, sanctionnée par un diagramme d'objets.

- Une dimension dynamique :
qui sert à décrire les règles d'évolution les objets dans le temps et sanctionnée par un diagramme d'états.

- Une dimension fonctionnelle :
Qui permet la description des flux d'informations qui circulent entre les différents acteurs du système, sanctionnée par un diagramme des flux de données.

Nous présentons dans cet annexe les méthodes objets les plus diffusées, en mettant simplement en avant quelques points clés de comparaison afin de découvrir plus en détail chacune des méthodes.

1. LA METHODE OOD : (Object Oriented Design)

- **Auteur** : GRADY BOOCH

- **Origine** : crée au début des années 80 afin de rationaliser les développements des applications en langage ADA.

- **Date réelle de diffusion** : 1985.

- **Particularités** : méthode de développement centrée sur la spécification technique et l'implémentation, l'analyse n'est couverte que depuis1993.

- **Modčles** : modèle statique (objets/associations)
modèle dynamique(états/transitions)

- **Niveaux** : niveau logique et niveau physique pour le modèle statique.

- **Démarches** : Cycle de développement itératif et incrémental.

- **Formalisme** : L'objet est représenté par des nuages, L'association est représentée par des arcs reliant les nuages.

- **Langages** : ada, c++, smaltalk, eiffel.

2. LA METHODES OOA : (Object Oriented Analysis)

- **Auteur** : S.Shlaer & S.J.Mellor

- **Origine** : crée en 1979 au Lawrence Berkeley Laboratory.

- **Date réelle de diffusion** : 1988.

- **Particularités** : méthode conçue au départ pour le temps réel , une seconde version plus complète couvre les aspects dynamique et fonctionnel (1991).

- **Modčles** : modèle statique (relationnel avec le concept de généralisation).
modèle dynamique (diagramme états/transitions).
modèle fonctionnel (diagramme de flux et de processus).

- **Niveaux** : niveau conceptuel et niveau logique où l'approche objet est nettement intégrée.

- **Démarches** : Deux démarches à savoir la construction des modèles et l'analyse.
Découpage de l'application en quatre catégories de domaines disjoints et complémentaires :
Les domaines d'application, Les services, L'architecture et L'implémentation.

- **Formalisme** : le modèle statique est une synthèse des diagrammes de Bachmann et du modèle entités/associations. Le diagramme états/transitions est celui de REMORA.

- **Langages** : aucun en particulier

3. LA METHODE OOA/OOD : (Object Oriented Analysis/Design)

- **Auteur** : P.Coad & E.Yourdon.

- **Origine** : Diffusée par Object International aux U.S.A.

- **Date réelle de diffusion** : 1991.

- **Particularités** : elle se démarque de la méthode d'analyse structurées de Yourdon en apportant des améliorations dans la terminologie et les notations.

- **Modčles** : modèle statique uniquement. Il est composé de classes et associations (Binaire, agrégation, généralisation et spécialisation).
Décomposition du modèle en sous-Schémas regroupant les objets par sujet (domaines).

- **Démarches** : Le niveau conceptuel intègre les domaines , les objets, les associations entre objets, Les attributs des objets et les opérations sur les objets.
Le niveau logique permet d'introduire les objets techniques relatifs aux interactions homme-machine, à la gestion des tâches et au stockage des données.

- **Formalisme** : Le modèle statique est une extension du modèle entités/associations.

- **Langages** : aucun en particulier

4. LA METHODE HOOD : (Hierarchical Object Oriented Design)

- **Auteur** : B.DELATTE, M,HEITZ, J.F.MULLER.

- **Origine** : crée en 1987 pour l'ESA (agence spacialle européenne) par un consortium composé CISI ingénierie, MATRA espace et CRI.

- **Date réelle de diffusion** : 1991.

- **Particularités** : Méthode qui est un sous ensemble de la méthode OOD. Elle fournit des règles de conception sur la définition des classes d'objets. Elle est orientée uniquement vers les systčmes utilisant le langage ADA.

- **Modčles** : modèle statique (objet, association)

- **Niveaux** : niveau logique pour le modèle statique.

- **Démarches** : deux démarches à savoir :la construction des modèles et la démarche d'analyse.
  L'application est découpée en quatre catégories de domaines disjoints et complémentaires : les domaines d'application, les services, l'architecture et l'implémentation.

- **Formalisme** : le modèle statique est une synthèse de diagrammes de BACHMANN et du modèle entités/associations.
  Le diagramme états/transitions est celui de REMORA.

- **Langages** : ADA.

5. LA METHODE OMT : (Object Modeling Technique)

- **Auteur** : James Rumbaugh

- **Origine** : Crée en 1978/1979 par le centre de recherche et de développement de Général Electric.

- **Date réelle de diffusion** : 1991.

- **Particularités** : Suit OOA, mais plus riche en concepts et sa démarche permet de travailler sur des cas complexes.

- **Modčles** : Le modèle statique est dérivé du modèle entités/associations, Les associations peuvent être n-aires et porteuses de données, les notion d'agrégation et de généralisation sont gérées et les cardinalités sont utilisées pour préciser les relations.
Le modèle dynamique est présenté sous forme de diagramme états/transitions, la transition est caractérisée par l'événement déclencheur, les conditions et l'action à réaliser.
Le modèle fonctionnel utilise les diagrammes des flux qui relient les acteurs et les classes d'objets pour décrire les processus de l'application(fonction)

- **Niveaux** : Niveau conceptuel composé des trois modèles précités, niveau logique permet de faire le lien entre les trois modèles et de réaliser l'implémentation des composants.

- **Démarches** : Le cycle de développement suit le cycle en CASCADE.

- **Formalisme** : Le modèle statique est dérivé du modèle E/A
Les modèles état/transitions, de scénarios et fonctionnel sont basés sur les diagrammes de flux.

- **langages** : aucun en particulier.

6. LA METHODE OOSE : (Object Oriented Software Engineering)

- **Auteur** : ivar jacobson.

- **Origine** : crée en 1980 à partir d'une méthode suédoise.

- **Date réelle de diffusion** : 1992.

- **Particularités** : Elle accorde une place importante aux objets-interface qui constituent une première étape de la modélisation.

- **Modčles** : le modèle de besoins sert à saisir les besoins fonctionnels(délimiter les frontières du système).
  le modèle d'analyse donne au système une structure robuste et modifiable.
  le modèle de conception permet d'adopter et d'affiner la structure objet pour l'environnement d'implémentation.
  le modèle d'implémentation sert à implémenter le système.
  le modèle de test a pour but la vérification du système.

- **Niveaux** : Conceptuel, logique et physique

- **Démarches** : le cycle de développement est composé de trois phases : la phase d'analyse, la phase de construction et la phase de test.

- **Formalisme** : basé sur les cas d'utilisation où chaque cas peut comporter :
  des objets-entité (aspect statique).
  des objets de contrôle (aspect dynamique).
  des objets-interface (aspect fonctionnel).

- **Langages** : aucun en particulier.

Après avoir présenté les principales méthodes orientées objet, on présente les résultats d'une analyse faite en 1996 concernant les phases de spécification, de conception technique, de codage, de test et de validation[AFCET 97].

L'enquête a révélé que l'activité de codage qui se situe au quatrième rang sur une échelle de onze activités de priorité est considérée comme la mieux couverte par les outils logiciels (57%),
alors que l'activité de spécification placée au deuxième rang est couverte à (38%),
quant à l'activité de conception technique, elle est couverte à (35%) au cinquième rang
et finalement l'activité de test et validation est classée au troisième rang, représente un taux de couverture de (17%).

Les pourcentages de citation de méthodes utilisées au cours de ces phases du cycle de développement (toutes méthodes confondues) sont données comme suit :

- Analyse et spécification : 75%
- Conception technique : 40%
- Test et validation : 25%

BIBLIOGRAPHIE ET REFERENCES

[Ald91] : A.ALDERSON, 'Meta-Case Technology'
Proceding of the CAISE Juin 1991.

[Ali99] : Z.ALIMAZIGHI, 'Environnement de développement de S.I :une approche méthodologique basée sur l'analyse de documents et scénarios'
Thèse de doctorat d'état U.S.T.H.B Alger Mars 1999.

[Ali99b] : Z.ALIMAZIGHI, 'Développement de systèmes d'information '
Cours de P.G Informatique U.S.T.H.B Alger 1999.

[Ben98] : M.BENABDELATIF, 'Modélisation de systèmes de développement de S.I application à la méthode OMT.'
Thèse de Magister U.S.T.H.B Alger 1998.

[Bub86] : J.R.BUBENKO, 'Information system methodologies a research view.'
Information System Design Methodologies Amsterdam 1986.

[Buw92] : J.R.BUBENKO, B.WRANLER, 'Research direction in conceptual specification development'
An integrated view of information systems development John Wiely & Sons Inc 1992.

[Chr95] : C.PASQUIER, P.ROUCOULET, M.LANASPEZE, 'L'approche objet'
Editions Hermes Paris 1995.

[Col91] : R.CONDARI, C.LIU, 'Process Modelling Paradigms : an evaluation'
first euopean workshop on software process modelling Milan 1991.

[Dei89] : W.DEITERS,V.GRUHEN, W.SHAEFER, 'Systematic development of formal software process models'
Proceding of The European Software Engineering Conference(ESEC) 1989.

[Des90] : C.DESCLAUX, A.GUTHAUSER, P.STORR, A.THIERRY, 'Design and maintenance rational under MACS'
Conférence sur Le génie logiciel et ses applications Toulouse Decembre 1990.

[Dow87] : E.M.DOWSON, 'Iteration in the software process'
9th International Conference On Software Engineering Californie 1987.

[Feh91] : P.H.FEILLER, W.S.HUNPHREY, 'Software process development and enactment'
SEI Report Octobre 1991.

[Fin90] : A.FINKELSTEIN, J.KRAMER, M.GOEDICKE, 'View point oriented software development'
Conférence sur Le génie logiciel et ses applications Toulouse Décembre 1990.

[Fra91] : M.FRANKSON, C.PEUGEOT, 'Specification of the object and process modelling language'
ESF Report #D122-OPML-1.0 1991.

[Fug93] : A.FUGGETTA, 'A classification of case technology'
IEEE computer vol. 8 N° 1 Décembre 1993.

[Gna99] : G.GNAHO, J.BARRIOS, 'A web based tool for helping I.S ingineers in method use'
Proceding of the 2nd MFPE Tunis Mai 1999.

[Har94] : F.HARMSEN, S.BRINKKEMPER, H.OEI , 'Situational method engineering for information system project approaches'
Proceding of the IFIP WG 8.1 Working Conference Maastricht Septembre 1994.

[Heb96] : B.S.HENDERSON, A.BULTUIS, 'The COMMA project : first draft for limited circulation'
COTTAR Technical Report Sydney 1996.

[Hhg97] : H.H.BENGHEZALA,Y.JAMOUSSI, 'une approche de méta-modélisation favorisant la réactivité au changement'
Proceding of the First MFPE Tunis Septembre 1997.

[Hum89] : W.S.HUMPHREY, 'Managing the software process'
Editions Addison Wesley 1989.

[Iat93] : J.C.LANESELLI, C.TRIOLAIRE, 'expériences d'évaluation des processus de developpement'
6eme Conférence sur le génie logiciel et ses applications Paris Décembre 1993.

[Jac93] : I.JACOBSON, 'Le génie logiciel orienté objet'
Editions Addisson Wesley 1993.

[Jam98] : Y.JAMOUSSI, 'Un support de modélisation évolutif favorisant le passage entre méthodes'
Thèse de Doctorat Faculté des Sciences Tunis Février 1998.

[Jar93] : M.JARKE, 'Requirment engineering : an integrated view of representation, process and domain'
Proceding of the 4th european Software Engineering Conference 1993.

[Jar92] : M.JARKE, J.MILOUPOLOS, J.W.SCMIDT, Y.VASSILIOU, 'DAIDA an environment for evolving information systems'
ACM trans. On Information Systems Vol 10 N° 1 1992.

[Kat89] :T.KATAYAMA, 'A hierarchical and functional process description and its enaction'
Proceding of the 11th inter. Conference On Software Engineering Pensylvanie 1989.

[Kuw92] : K.KUMAR, R.J.WELKE, 'Methodology engineering : a proposal for situation specific methodology costruction'
Challenges and strategies for research in system development 1992.

[Lam99] : L.H.MEHRI, N.KRAIEM, H.H.BENGHEZALA, 'Une démarche pour l'ingénierie des processus'
Proceding of the 2nd MFPE Tunis Mai 1999.

[Lee91] : J.LEE, 'Extending the potts and bruns model for recording design rationale'
Proceding of the 13th international Conference On Software Engineering Austin 1991.

[Mfp97] : F.KAMOUN, G.GROSZ
Proceding of the first MFPE Tunis Septembre 1997.

[Ost87] : L.OSTERWEIL, 'Software processes are software too'
Proceding of the 9th international Conference On Software Engineering Washington 1987.

[Pad98] : P.SILVESTRE, D.VERLHAC, 'Stratégie de conception de systèmes d'information'
Les techniques de l'ingénieur H5170 Paris1998.

[Pot89] : C.POTTS, 'Ageneric model for representing design methods'
Proceding of the 11th international Conference On Software Engineering Pensylvanie 1989.

[Pot88] : C.POTTS, J.BRUNS, 'Recording the reasons for design decisions'
Proceding of the 10th international Conference On Software Engineering 1988.

[Ram92] : B.RAMESH, V.DHAR, 'Supporting systems development by capturing deliberations during requirments engineering'
IEEE transactions on software engineering Vol 18 N° 6 Juin 1992.

[Rfb88] : C.ROLLAND, O.FOUCAUT, G.BENCI, 'Conception des systèmes d'information, la méthode REMORA'
Editions Eyrolles Paris 1988.

[Rol97] :C.ROLLAND, 'A primer for method engineering'
Acte des conférences du 15ème congrès INFORSID Toulouse Juin 1997.

[Rol96] : C.ROLLAND, ' L'ingénierie des processus de développement, un cadre de référence'
Ingénierie des systèmes d'information Vol 4 N°6 1996.

[Rol95] : C.ROLLAND, C.SUVEYET, M.MORENO, 'An approach for defining ways of working'
Information system journal Vol 20 N°4 1995.

[Rol92] : C.ROLLAND, 'Outils CASE : état de l'art et perspectives'
Acte des 6èmes journées internationales des sciences de l'informatique Tunis Mai 1992.

[Sah95] : H.A.SAHRAOUI, 'Application de la meta-modélisation à la génération des outils de conception et de mise en œuvre des bases de données'
Thèse de doctorat Paris6 1995.

[See97] : J.EBERT, 'Proceding of the 8th international conference on software engineering environments' Berlin Avril 1997.

[Sho91] : K.W.SHORT, 'Methodology integration : evolution of information engineering'
Information and Software Technology Vol 33 N°9 Novembre 1991.

[Swv93] : V.SWEDE, J.C.VLIET, 'A flexible framework for contingent information systems modelling'
information and software tecnology Vol 39 N°9 1993.

[Vau93] : D.VAUQUIER, 'Développement orienté objet'
Editions Eyrolles Paris 1993.

[Wil88] : L.G.WILLIAMS, 'Software process modelling : Abehavioral approach'
Proceding of the 10th international conference on software engineering 1988.