Approche de Développement centré décideur et à l'aide de patrons de Systèmes Interactifs d'Aide à la Décision - Application à l'investissement dans le domaine ferroviaire

Sophie Lepreux

▶ **To cite this version:**

Sophie Lepreux. Approche de Développement centré décideur et à l'aide de patrons de Systèmes Interactifs d'Aide à la Décision - Application à l'investissement dans le domaine ferroviaire. Interface homme-machine [cs.HC]. Université de Valenciennes et du Hainaut-Cambresis, 2005. Français. NNT : . tel-00010611

HAL Id: tel-00010611
https://theses.hal.science/tel-00010611

Submitted on 13 Oct 2005

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Université de Valenciennes et du Hainaut-Cambrésis

Numéro d'ordre : 05-06

THÈSE

pour l'obtention du

Doctorat de l'Université de Valenciennes et du Hainaut-Cambrésis

Spécialité Automatique et Informatique des Systèmes Industriels et Humains
Mention Informatique

Sophie Lepreux

Maître ès Sciences

Approche de développement centré décideur et à l'aide de patron de Systèmes Interactifs d'Aide à la Décision

Application à l'investissement dans le domaine ferroviaire

soutenue publiquement le 02 juin 2005 devant le jury composé de :

M. Abed	Professeur à l'Université de Valenciennes	Co-Directeur
A. Derycke	Professeur à l'Université de Lille 1	Rapporteur
A. Fréville	Professeur, Conseil Régional Nord - Pas de Calais	Président
C. Kolski	Professeur à l'Université de Valenciennes	Directeur
J. Nanard	Professeur à l'Université de Montpellier	Rapporteur
G. Quéric	Services études économique et générales, direction de la stratégie et du développement à Réseau Ferré de France	Invité

Laboratoire d'Automatique, de Mécanique et d'Informatique industrielles et Humaines
CNRS UMR 8530

Université de Valenciennes et du Hainaut-Cambrésis

Numéro d'ordre : 05-06

THÈSE

pour l'obtention du

Doctorat de l'Université de Valenciennes et du Hainaut-Cambrésis

Spécialité Automatique et Informatique des Systèmes Industriels et Humains
Mention Informatique

Sophie Lepreux

Maître ès Sciences

Approche de développement centré décideur et à l'aide de patron de Systèmes Interactifs d'Aide à la Décision

Application à l'investissement dans le domaine ferroviaire

soutenue publiquement le 02 juin 2005 devant le jury composé de :

M. Abed	Professeur à l'Université de Valenciennes	Co-Directeur
A. Derycke	Professeur à l'Université de Lille 1	Rapporteur
A. Fréville	Professeur, Conseil Régional Nord - Pas de Calais	Président
C. Kolski	Professeur à l'Université de Valenciennes	Directeur
J. Nanard	Professeur à l'Université de Montpellier	Rapporteur
G. Quéric	Services études économique et générales, direction de la stratégie et du développement à Réseau Ferré de France	Invité

Laboratoire d'Automatique, de Mécanique et d'Informatique industrielles et Humaines
CNRS UMR 8530

Remerciements

Je tiens à adresser mes remerciements en tout premier lieu à mes directeurs de thèse les professeurs Christophe Kolski et Mourad Abed pour leurs collaborations, encadrements, soutiens, encouragements tout au long de ce travail. Ils m'ont permis d'effectuer ce travail dans de bonnes conditions au sein de l'équipe RAIHM (Raisonnement Automatique et Interaction Homme-Machine) du LAMIH (Laboratoire d'Automatique, de Mécanique, d'Informatique industrielles et Humaines).

Mes remerciements vont ensuite aux partenaires financiers, Réseau Ferré de France (RFF) et la Région Nord - Pas-de-Calais, sans qui ces travaux n'auraient pas eu lieu.

Je souhaite également remercier les rapporteurs de ces travaux, les professeurs Jocelyne Nanard et Alain Derycke, qui ont accepté de lire, critiquer et valider ces travaux.

Je remercie Arnaud Fréville, professeur à l'Université de Valenciennes et du Hainaut-Cambrésis et membre du conseil régional du nord - Pas-de-Calais, pour avoir lu mes travaux et avoir accepté de présider le jury de thèse. Mes remerciements vont également à monsieur Guénhaël Quéric, membre de la direction du développement, qui m'a accompagnée durant ces trois années et a accepté de représenter Réseau Ferré de France dans le jury.

Je tiens ensuite à remercier Michel Legendre pour sa disponibilité, ses conseils, ses encouragements, son soutien et sa passion du ferroviaire qu'il a su me transmettre tout au long de ces quatre années. Sans lui, cette thèse n'aurait pas eu lieu et n'aurait pas pu aboutir.

Je remercie les personnels de RFF et les collègues du LAMIH et des autres laboratoires qui ont contribué de près ou de loin au bon déroulement de cette thèse.

Pour finir, mes remerciements iront à ma famille et mes amis qui sont restés présents quelles que soient les épreuves et qui m'ont toujours soutenue.

Table des matières

Liste des figures

Liste des tableaux

Introduction générale

Réseau Ferré de France (RFF) est né en 1997. Cet Etablissement Public Industriel et Commercial (EPIC) est propriétaire et gestionnaire du réseau ferroviaire français. RFF a depuis sa création la responsabilité des infrastructures ferroviaires, de leur développement et de leur financement. Environ 750 millions d'euros sont mobilisés chaque année sur le compte d'investissement de RFF pour le renouvellement et la modernisation des infrastructures [RFF, 2005]. L'émergence d'un projet se fait sur la base d'une demande sociale, d'une concertation avec les co-financeurs, d'une optimisation technique et d'une bonne insertion environnementale. Le projet est ensuite mené suivant trois étapes :

- La définition des objectifs d'amélioration,
- Les études préliminaires permettant de vérifier l'opportunité de ces objectifs, de déterminer les travaux d'infrastructures nécessaires pour les atteindre et d'évaluer leurs coûts et leurs délais.
- Les études d'avant-projet permettent de finaliser la consistance des travaux, de réduire les incertitudes administratives et financières, de quantifier les impacts et de vérifier la cohérence du projet avec la stratégie de développement de RFF.

Pour mener à bien ces études, RFF a besoin d'un système lui permettant de gérer l'ensemble du projet. Ces études étant menées par divers acteurs (chargés d'études ferroviaire, chargés d'études socio-économiques, décideurs), ce système devra être adapté à l'ensemble des acteurs. Dans ce cadre, les Systèmes Interactifs d'Aide à la Décision (SIAD) pourront intervenir dans les situations complexes dans lesquelles les utilisateurs doivent se baser sur leurs connaissances pour analyser le problème et prendre des décisions, de tels systèmes seraient appropriés au problème de RFF.

Le premier chapitre a pour but d'analyser les SIAD ; les définitions et l'évolution des technologies seront présentées dans une première partie. Les SAD et les SIAD seront alors bien distingués pour montrer l'importance de l'interaction homme-machine (IHM) dans le processus de prise de décision. La seconde partie montrera qu'il y a de nombreux points communs entre SIAD et systèmes de connaissance, la décision étant, dans de nombreux cas, basée sur la connaissance. Cette partie concernera donc les systèmes de connaissance. Ensuite, la nécessité d'intégrer la réutilisation dans les systèmes nous a amené à étudier, dans une troisième partie, les systèmes à base de composants. Les différentes sortes de composants seront évoquées. Parmi ceux-ci, trois types de composants retiendront notre attention : les composants métier, les composants de conception et les composants logiciel. Les travaux existants dans le domaine ferroviaire feront l'objet d'une synthèse. En particulier les systèmes d'aide au calcul de capacité, les simulateurs et les outils d'évaluation de grille horaire seront traités car ils fournissent des critères de décision en investissement. Les insuffisances de ces outils seront également introduites.

Le second chapitre propose un état de l'art sur les approches de développement des systèmes présentés auparavant. Les modèles, les méthodes et les architectures sont exposés par domaine, du plus général au plus spécifique. Le premier domaine présenté est alors celui du génie logiciel. Ensuite, nous nous

sommes intéressés au domaine des Interactions Homme-Machine du fait que les interactions jouent un rôle primordial dans les SIAD. Les approches issues des domaines de la gestion de la connaissance et des systèmes à base de composants ont également étaient traitées. Ce chapitre se termine avec une présentation des approches spécifiques au développement de SIAD.

Dans ce cadre de recherche, une approche pour le développement de SIAD est proposée dans le troisième chapitre. L'approche proposée, nommée Approche de Développement de Système Interactif d'Aide à la Décision (ADESIAD), combine plusieurs principes issus des domaines étudiés dans le second chapitre. De manière à favoriser le développement des interactions homme-machine dans les SIAD, les utilisateurs ont été intégrés aux étapes de développement. Les experts ont eux aussi été intégrés dans le cycle pour améliorer la gestion de la connaissance. Le développement est basé sur l'intégration de composants, éventuellement conçus en parallèle du cycle de développement du système pour une meilleure évolutivité et pour faciliter la réutilisation. Enfin, l'évaluation est omniprésente car elle est à la fois centrale et terminale. Le modèle, la méthode et l'architecture, de l'approche proposée, ont été définis de manière à respecter ces principes, issus de modèles existants dans les divers domaines présentés et établis comme étant la base d'ADESIAD.

Le développement du Système Interactif d'Aide à la Décision relatif aux Investissements dans les infrastructures Ferroviaires (SIADIF) a été mené sur la base d'ADESIAD. Après avoir introduit les travaux antérieurs permettant de montrer le contexte de ces travaux, le quatrième chapitre a pour objectif de rapporter les étapes de développement de SIADIF. Le rôle qu'ont joué les utilisateurs et les experts dans le développement sera précisé. ADESIAD étant un cycle itératif, le travail présenté concerne particulièrement la première itération. A chaque étape, les résultats et les difficultés rencontrés seront présentés. Il sera alors possible de se rendre compte de l'évolution du système au cours de son développement. Le système, à la fin de la première itération, doit être évalué par les utilisateurs. Une évaluation du système, sans intervention des composants métier, a eu lieu pour valider son principe de fonctionnement. Les résultats de cette évaluation sont exposés dans la dernière partie de ce quatrième chapitre.

Le cinquième chapitre a pour but de proposer un bilan global d'ADESIAD et des principes mis en œuvre dans cette approche. Sur la base de constats issus du développement de SIADIF, certaines hypothèses et principes pourront être validés. D'autres aboutiront à des perspectives de recherche en terme d'amélioration d'ADESIAD, qui seront présentées en seconde partie de ce cinquième chapitre. Parmi ces perspectives, nous retrouverons l'ouverture d'ADESIAD à d'autres courants de recherche tels que l'apprentissage, le travail collaboratif ou la personnalisation du système à l'utilisateur.

Chapitre 1

Les systèmes interactifs d'aide à la décision

Sommaire

1.1 Introduction

Ce chapitre a pour but de présenter les systèmes d'aide à la décision. Ces systèmes sont apparus au cours des années soixante. Il y a eu depuis de nombreuses éévolutions. La première partie de ce chapitre est axée sur les définitions permettant de retracer l'historique de ces systèmes et les évolutions technologiques dans le temps. Seuls les systèmes interactifs d'aide à la décision, tels que nous les définissons, nous intéresserons par la suite. Les types de décision et les sortes de processus de prise de décision sont nombreux. Certaines décisions nécessitent l'accès à des connaissances plus approfondies que d'autres. Il y a donc un lien entre les systèmes d'aide à la décision et les systèmes de connaissance. Pendant longtemps, ce lien était fait par les systèmes experts. En conséquence, un état de l'art sur les systèmes à base de connaissance et leur évolution sera présenté. La connaissance peut provenir de diverses sources, elle sera analysée selon deux approches : le capital humain et le capital d'information. L'évolution des systèmes de connaissance permet de montrer l'enjeu économique induit par la gestion des connaissances et par conséquent par la réutilisation des connaissances. De la même manière, la réutilisation a un rôle à jouer dans les Systèmes Interactifs d'Aide à la Décision (SIAD). Dans la troisième partie, elle sera analysée selon deux points de vue : la réutilisation des connaissances et des idées et la réutilisation des composants et des artefacts. Les développements à base de composants seront abordés sous l'angle de la réutilisation de code. Pour finir, la quatrième partie sera axée sur la problématique applicative de la thèse concernant l'aide à la décision dans les investissements ferroviaires. Des outils représentatifs existants dans ce domaine, plus particulièrement en terme d'outils d'analyse des infrastructures, seront présentés et critiqués.

1.2 SIAD

On ne peut étudier les Systèmes Interactifs d'Aide à la Décision (SIAD) sans aborder les autres systèmes d'aide à la décision qui les ont précédés. Cette partie commencera en traitant les systèmes d'aide à la décision d'une manière générale puis se focalisera sur les SIAD. Comme l'a précisé Arun Sen [Sen, 1998], la difficulté des systèmes d'aide à la décision est que leur recherche a été fragmentée et dispersée à travers plusieurs domaines tels que les systèmes de gestion de bases de données, l'intelligence artificielle, le génie logiciel, les interfaces utilisateurs graphiques, et les sciences cognitives. La première section donnera les principales définitions et l'historique des systèmes d'aide à la décision. Ensuite, l'évolution des technologies dans les SIAD sera abordée en seconde section.

1.2.1 Informatique décisionnelle

Nous allons tout d'abord rappeler les premières définitions des systèmes d'aide à la décision et les principaux processus de prise de décision.

1.2.1.1 Définitions

En 1965, Anthony [Anthony, 1965] a décrit les activités de gestion comme correspondant à une planification stratégique, un contrôle de gestion et un contrôle opérationnel. Il ne parle pas encore de système d'aide à la décision ; il faudra attendre la définition de Gorry et Scott Morton qui ont défini un SIAD comme un système informatique qui supporte le décideur dans des situations de prise de décision non structurée[1] [Gorry and Scott Morton, 1971]. Ensuite, pour Keen et Scott Morton les systèmes interactifs d'aide à la décision couplent les ressources intellectuelles des individus avec les capacités des ordinateurs pour améliorer la qualité des décisions [Keen and Scott-Morton, 1978]. Ce sont des systèmes d'aide logiciels pour les décideurs de direction qui traitent des problèmes semi-structurés. La notion de coopération homme-machine apparaît et gardera une place importante dans la suite de l'histoire de ces systèmes. Nous nous intéresserons dans la suite du mémoire à l'importance de l'interaction homme-machine dans la conception de SIAD.

Les systèmes d'information apparaissent dans les définitions, d'une part, de Mallach pour qui un SIAD est un système d'information qui a pour mission principale de fournir aux décideurs les informations sur lesquelles seront basées les décisions, et d'autre part de Checroun [Checroun, 1992] qui voit les SIAD comme des systèmes d'information interactifs destinés à aider les décideurs à exploiter des données et des modèles pour résoudre des problèmes peu ou non structurés. Ce dernier auteur donne une signification à chaque mot clé de sa définition :

- Système : ensemble complexe et maîtrisable (au sens pilotable).
- Interactif : couplage homme-machine qui sous-entend ergonomie et contrôle par l'utilisateur. L'utilisation conversationnelle de l'ordinateur est nécessaire. Le dialogue est dirigé par le système et non par l'homme.
- Données et modèles : le système d'information comporte non seulement les informations brutes mais aussi les traitements nécessaires à une mise en forme compréhensible (tris, sélections, calculs, éditions), de même que les outils élaborés pour analyser, comprendre, communiquer, démontrer…
- Problèmes non structurés : c'est le lot commun à tous les problèmes posés par le management. Une grande part est faite à l'intuition, au tâtonnement, à l'expérience du décideur. Le SIAD ne constitue qu'un élément du processus de décision.
- Aider : il s'agit de fournir au décideur une amplification du pouvoir de raisonnement et non pas de se substituer à ce raisonnement par une modélisation des processus qui caractériseraient ce dernier.

Cette définition nous satisfait en partie. Les points sur lesquels nous ne sommes pas en accord concernent le fait que pour ces auteurs (1) un SIAD ne constitue qu'un élément du processus de décision car nous pensons que le SIAD doit intervenir tout au long de ce processus ; (2) en outre, le dialogue dirigé uniquement par le système ne nous semble pas cohérent avec une bonne prise en compte de l'utilisateur.

Pour nous, un SIAD est avant tout un système interactif. Il doit assister un décideur tout au long de son processus de décision par des interactions adaptées. Il est composé d'outils de mesure, d'analyse, de

[1] Selon Checroun, Simon [Simon, 1960] défi nit une décision non structurée comme un processus qui ne peut être décrit en détail avant la prise de décision. La prise de décision peut être entièrement, peu ou pas structurée.

comparaison qui doivent l'aider dans l'évaluation des solutions possibles. Les interactions entre l'utilisateur, le SIAD et l'ensemble des outils permettent à l'utilisateur de prendre une décision. Pour cela, les interactions entre le SIAD et l'utilisateur doivent respecter les processus de prise de décision de l'utilisateur. Nous excluons des SIAD les systèmes de décision qui sont des systèmes fermés à l'utilisateur.

Les processus de décision tiennent une place importante dans les systèmes de décision, ils vont être présentés dans la section suivante.

1.2.1.2 Processus de prise de décision

Quelques processus de décision vont être présentés dans l'ordre chronologique de leur proposition pour montrer l'évolution et les différentes vues qu'en ont eu les auteurs.

Simon [Simon, 1972] est un des premiers auteurs à décrire le processus de prise de décision, il le situe selon trois phases :

- L'intelligence consiste à rechercher des conditions nécessaires aux décisions,
- La conception intègre l'invention, le développement et l'analyse de séries d'actions possibles,
- Le choix consiste à sélectionner une série d'actions parmi celles disponibles.

Pour Checroun [Checroun, 1992], le processus de décision peut être regroupé selon trois principes :

- Le principe rationnel (économique) où les processus de décision doivent maximiser la valeur attendue du résultat en déterminant les coûts et risques de chaque alternative.
- Le principe heuristique où la prise de décision consiste à rechercher la première alternative dont le rapport coût/efficacité soit acceptable.
- Le principe du consensus qui préconise d'effectuer des comparaisons successives entre les alternatives jusqu'à obtenir le consensus des décideurs.

Quelques années plus tard, Balasubramanian et al. [Balasubramanian et al., 1999] ont présenté le processus de décision comme étant en cinq étapes :

- La définition du contexte et du but de la décision. La décision est faite pour réaliser un but, résoudre un problème ou pour construire un projet. Les buts et les objectifs sont structurés suivant une hiérarchie contribuant à un but père,
- L'identification ou génération des options à considérer,
- La spécification des facteurs, hypothèses, raisons et autres informations pertinentes à considérer,
- L'évaluation des options par rapport aux facteurs, hypothèses et autres variables pertinentes pour prendre une décision,
- La promulgation de la décision avec examen des résultats.

Dans nos travaux, le but n'est pas de proposer un processus de décision, l'important est de montrer qu'il existe et, avant tout que ce processus doit être respecté lors de l'interaction entre un utilisateur (décideur) et un SIAD.

Rasmussen [Rasmussen, 1983] fait émerger **trois types de comportement** dans un contexte de prise de décision : des comportements basés sur l'habileté (ou l'entraînement), les règles ou la connaissance, cf. figure1.1. Le comportement basé sur les compétences représente une performance sensori-motrice

pendant une action ou une activité qui prend place inconsciemment. La personne n'est pas consciente de choisir parmi plusieurs moyens d'action. Le niveau de comportement intermédiaire correspond à celui basé sur les règles. La composition d'une séquence de sous-routines, dans une situation de travail familière, est contrôlée consciemment par une corrélation signal - réaction empirique. La personne est consciente que des actions alternatives sont possibles et doit faire un choix. Durant des situations non familières, pour lesquelles aucun savoir-faire ou règle de contrôle ne sont disponibles, la performance est contrôlée par le but et basée sur la connaissance. C'est à ce dernier cas que nous nous intéresserons dans cette thèse. La gestion de la connaissance sera présentée dans la partie suivante (§ 1.3).

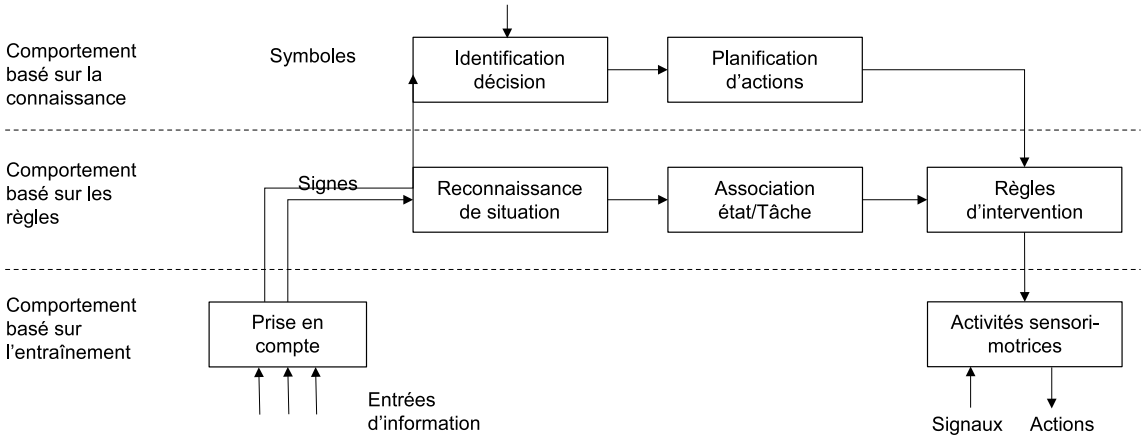


FIGURE 1.1 – Types de comportements [Rasmussen, 1983]

D'après Checroun [Checroun, 1992], Alter [Alter, 1977] essaye de classifier 56 systèmes ayant des caractéristiques de SIAD suivant deux catégories : systèmes orientés "données" et systèmes orientés "objets" mais ces deux approches ne donnent pas satisfaction aux décideurs principalement à cause du manque d'équilibre entre les fonctions offertes. Ceci montre qu'il est difficile de classifier les SIAD et amène à présenter l'évolution des technologies dans ce domaine.

1.2.2 Evolution des technologies dans les SIAD

Jusqu'aux années 70, les systèmes de décision étaient associés à des méthodologies propres à la recherche opérationnelle, l'analyse des données, le calcul optimal, etc. Le but de ces systèmes était de résoudre des problèmes par la recherche d'une solution optimale en calculant les maxima ou les minima de fonctions mathématiques exprimant un objectif à atteindre. Grâce aux progrès informatiques l'une des évolutions des systèmes d'aide à la décision a été de se rapprocher des utilisateurs pour leur permettre d'intervenir dans le processus de décision [Maret and Pinon, 1997].

Dans leur article, Shim et al. [Shim et al., 2002] présentent diverses technologies récentes utilisées dans les SIAD. Au début des années 1990, quatre outils puissants émergent. Le premier fut les entrepôts de données (data warehousing) puis une technique d'analyse de ces données (on-line analytical processing OLAP) et l'interprétation de ces données (data mining), le dernier outil est associé au World Wide Web [Bret et al., 2001]. Ces quatre types d'outils permettent d'analyser des données d'une manière

de plus en plus sophistiquée. L'arrivée du Web a introduit le SIAD basé sur le web ; c'est un système informatique délivrant une information d'aide à la décision ou des outils d'aide à la décision pour un gestionnaire ou un analyste utilisant un explorateur web tel que Netscape Navigator ou Internet Explorer [Power, 2002]. Les systèmes d'aide collaboratifs ont émergé il y a une vingtaine d'années, avec l'apparition des réseaux locaux qui ont permis un partage des informations visant à faciliter les prises de décision. Ces systèmes comprennent :

- les processus en groupe supportant la prise de décision où la décision de groupe résulte des communications interpersonnelles parmi les membres du groupe,
- les systèmes d'aide au groupe (Group Support Systems GSS) qui facilitent la communication et la coordination des activités des membres d'une équipe, et
- les équipes virtuelles qui sont des équipes de collègues géographiquement distribuées pour collaborer sur une variété de tâches communes [Warkentin et al., 1997].

Les méthodes d'aide à la décision basées sur l'optimisation tiennent une place importante dans l'aide à la décision [Labbé, 2001]. Ils commencent par une étape de formulation qui consiste à rechercher un modèle dans une forme acceptable. Puis l'étape de solution réfère à la solution algorithmique du modèle. L'analyse consiste à interpréter la solution ou l'ensemble de solutions trouvées. De nombreux modèles sont basés sur la théorie des jeux pour intégrer le coté subjectif des décideurs.

1.2.3 Conclusion

Il existe de nombreux systèmes différents dans le domaine des systèmes d'aide à la décision. Très peu de ces systèmes accompagnent le décideur durant la totalité du processus de décision. Ils sont pour la plupart des systèmes, "intelligents" ou non, cherchant à optimiser des solutions sans être réellement interactifs. Pour la résolution de certains problèmes le processus de décision est lié à la connaissance. Celle-ci tient une place importante et nous verrons dans la suite que les systèmes de connaissance et les SIAD sont liés.

1.3 Gestion de la connaissance - système de connaissances

La définition courante de la connaissance est celle-ci : "La connaissance est le fait de connaître une chose, le fait d'avoir une idée exacte de son sens, de ses caractères, de son fonctionnement" ou "les notions acquises ; ce que l'on a appris d'un sujet" [Hachette 2004] ; il n'existe pas de définition scientifique de la connaissance [Ermine, 2000]. Cependant, il y a souvent confusion entre données, information et connaissance. Certains auteurs donnent leur signification et la correspondance entre ces entités. Sandoval [Sandoval, 1997] hiérarchise les données, les informations et les connaissances. Pour ce dernier, les données se trouvent à la base. Elles servent à produire de l'information qui, à son tour, servira de base pour enrichir les connaissances. Alors que pour Ermine [Ermine, 2000] un système de connaissance est vu comme de l'information qui prend une certaine signification dans un contexte donné. En effet, l'information est la partie visible de la connaissance, mais elle ne crée de la connaissance que si on peut lui rattacher du sens dans un contexte opérationnel.

[Ermine, 2000] trace l'historique de la gestion d'entreprise qui a débuté par les systèmes experts qui ont permis de formaliser la connaissance - elle était alors devenue un objet susceptible de traitement automatique, au même titre que les données d'un problème mathématique ou les informations d'un problème de gestion [Lévine and Pomerol, 1989]. L'ingénierie des connaissances qui consistait à capter l'expertise d'un ou quelques experts pour la "mettre en boîte" dans un système informatisé a donc suivi [Hart, 1988]. Une ingénierie des systèmes d'information, qui est un élément clé des stratégies d'entreprise, se développe avec les NTIC et les moyens dits de "traitements d'information". Cette connaissance structurée dans les systèmes tend à négliger le rôle des opérateurs qui semble alors moins important que l'information traitée. Le patrimoine composé du savoir-faire, de l'expérience et du partage de culture est perdu au fur et à mesure des départs en retraite, des plans sociaux... L'enjeu n'est plus de gérer l'information de l'organisation, mais de gérer son patrimoine de connaissance, d'où la naissance des systèmes de connaissances.

La connaissance est un enjeu économique majeur de demain [Zacklad and Grundstein, 2001a]. Créer, capitaliser et partager son capital de connaissances est une préoccupation de toute organisation performante [Zacklad and Grundstein, 2001b]. Les trois défis de la gestion de la connaissance sont :

- Capitaliser, savoir d'où l'on vient et où l'on est pour mieux savoir où l'on va.
- Partager, passer de l'intelligence individuelle à l'intelligence collective.
- Créer, innover pour survivre.

Liao [Liao, 2003] présente une revue des technologies de la gestion des connaissances (le terme anglophone équivalent est "knowledge management" KM) de 1995 à 2002. L'auteur a analysé un grand nombre d'articles et les a classé suivant sept catégories : cadre de gestion de connaissance, systèmes basés sur la connaissance, interprétation de données ou data mining, technologie de l'information et de la communication, intelligence artificielle/systèmes experts, technologie de base de données et modélisation. Dans la première catégorie, l'auteur distingue, entre autres, la création de connaissance, l'évaluation de la connaissance, la gestion de stratégie et la capitalisation intellectuelle. On retrouve dans la seconde catégorie un ensemble d'articles sur les systèmes experts, les raisonnements à base de règles et les raisonnements à base de cas. Ensuite, les articles connus sur le data mining, qui est une technologie pour la découverte de connaissance dans les bases de données, sont répertoriés. Les technologies de l'information et de la communication permettent des activités de gestion de connaissance pour l'aide à la décision collaborative, le partage d'information, l'apprentissage organisationnel et la mémoire organisationnelle. Les systèmes experts regroupent une base de connaissance, des systèmes basés sur des règles, un moteur d'inférence et un raisonnement à base de cas. Ils peuvent être couplés à d'autres méthodes d'intelligence artificielle tels que les réseaux de neurones, la logique floue, les algorithmes génétiques et les agents intelligents. Les technologies de base de données permettent d'organiser les données en les centralisant et en minimisant les données redondantes. Ensuite, elles permettent d'implémenter les heuristiques de création d'ontologies. Les technologies de modélisation peuvent fournir des méthodes quantitatives pour analyser les données subjectives pour représenter ou acquérir la connaissance humaine avec une logique inductive. Nous pouvons noter qu'un bon nombre de technologies sont communes au domaine des SIAD et au domaine de la gestion des connaissances.

D'après [Ermine, 2003], le contenu du patrimoine est à la fois caché et disséminé dans deux composantes essentielles de l'entreprise : le capital humain et le capital d'information. Après avoir vu l'historique et l'importance de la gestion de la connaissance, nous allons analyser la gestion des connaissances selon ces deux composantes.

1.3.1 Capital humain

Le capital humain signifie pour [Ermine, 2003] que la quintessence de la connaissance de l'entreprise est dans la tête de ses employés. C'est le "lieu de stockage" ultime avant son utilisation opérationnelle. Cette connaissance est tacite et difficilement exprimable. Pour Hart [Hart, 1988], un expert :

- a un savoir réel, i.e. il sait résoudre des problèmes avec un pourcentage acceptable de succès,
- a un savoir efficace, i.e. il résout le problème de manière rapide et efficace, et
- connaît ses limites, i.e. il sait ce qu'il sait.

A partir de ce principe, pour la conception de systèmes experts, Hart veut utiliser les experts pour les informations qu'ils fournissent, leur capacité à résoudre des problèmes, et pour les explications qu'ils donnent.

Pour capitaliser, partager et créer des connaissances provenant ou à destination du capital humain, il faut des méthodes, des techniques et des systèmes adaptés. Dans la suite, un certain nombre de méthodes sont présentées.

1.3.1.1 Modèle de Nonaka

Nonaka et al. [Nonaka and Takeushi, 1995] ont également classifié la connaissance selon deux types : explicite (dimension technique), tacite (dimension cognitive). Ensuite leurs recherches ont évolué vers un processus complet de création de la connaissance tout en se basant sur la typologie précédente [Nonaka et al., 2000]. Les auteurs proposent alors un modèle de création de connaissance compris en trois parties :

- le processus nommé SECI (Socialisation, Externalisation, Combinaison, Internalisation) qui permet de créer de la connaissance en faisant interagir les connaissances tacites et explicites déjà acquises. La socialisation permet de créer de la connaissance tacite à partir de connaissance tacite ; l'externalisation articule la connaissance tacite vers une connaissance explicite ; la combinaison convertit la connaissance explicite en un ensemble plus complexe de connaissance explicite et l'internalisation permet aux individus d'augmenter leur connaissance tacite à partir des connaissances explicites (cf. figure 1.2).
- le partage du contexte de connaissance permet de partager des connaissances et
- les connaissances acquises sont les sorties d'un cycle pour devenir les entrées d'un autre dans le processus de création de connaissance.

Les processus de création de connaissance forment selon eux une spirale reliant ces trois éléments.

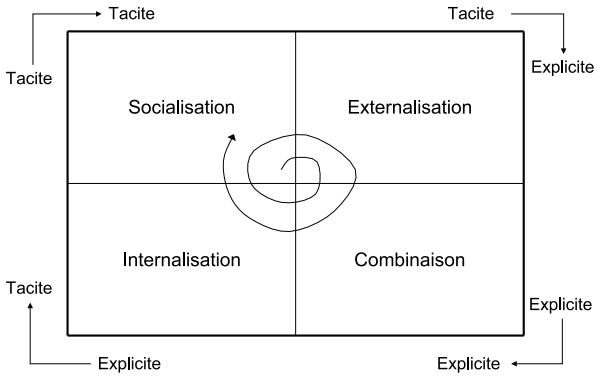


FIGURE 1.2 – Processus SECI [Nonaka et al., 2000]

1.3.1.2 Modèle de Buchanan et ses collègues

Buchanan et ses collègues [Buchanan et al., 1983] distinguent cinq étapes typiques du cycle d'acquisition des connaissances :

- L'identification qui détermine les caractéristiques du problème, les intervenants, les ressources et définit les objectifs ;
- La conceptualisation des connaissances qui consiste à expliciter les concepts relevés lors de l'identification ;
- La formalisation qui permet de concevoir les structures pour organiser la connaissance ;
- L'implémentation qui consiste à coder la connaissance selon un formalisme contraint par l'outil choisi ;
- Les tests dont le but est de valider le système dans des situations réelles. La confrontation des résultats avec l'expert permet d'enrichir ou de modifier les connaissances jusqu'à satisfaction.

1.3.1.3 Méthode MKSM

La démarche de MKSM (Method for Knowledge System Management) est une méthode de gestion des connaissances [Ermine, 1996, Ermine, 2000]. Elle consiste à modéliser les connaissances selon différents points de vue des sources de connaissances de l'entreprise. La méthode MKSM conduit à la réalisation d'un livre de connaissances. La connaissance est structurée suivant trois points de vues (information, sens, contexte) et modélisée suivant cinq modèles :

- Le modèle du domaine participe à la mise en contexte des connaissances du domaine de connaissances concerné en déterminant les phénomènes généraux qui sont à la base du savoir.
- Le modèle d'activité correspond à la mise en contexte des connaissances. Il permet une analyse de l'activité du système qui produit ou utilise les connaissances. Ce modèle permet de replacer les connaissances du domaine (décrites par les phénomènes) dans le cadre d'une utilisation opérationnelle.
- Le modèle des concepts représente l'aspect "statique" de la connaissance. Il traduit la structuration conceptuelle d'un expert, d'une personne habituée à travailler dans un domaine précis.

- Le modèle des tâches décrit la connaissance dynamique. Il représente le savoir-faire utilisé pour réaliser tout ou partie de certaines activités identifiées dans le modèle d'activités.
- Le modèle d'évolution prend en compte l'adaptation des organisations à leur environnement et la problématique de l'innovation.

1.3.1.4 Méthodes de Cooke

Cooke [Cooke, 1994] a répertorié des techniques d'élicitation des connaissances suivant trois familles : (1) Les méthodes directes consistent à observer les experts, procéder à des verbalisations et analyser leur tâche ; (2) le suivi de processus intègre les rapports verbaux, et non-verbaux, l'analyse des protocoles et des décisions et (3) les techniques conceptuelles telles que les méthodes d'élicitation de concepts, de collection de données et d'analyses structurées, sont des techniques plus indirectes demandant moins d'introspection et d'interviews que les premières. Les observations et interviews sont relativement informelles avec une grande partie de méthodes et d'analyses partant de l'intuition de l'élicitateur. Les méthodes de suivi de processus sont mieux spécifiées et les techniques conceptuelles sont assez formelles et bien spécifiées.

1.3.1.5 Ontologies

De manière à capitaliser et partager les connaissances humaines, il faut trouver un moyen de définir les terminologies métier. Ces terminologies doivent être consensuelles, c'est-à-dire que les experts doivent s'accorder quant à la signification des termes, cohérentes, au sens de la consistance logique de l'ensemble des significations ; partageables et réutilisables. Pour cela les ontologies ont été utilisées [Biebow and Szulman, 2000]. La recherche dans le domaine des ontologies est croissante dans le domaine informatique. Par exemple :

- Roche et Rousseau [Roche and Million-Rousseau, 2003] ont proposé un modèle ontologique pour la représentation et la signification des termes. Ce modèle permet de définir une "terminologie ontologique" qui se situe entre la "terminologie textuelle" et la "terminologie conceptuelle". En effet, leur méthodologie de conception des terminologies se situe entre le "mot" et le "concept" en tentant d'unifier les principes linguistiques et épistémologiques.
- Shamsfard et Barforoush [Shamsfard and Barforoush, 2004] ont proposé une approche automatique de construction d'ontologie à partir d'un noyau qui contient les concepts, les relations et les opérateurs primaires.

1.3.1.6 Approche de Kwan et Balasubramanian

Le système de gestion des connaissances, proposé par Kwan et Balasubramanian nommé knowledgeScope [Millie Kwan and Balasubramanian, 2003], permet de gérer la connaissance dans son contexte. La connaissance est catégorisée en trois types : la connaissance du processus, la connaissance du cas et les ressources de la connaissance. La connaissance du processus correspond aux procédures opérationnelles standards du processus comme, par exemple, les techniques de rédaction d'article. La connais-

sance de cas comprend les instances des procédures d'exécution standard dans un contexte particulier, par exemple, la rédaction d'un article particulier pour répondre à un appel à communication dans un congrès. Les ressources de connaissance sont les connaissances techno-scientifiques utilisées dans l'exécution du processus, par exemple, la gestion de la bibliographie. Les informations contextuelles sont cruciales pour le partage et la réutilisation de la connaissance. Nous verrons dans la suite que la réutilisation joue un rôle important dans nos travaux ; c'est pourquoi nous avons choisi de présenter cet exemple.

1.3.1.7 Point de vue de Rolland et ses collègues

Rolland et al. [Rolland et al., 2000] proposent un patron de prise de décision pour guider le processus de développement de connaissance d'une entreprise. Ce patron a pour but de supporter un ingénieur en développement de connaissance d'entreprise durant son processus de prise de décision en étudiant l'impact du changement dans l'organisation. Ce patron est un mécanisme de raisonnement supportant la prise de décision en utilisant une bibliothèque de lignes directrices et un mécanisme d'interprétation, cf. Figure 1.3. Les concepts utilisés pour décrire ce patron sont le concept de contexte, qui est défini comme une paire <situation, intention> ; une situation est constituée d'un ensemble de parts de produit qui sont appropriées à la prise de décision dans le contexte, et l'intention correspond au but. Dans cette approche, c'est l'utilisation des patrons pour la gestion des connaissances qui nous a séduit ; l'approche de conception de SIAD se basera sur une approche par patrons pour intégrer la gestion de la connaissance lors du développement d'un SIAD.

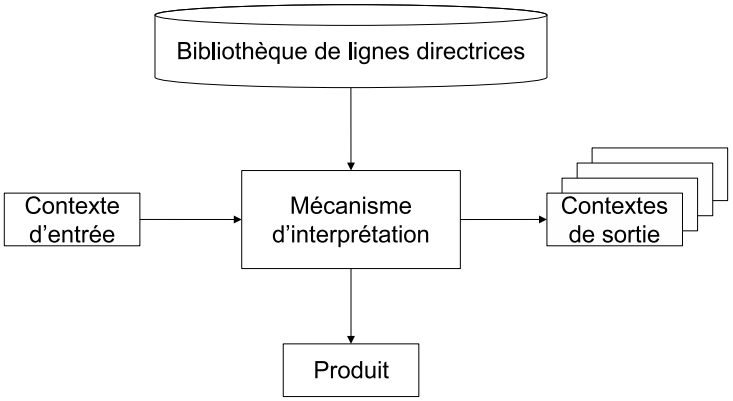


FIGURE 1.3 – Patron de prise de décision en développement de connaissance d'entreprise selon [Rolland et al., 2000]

Nous pouvons souligner aussi que la connaissance est souvent tacite ou explicite. Elle peut passer d'une forme à l'autre en fonction du contexte. Ce contexte accompagne toujours la connaissance. Ce contexte sera intégré à notre approche de développement de SIAD. Après le capital humain, le capital d'information va être abordé.

1.3.2 Capital d'information

Les entreprises ont engrangé, depuis des décennies, des masses d'informations colossales qu'elles ont stockées et qu'elles diffusent par des systèmes d'information. Les employés de l'entreprise utilisent sans cesse ce (parfois gigantesque) système d'information dans leurs activités opérationnelles. Ils acquièrent de l'information, y attachent un sens précis dans leur contexte opérationnel pour les transformer en connaissance utile à leur métier [Ermine, 2003].

L'Extraction de Connaissance à partir de Données (ECD) joue un rôle important pour ce capital. Il comprend la fouille de données, plus connue sous son terme anglophone "data mining", qui est apparu vers le milieu des années 1990 dans le but de valoriser les données stockées dans leur base [Bret et al., 2001]. Les données sont stockées dans des entrepôts nommés "entrepôts de données" (ou "data warehouse"), dans des bases de données distribuées, ou sur Internet ("Web mining").

L'ECD est un processus complexe qui se déroule suivant une suite d'opérations. Des étapes de prétraitements seront suivies du "data mining". Les prétraitements consistent à construire les "datamarts" ou corpus de données spécifiques ainsi qu'à faire le nettoyage des données, le traitement des données manquantes, la sélection d'attributs ou la sélection d'instances. Cette étape est cruciale car la mise au point des traitements dépend des choix des descripteurs et de la connaissance précise de la population. En effet, un choix inapproprié de variables peut faire échouer l'opération [Zighed and Rakotomalala, 2002]. Le data mining peut alors opérer pour aboutir à des connaissances mises sous la forme de modèles. Ils doivent alors être validés. Des post-traitements sont nécessaires pour rendre ces modèles intelligibles soit par un humain soit par une machine. D'après Jambu [Jambu, 1999], le data mining s'inscrit dans un processus qui va de l'information à la décision.

Le Capital d'information comme le capital humain est source de travaux de recherche, de techniques et de méthodes. Ces deux types de connaissance ainsi présentés, nous allons voir quel est leur rôle dans les SIAD.

1.3.3 Gestion de la connaissance et SIAD

Nous avons déjà vu précédemment qu'il existe de nombreux articles qui traitent des SIAD et de la gestion de la connaissance. D'une part dans les travaux de [Rolland et al., 2000], des SIAD sont utilisés pour la gestion de la connaissance. D'autre part dans les travaux de [Gray, 2001], la gestion de la connaissance est utilisée dans les SIAD.

De plus, dans la littérature, on trouve une grande quantité d'articles qui traitent conjointement ces deux domaines, par exemple, dans le domaine financier Zopounidis et ses collègues parlent de SIAD basé sur la connaissance [Zopounidis et al., 1997] ou dans les organisations où Courtney et ses collègues proposent un nouveau paradigne de SIAD basé sur le couplage de la prise de décision et de la gestion des connaissances [Courtney, 2001] tandis que Yim et ses collègues présentent la prise de décision basé sur la connaissance comme base à une approche sur les dynamiques du système [Yim et al., 2004].

Un autre exemple concerne les travaux de Lind qui regarde comment un modèle de décision, en particulier celui proposé par Rasmussen (cf. 1.2.1.2), peut être utilisé pour la conception de Systèmes à Base

de Connaissance (SBC). Les étapes du modèle de décision, une fois formalisées, peuvent être utilisées comme un outil de conception de système à base de connaissance. En conclusion de leurs travaux, le modèle de décision a pu être appliqué à des niveaux fonctionnels d'un système à base de connaissances [Lind, 1991].

On peut alors constater que ces deux domaines sont fortement liés. Nous verrons dans la suite qu'il faudra tenir compte de cette relation dans la conception d'un SIAD.

1.3.4 Conclusion

Nous avons vu que la connaissance d'une entreprise peut être distinguée suivant deux axes : le capital humain et le capital d'information.

L'ensemble des travaux présentés dans cette partie ne correspond qu'à un échantillon de ce qui se fait dans le domaine de la gestion des connaissances. La quantité de travaux montre que ce domaine est d'un grand intérêt pour les entreprises [Prax, 2003], que ce n'est pas un problème facile à résoudre et qu'il est sans cesse en pleine évolution. Ensuite, nous avons, en nous basant sur plusieurs articles représentatifs, vu qu'il existait un lien important entre la gestion de la connaissance et les SIAD.

Dans leurs ouvrages, Maret et Pinon [Maret et al., 1996, Maret and Pinon, 1997] insistent sur l'importance pour une entreprise de maîtriser les savoirs et savoir-faire pour aboutir à leur réutilisation. Ils démontrent l'intérêt du cadre conceptuel pour gérer efficacement les processus de capitalisation et de réutilisation des expériences. La partie suivante traitera de la réutilisation et de son rôle dans les SIAD.

1.4 Rôle de la réutilisation dans les SIAD

Prietro-Diaz et ses collègues [Prieto-Diaz and Freeman, 1987] considèrent deux niveaux de réutilisation : (1) la réutilisation des idées et de la connaissance et (2) la réutilisation de composants et d'artefacts particuliers. Ainsi, ces deux niveaux de réutilisation seront décrits dans cette partie.

1.4.1 Réutilisation des idées et de la connaissance

La réutilisation des idées et de la connaissance fait partie de la gestion de la connaissance que nous avons abordée dans la partie précédente. Nous ne souhaitons pas distinguer la connaissance des compétences ni le savoir du savoir-faire ; nous considérons que les savoir-faire font partie de la connaissance même si celle-ci est spécifique à un processus.

D'un point de vue humain, le processus d'apprentissage consiste à acquérir des connaissances, de manière à pouvoir les réutiliser. On considère généralement que l'objectif de la réutilisation des idées et de la connaissance dans le domaine industriel est d'améliorer l'apprentissage des employés en leur faisant partager leur connaissance ; en principe, ils acquièrent alors plus vite des connaissances et des idées [Hugues, 2000].

Comme nous l'avons précisé, la réutilisation des idées et de la connaissance fait partie de la gestion des connaissances que nous avons abordée précédemment. Nous allons rappeler le principe des patrons

et des ontologies et leur rôle dans la réutilisation.

1.4.1.1 Les ontologies

Il existe de nombreuses définitions du mot "ontologie". Nous retiendrons la définition suivante de Roche et Tutoriel [Roche and Million-Rousseau, 2003] : l'ontologie d'un domaine est un vocabulaire de termes dont les significations se structurent en un système. Elle est définie pour un objectif donné et exprime un point de vue partagé par une communauté. Une ontologie s'exprime dans un langage et repose sur une théorie (sémantique) garante des propriétés de l'ontologie en termes de consensus, cohérence, réutilisation et partage.

Les ontologies permettent de poser des définitions dans un domaine ce qui facilite la compréhension entre les acteurs du domaine. C'est, par conséquent, une base pour la réutilisation. Des travaux dans ce domaine ont déjà été présentés dans le § 1.3.1.5.

1.4.1.2 Les patrons

Cette notion de patron est née d'un architecte [Alexander et al., 1977] pour qui "Chaque patron décrit un problème qui se manifeste constamment dans notre environnement, et décrit l'architecture de la solution à ce problème, d'une façon telle que l'on puisse réutiliser cette solution des millions de fois sans jamais l'adapter deux fois de la même manière". Un patron capitalise un savoir-faire permettant de résoudre un problème récurrent du domaine.

En situation de résolution de problème, la formulation d'une solution implique trois types d'actions : abstraction, identification et description. L'abstraction exprime ce qui est essentiel à la résolution des difficultés et ce qui caractérise la solution. L'identification de la solution est liée au problème qu'elle permet de résoudre et à son contexte. La description fournit un schéma de la solution qui puisse faire office de modèle réutilisable [Nanard, 2002]. Un patron sert à transférer des connaissances en conception mais aussi à produire des solutions lors de son application.

1.4.1.3 La réutilisation dans les SIAD

Un décideur doit prendre une décision pour résoudre un problème en choisissant le meilleur compromis parmi les solutions possibles. Il traite le problème en fonction de ses connaissances. Les SIAD consistent à aider les décideurs tout au long de ce processus, en particulier au moment du choix des études à effectuer ou des outils à utiliser. Ils doivent aider l'utilisateur à réutiliser ses propres connaissances mais aussi celles acquises par l'entreprise (d'autres utilisateurs). La réutilisation des connaissances et des idées dans un SIAD tend à enrichir la connaissance des utilisateurs.

1.4.2 Réutilisation de composants et d'artefacts particuliers

En informatique, l'objectif de la réutilisation est de minimiser les coûts de développement tout en augmentant la fiabilité des systèmes conçus. En effet, réutiliser des systèmes conçus antérieurement suppose qu'ils ont déjà été évalués, maintenus, améliorés. La réutilisation est un enjeu majeur de l'industrie

informatique qui est en pleine évolution. Bien que la réutilisation de code soit un problème ancien et qu'elle évolue avec les technologies, elle n'est toujours pas résolue à ce jour. Il existe cependant un certain nombre de moyens permettant d'avancer vers une réutilisation de code.

1.4.2.1 Moyens de réutilisation

Détienne et ses collègues [Détienne and Burkhardt, 2001] présentent une revue des approches du génie logiciel pour la réutilisation. Les auteurs distinguent la réutilisation par clonage et modification du code, la réutilisation par récupération et particularisation de composants, la réutilisation virtuelle (héritage, composition) et la réutilisation à partir de patrons de conception. Il s'avère que trois approches ont fait évoluer la réutilisation de code : les approches orientées objets, les approches de développement par composants et la notion de patron.

Approches orientées objets

D'après Coulange [Coulange, 1996] "la réutilisation est restée un rêve impossible depuis toujours quand un événement nouveau est arrivé, une nouvelle magie : l'Orienté Objet", cette phrase n'est qu'une introduction de son livre qui lui permet de montrer qu'il n'est pas si facile de réutiliser.

Le principe de base de l'orienté objet est l'utilisation d'objets, associé à la possibilité de mettre en œuvre en particulier trois techniques appelées héritage, polymorphisme et liaison dynamique. L'héritage permet de définir une nouvelle classe objet à partir de la définition d'une ou plusieurs autres classes sans duplication de code. Le polymorphisme est une technique qui complète l'héritage et dont le principe est d'utiliser une méthode d'un objet sans se soucier de son type, les classes spécialisées étant masquées par une interface commune (polymorphisme d'héritage). Par le principe de la liaison dynamique tout objet exécute toujours la méthode qui correspond à son vrai type.

La conclusion du même auteur [Coulange, 1996] est que la force de l'orienté objet par rapport aux autres techniques citées précédemment est de permettre l'adaptation des "composants" aux nouveaux besoins grâce à l'héritage et à la redéfinition de méthodes.

Approches composants

La réutilisation est abordée selon une approche "composant" qui consiste à segmenter, rationaliser, encapsuler et plus généralement modulariser les systèmes d'information [Barbier et al., 2002]. Un des intérêts est que les composants sont utilisés plus souvent, ils sont plus souvent testés et deviennent individuellement tous plus fiables. Les développements à base de composants consistent à assembler ces blocs préfabriqués, paramétrables et indépendants pour concevoir un système logiciel.

Il n'y a pas vraiment de consensus quant à la définition exacte d'un composant. Cependant une des définitions la plus souvent énoncée est celle de Szyperski [Szyperski, 1998] pour qui "un composant est une unité de composition avec des interfaces contractualisées et un contexte de dépendance exprimé explicitement. Un composant peut être déployé indépendamment et il est sujet à composition par une tierce personne".

Les composants sont des entités logicielles possédant des interfaces pour interagir avec leur environnement d'assemblage. Parfois il n'existe qu'une interface ; certains chercheurs, comme Aniorté [Aniorté, 2002], préféreront séparer l'interface de données (Data Input (DI) et Data Output (DO)) de l'interface de contrôle (CI et CO pour Control Input et Control Output) représenté par la Figure 1.4 où IIP correspond aux points d'entrée d'information (Input Information Point), OIP aux points de sortie d'information (Output Information Point), SRP est le point de réception de signal (Signal Reception Point), SEP est le point d'émission de signal (Signal Emission Point), RAP est le point d'accès à la ressource (Ressource Acces Point) et RRP est le point de libération de ressource (Ressource Release Point). D'autres chercheurs mettront une interface de service qui définit les services fournis par le composant et une interface cliente qui représente les collaborations du composant (services acquis) dans la définition proposée par WCOP-96 (Workshop on Component Oriented Programming at ECOOP' 96) et citée dans [Hassine et al., 2002b] (cf. Figure 1.5).

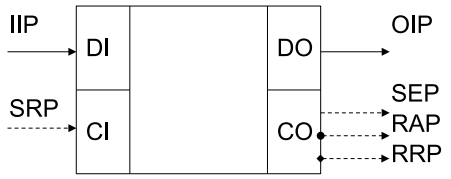


FIGURE 1.4 – Interface d'un composant selon [Aniorté, 2002]

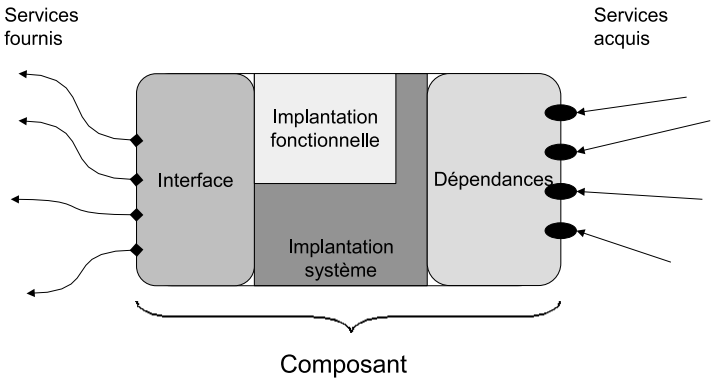


FIGURE 1.5 – Vue générale d'un composant d'après [Hassine et al., 2002a]

[Hassine et al., 2002b] distinguent deux principaux types de composants :

– Les composants métier qui couvrent les besoins du domaine. Ces composants sont divisés en composants verticaux qui sont spécifiques à un domaine (par exemple les modèles d'objets métier financiers) et en composants horizontaux qui représentent des éléments récurrents dans des applications appartenant à différents domaines, comme les services d'authentification ou l'édition de rapport, cf. Figure 1.6.
– Les composants techniques sont des composants non fonctionnels, ils sont utilisés pour la gestion même de la programmation. Par exemple, les composants techniques pour les communications réseaux, les connexions d'unités physiques, pour la gestion des transactions [Herault, 2003].

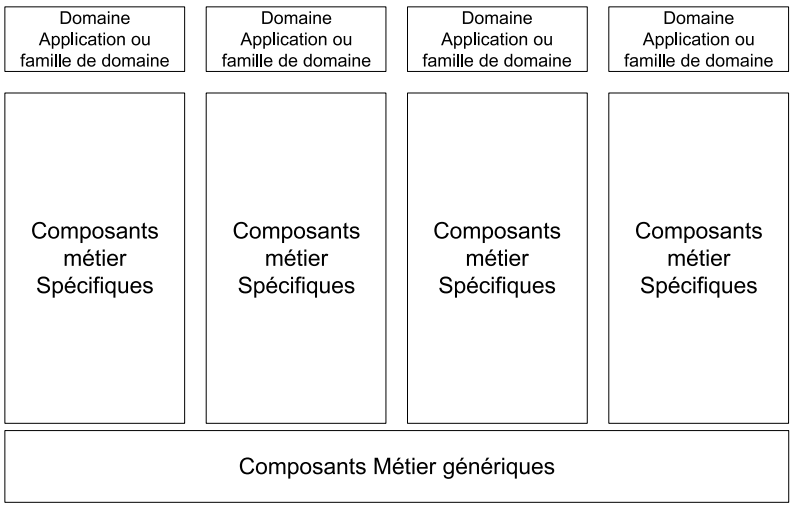


FIGURE 1.6 – Composants métier Horizontaux versus Verticaux

On peut les distinguer également en fonction de leur niveau d'application dans le cycle de développement [Cauvet and Semmak, 1999] :

- Les composants de domaine expriment des connaissances réutilisables dans le développement de tous les systèmes d'un champ d'application. Ces composants sont utilisables et réutilisables dès la phase d'analyse du processus de conception car ils expriment souvent une connaissance de haut niveau, orientée sur la définition de problèmes, de besoins... C'est dans cette catégorie que l'on place les composants métier définis précédemment.
- Les composants de conception sont des structures complexes qui permettent la réutilisation de décisions de conception. Ils peuvent correspondre à des architectures logicielles complètes ou à des modèles génériques ou encore à des fragments d'architectures logicielles fournissant une solution à un problème de conception particulier [Gamma et al., 1994].
- Les composants logiciels sont utilisés dans la production du logiciel. Ils sont principalement utilisés dans la phase de programmation. Ils permettent de réutiliser les fonctionnalités de base de la programmation tels que les tris, les recherches et les comparaisons. Ils englobent les fonctions mathématiques et les structures de données.

Des plateformes particulières, nommées "intergiciels", permettent de concevoir des applications distribuées à base de composants. Certaines sont maintenant bien établies :

- J2EE (Java 2 Enterprise Edition) distribué par Sun [sun microsystems, 2004] propose un modèle de composant nommé EJB (Entreprise Java Bean) et un environnement de programmation. Cette plateforme ne permet l'utilisation que d'un seul langage de programmation : Java, mais permet une passerelle avec CORBA pour intégrer d'autres types de composants.
- CORBA (Common Object Request Broker Architecture) de l'[OMG, 2004] propose un modèle de composant CCM (Corba Component Model) et une plateforme de développements,
- .NET [Microsoft, 2004] propose lui aussi un modèle de composants COM (Component Object Model). La plateforme .NET est basée sur les langages de programmations proposés par Microsoft

(C#, VisualBasic, JavaScript...). Elle propose des bibliothèques de classes étendues (XML.NET pour analyser et manipuler des documents, ASP.NET pour le développements de services Web) le tout dans un environnement de développement Visual Studio.Net et

– Fractal distribué par ObjectWeb [objectweb, 2005] qui est un modèle de composant modulaire et extensible et peut être utilisé avec plusieurs langages de programmation.

Cette approche de développement par composant dans un but de réutilisation est en pleine expansion. Elle permet la réutilisation dans une certaine mesure mais il reste encore de nombreux progrès à réaliser comme la mise au point de normes de modèles de composants pour permettre une meilleure interopérabilité. Il reste encore des progrès à faire concernant les composants techniques et l'intégration des composants métier. De nombreuses recherches dans ce domaine sont menées également sur la recherche et la récupération de composants [Khayati and Giraudin, 2002, Khayati et al., 2003] ; ces problèmes avaient déjà été mis en évidence par Prietro-Diaz et ses collègues [Prieto-Diaz and Freeman, 1987] à la fin des années 80. Ces approches sont les plus répandues désormais dans le monde informatique. Elles sont souvent accompagnées des approches par patrons [Front-conte et al., 1999]. Dans le même sens que ces approches, nos travaux consistent à intégrer la programmation par composants dans une démarche de développement de SIAD.

Approches par patrons

Cette notion est utilisée de manière générale pour la réutilisation, c'est pourquoi elle a déjà été présentée dans la partie précédente, §1.4.1.2. Les patrons (pattern en anglais) ont été adaptés à la réutilisation de code informatique par Gamma [Gamma et al., 1994] qui a proposé un catalogue de patrons de conception basé sur le canevas présenté en Tableau 1.1. Nanard les a adaptés à la conception de documents hypermédia [Nanard, 2002]. Front-Conte et ses collègues utilisent des patrons orientés objet pour intégrer la réutilisation durant les premières phases du cycle de développement d'un logiciel [Front-conte et al., 1999]. Ils distinguent les patrons d'analyse, les patrons de conception et les patrons d'implémentation, selon la description présentée en Figure 1.7. D'après les mêmes auteurs, un patron transforme un problème soit en solution soit en de nouveaux problèmes. Il utilise d'autres patrons lorsque ces derniers peuvent résoudre en partie ces problèmes : "un patron X utilise un patron Y, si une partie des problèmes posés par X peuvent être résolus en partie ou complètement par Y.". Nous verrons dans la suite que nous sommes en accord avec cette règle d'utilisation.

1.4.2.2 Réutilisation dans les SIAD

En conclusion, la réutilisation est indispensable aussi bien lors de développements informatique que pour la gestion des connaissances. Or, nous avons vu que les SIAD font partie des systèmes informatiques ; ils devraient donc être conçus avec un but de réutilisation de code. Nous avons vu également dans la partie portant sur les systèmes de gestion de connaissance que les SIAD étaient très liés à ces systèmes ; ils doivent donc également être conçus dans cette optique de réutilisation de connaissance. La réutilisation permettra aux SIAD d'atteindre ses buts de partage de connaissance tout en pratiquant de la réutilisation de code.

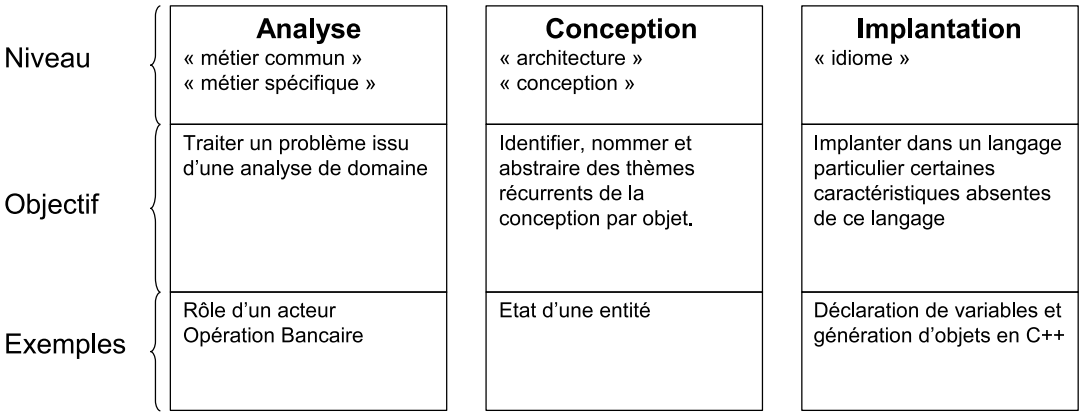


FIGURE 1.7 – Classification élémentaire et exemples de patrons [Front-conte et al., 1999]

Noms de modèle	Le nom du modèle contient succinctement son principe. Un bon nom est vital, car il va faire partie du vocabulaire du concepteur
Intention	C'est une courte déclaration qui répond aux questions suivantes : que fait effectivement le modèle de conception ? Quelle est sa raison d'être et son but ? Quel cas ou quel problème particulier de conception concerne-t-il ?
Alias	Quelques autres noms reconnus du modèle, s'il en existe.
Motivation	C'est un scénario qui illustre un cas de conception, et qui montre comment la structure de classe et d'objet du modèle contribuent à la solution de ce cas. Ce scénario aide à comprendre les descriptions plus abstraites du modèle dans les sections qui suivent.
Indications d'utilisation	Quels sont les cas qui justifient l'utilisation du modèle de conception ? Quelles situations de conception peu satisfaisantes peuvent tirer avantage de l'utilisation du modèle ? Comment reconnaître ces situations ?
Structure	C'est une représentation des classes du modèle, qui utilise une notation issue de la méthode OMT. Nous utilisons également les diagrammes d'interaction pour représenter les séquences de requêtes et la coopération entre objets.
Constituants	Les classes et/ou les objets intervenant dans le modèle de conception, avec leurs responsabilités.
Collaborations	Comment les constituants collaborent-ils pour assumer leurs responsabilités ?
Conséquences	Comment le modèle de conception assume-t-il ses objectifs ? Quels sont les compromis qu'implique son utilisation et quel en est l'impact ? Quelles parties de la structure d'un système permet-il de modifier indépendamment ?
Implémentation	De quels pièges, astuces, ou techniques faut-il être averti lors de l'implémentation du modèle, y a-t-il des solutions typiques du langage utilisé ?
Exemples de code	Ce sont des extraits de programmes, qui utilisent la façon qu'il convient d'employer pour développer le modèle en langage C++ ou Smalltalk.
Utilisations remarquables	Exemples de modèles appartenants à des systèmes existants. Il s'agit de présenter au moins deux exemples issus de domaines différents.
Modèles apparentés	Quels modèles de conception sont en relation étroite avec celui traité ? Quelles sont les différences importantes ? Avec quels autres modèles peut-il être employé ?

TABLEAU 1.1 – Canevas proposé par Gamma pour la description des modèles de conception

1.4.3 Conclusion

L'enjeu économique induit par la réutilisation, qu'elle soit des connaissances ou de développements informatiques, est important. Les SIAD sont des systèmes à la fois basés sur la connaissance et informatiques. Ils devraient donc être conçus de manière à intégrer ces deux types de réutilisation. Nous verrons dans la suite du mémoire que cet aspect tiendra une place importante. L'approche que nous proposerons sera basée sur les patrons de manière à respecter cette nécessité de réutilisation. Nos travaux seront principalement centrés sur les composants métier qui nous permettront de faire de la réutilisation logicielle tout en permettant une réutilisation de connaissance métier.

Depuis le début de ce chapitre, un état de l'art concernant les systèmes d'aide à la décision, la gestion de connaissance et la réutilisation a été proposé. Le contexte industriel dans lequel ces travaux ont été effectués est abordé dans la section suivante. Il concerne la problématique des décisions qui doivent être prises concernant les investissements dans les infrastructures ferroviaire.

1.5 Contexte applicatif : investissement transport

Ce travail se situe dans le cadre applicatif de l'aide à l'investissement dans les infrastructures ferroviaires. Les objectifs poursuivis par la loi 97-135 du 13 février 1997 se sont traduits par la création de Réseau Ferré de France et par la mise en place de l'expérimentation de la régionalisation des services de voyageurs.

Avec la création de RFF [RFF, 2005] :

- La SNCF est libérée de la charge de l'infrastructure ferroviaire.
- La propriété du domaine public ferroviaire est transférée pour l'essentiel à RFF.
- Ce transfert de propriété modifie les relations entre infrastructure (gérée par RFF) et transport (exploité par la SNCF) : la SNCF est désormais cliente de RFF pour son accès au réseau et lui doit à ce titre des redevances.

A sa création, RFF ne possède aucune capitalisation de connaissance ni aucune mémoire d'entreprise. Dans un premier temps des experts de la SNCF en nombre restreint ont été mis à disposition de RFF pour apporter leur connaissance (de manière générale, donc théorique et pratique). Il fallait que cette connaissance soit partagée et utilisée pour former l'ensemble du personnel de RFF, souvent novice au monde ferroviaire.

1.5.1 Problématique

De par sa nouvelle fonction, RFF est devenu maître d'ouvrage des projets d'investissements pour les infrastructures ferroviaires. L'investissement se présente souvent sous la forme d'un objectif de circulation accompagné de données génériques d'infrastructure (cf. Figure 1.8). Le rôle de RFF est de s'assurer que la modification d'infrastructure soit nécessaire pour satisfaire la demande en circulation. Par exemple, l'augmentation des dessertes voyageurs entre le littoral et les villes de Lille et Arras et le développement prévisible du trafic fret généré par le complexe de Dunkerque débouchent sur un projet

de modification du plan de voies de la gare d'Hazebrouck.

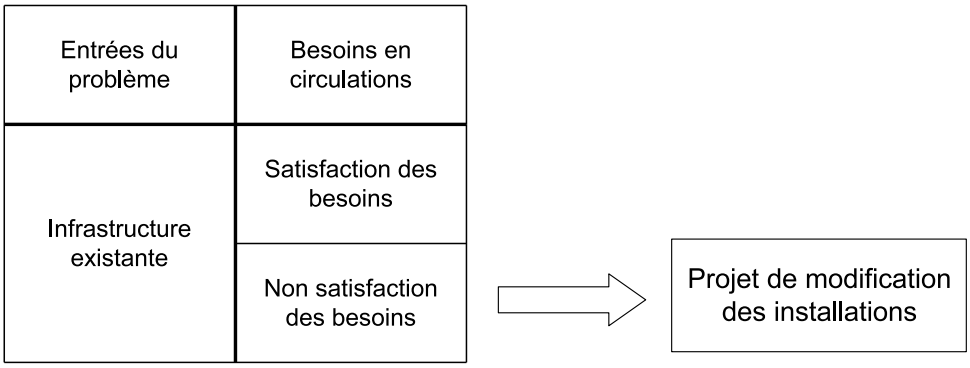


FIGURE 1.8 – Problème d'investissement

Un projet passe par trois phases avant de se concrétiser (cf. Figure 1.9) :

– La phase préliminaire justifie les travaux demandés. Le service d'exploitation est chargé de présenter les coûts associés aux fonctionnalités.
– La phase d'avant-projet consiste à mener des études plus détaillées que lors de la phase précédente. Des études socio-économiques sont réalisées pour, entre autre, calculer le bénéfice de l'investissement et la part pour laquelle RFF participera au financement. Une commission constituée du directeur général, du comité d'investissement et du conseil d'administration confirme alors l'engagement de RFF.
– La phase finale est la phase de réalisation du projet.

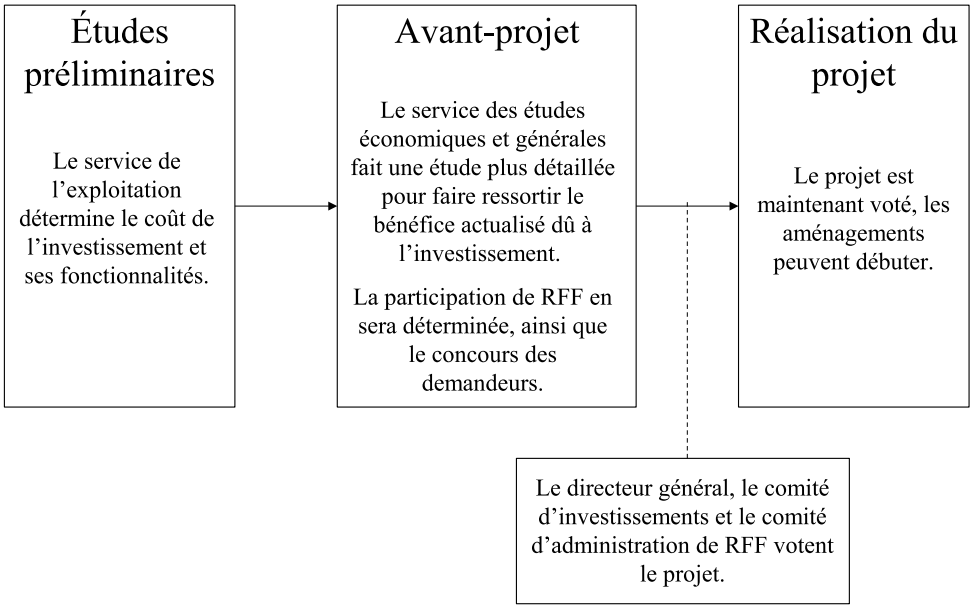


FIGURE 1.9 – Phases de réalisation d'un projet à RFF

Pour compléter notre exemple de modification du plan de voie de la gare d'Hazebrouck, les travaux à réaliser, estimés à 38,11 M€, consistent en :

- l'aménagement des voies
  - Suppression de la bifurcation de Haute Loge.
  - Création de deux voies de circulation pour séparer les courants de trafic Lille-Dunkerque et Arras-Calais.
  - Dédoublement des jonctions vers Calais afin de permettre des simultanéités de mouvement.
  - Maintien à la bifurcation d'Hazebrouck du raccordement à niveau des voies de Béthune aux voies de Lille-Calais.
- La création d'un poste d'aiguillage unique de technologie moderne pour visualiser l'ensemble des circulations dans toute la zone du nœud ferroviaire.

Pour mener à bien ces différentes phases, RFF a besoin d'outils de mesure. Pour des projets de petite ampleur, les délais entre chacune des étapes peuvent être courts ; des outils de mesure requérants une grande précision peuvent être utilisés. *A contrario*, des projets de plus grande ampleur ou à horizon plus lointain (20, 30 ans) ne permettent pas d'utiliser dès le départ des outils trop précis. La situation exacte de l'infrastructure et les demandes en circulations ne sont pas établies, seules des estimations peuvent être utilisées. Dans ce cas, RFF a également besoin d'outils ne demandant pas de données fines pour évaluer ses investissements. Dans le cadre de nos travaux, nous avons étudié les besoins de deux départements de RFF les plus concernés par les investissements : le département d'études socio-économiques et le département de l'exploitation. Le constat fut que ces deux départements n'utilisent pas tout à fait les mêmes données et n'ont pas les mêmes objectifs mais ont un but commun, celui de réaliser le meilleur investissement possible. Il manque d'outils communs pour la simulation et l'étude de la rentabilité économique.

La prise de décision se situe dans un processus axé sur la connaissance. Le décideur doit connaître les avantages et inconvénients de sa décision. La décision se prend à la suite d'études au cours desquelles des solutions différentes seront analysées et comparées. C'est pourquoi des outils pour tester dans des délais raisonnables différentes solutions sont nécessaires. A ce jour, des outils existent mais seuls les experts peuvent les comprendre ; ils sont souvent spécifiques à un problème. La difficulté d'un décideur est de connaître les outils existants et de choisir les outils qui sont adaptés à son problème, et même plus précisément à son processus décisionnel.

Ainsi, pour évaluer les projets d'investissement, RFF a besoin d'outils pour mesurer les conséquences des choix selon chaque critère jugé important concernant l'investissement demandé. Des outils de comparaison seront utiles pour rassembler les différentes mesures faites par les outils de mesure et fourniront une vision globale au décideur pour faciliter sa prise de décision. RFF a aussi besoin d'un moyen de répertorier et proposer ces outils à l'utilisateur (décideur) pour le guider dans son choix d'outils tout au long du processus afin de lui apporter une aide méthodologique. En résumé, RFF a besoin d'un SIAD. La première difficulté sera de répertorier les outils utiles en fonction du type de problème en investissement. Pour mieux comprendre et analyser les besoins, les systèmes existants dans le domaine de l'exploitation ferroviaire en relation avec l'étude des infrastructures vont être présentés.

1.5.2 Systèmes d'aide existants

Il existe d'ores et déjà un certain nombre de systèmes pour l'analyse du réseau ferroviaire. Quelques uns parmi les plus représentatifs seront présentés. Il est difficile de les détailler tous, seule une description succincte sera fournie. Nous commencerons par présenter les outils concernant l'évaluation de la capacité ferroviaire qui est un critère majeur pour un investissement. Ces outils sont souvent liés aux outils de conception horaire ; ils seront donc présentés conjointement. Nous passerons ensuite en revue les outils d'évaluation de grille horaire [2]. Ils correspondent à un autre critère important de décision notamment dans les investissements d'amélioration de la robustesse[3]. Ils se basent sur des méthodes de simulation ferroviaire utilisés de manières diverses et principalement pour confronter une grille horaire et une infrastructure.

1.5.2.1 Systèmes d'aide à l'évaluation de la capacité ou à la conception de grille horaire

Définition

Il existe plusieurs définitions de la capacité ferroviaire. Nous allons donner une définition générale qui permet au lecteur d'appréhender le sujet. La capacité ferroviaire correspond au nombre de sillons [4] définis que l'on peut placer sur une infrastructure ferroviaire durant un intervalle de temps donné. Par exemple, dans le cas d'une ligne permettant un espacement entre deux trains de 3 minutes (ou espacement à 3 minutes), la capacité de la ligne est de 20 sillons identiques par heure (cf. Figure 1.10). Dans ce cas simple, la capacité est facile à calculer mais elle ne représente pas la réalité car il y a rarement un seul type de sillon circulant sur une voie. Il faut en effet prendre en compte d'autres paramètres comme : l'hétérogénéité et l'ordonnancement des sillons, les performances du matériel, les cisaillements, les croisements, et les aléas dans l'exécution du service.

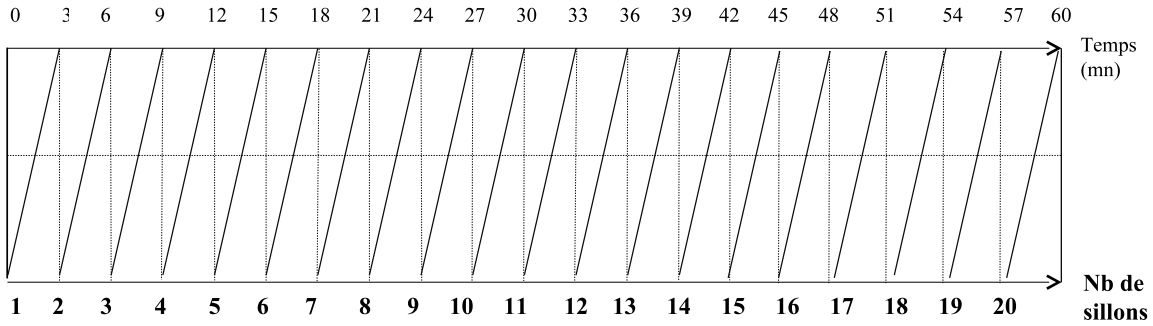


FIGURE 1.10 – Evaluation de la capacité d'une infrastructure

[2]Une grille horaire intègre l'ensemble des circulations prévues sur une infrastructure donnée.

[3]La robustesse correspond à la capacité qu'a une grille horaire à se rétablir en cas d'incidents.

[4]Un sillon est un créneau horaire permettant la circulation d'un train de type donné (TER, FRET, GL) sur un itinéraire donné (Valenciennes-hirson).

Méthodes d'évaluation de la capacité

Il existe diverses méthodes pour calculer la capacité ou plus exactement pour l'évaluer. Une étude menée sur la saturation des lignes au sein de RFF [RFF, 2000] a permis de classer ces méthodes en quatre catégories comme présentées par la suite. Nous distinguerons également à la fin de cette partie les outils utilisables uniquement pour une partie spécifique du réseau tels qu'une ligne ou un nœud et ceux valables sur le réseau global.

a. Méthodes basées sur des formules analytiques

Ces méthodes se basent sur une évaluation de la moyenne des temps minimums de successions Ts des différents trains. Ces formules se distinguent entre elles par les différentes méthodes d'approche de Ts et par les différentes marges adoptées en fonction du niveau de qualité souhaité (par exemple l'augmentation de la marge de détente, destinée à éviter les retards en cascade, implique souvent une meilleure qualité de l'exploitation). Les résultats fournis par ces méthodes sont théoriques et imposent des contraintes comme le regroupement de sillons de même type. C'est pourquoi elles sont de moins en moins utilisées. Seule la formule proposée par l'UIC (Union Internationale des Chemins de fer) qui est la plus connue sera présentée ici. Elle constitue la base des autres formules que l'on trouve dans la littérature.

La formule analytique que nous désirons présenter dans un but d'illustration est celle de la fiche UIC de 1979 qui donne la capacité d'une ligne par calcul :

$$C = \frac{T_{ref}}{T_{fin}+T_r+T_{zu}}$$

où :

- T_{ref} est la période de référence définie par l'utilisateur,
- T_{fm} est la durée moyenne de succession (moyenne des temps minimaux de succession nécessaires entre deux trains),
- T_r est la marge de détente (pour éviter les retards en cascade et pour tenir compte du niveau de qualité) ; elle prend en compte le coefficient de souplesse k. $T_r = (1-k)/k * T_{fm}$ avec k=75% si Tréf correspond à une période de plusieurs heures,
- T_{zu} est le temps supplémentaire (perturbations aux extrémités des sections du tronçon) ; $T_{zu} = 0.25*$ nombre de sections de ligne du tronçon considéré. La Figure 1.11 présente un découpage de ligne en tronçons. Les principaux éléments d'une ligne (ici Bordeaux-Hendaye) sont présentés en Figure 1.11 tels que les voies, les points remarquables. Un zoom sur un tronçon permet d'intégrer la notion de section de ligne utilisée dans la formule.

La difficulté de cette méthode réside dans la détermination de Tfm. Tfm est obtenue en rangeant les trains dans p catégories compte tenu de leur temps de parcours (p=4 au maximum). Ces catégories sont généralement associées au type de transport (fret, banlieue, régional, grande vitesse). Pour chaque couple de catégorie de trains (i,j), on définit le temps minimum de succession Tsij, appelé également temps d'espacement, qui dépend de la longueur entre deux signaux d'espacement [5], des catégories données (i,j)

[5]Pour des raisons de sécurité, deux trains successifs doivent toujours être séparés par une distance permettant l'arrêt du second train sans percuter le premier. Pour cela, des signaux jalonnent les voies pour gérer cet espacement.

et du sens du parcours. Dans le cas où le graphique est connu, on déduit le nombre de cas où un train de classe i précède un train de classe j et on en déduit Tfm. Dans l'autre cas, on connaît le nombre de trains de classe i et on procède à une répartition statistique des cas de successions pour en déduire Tfm.

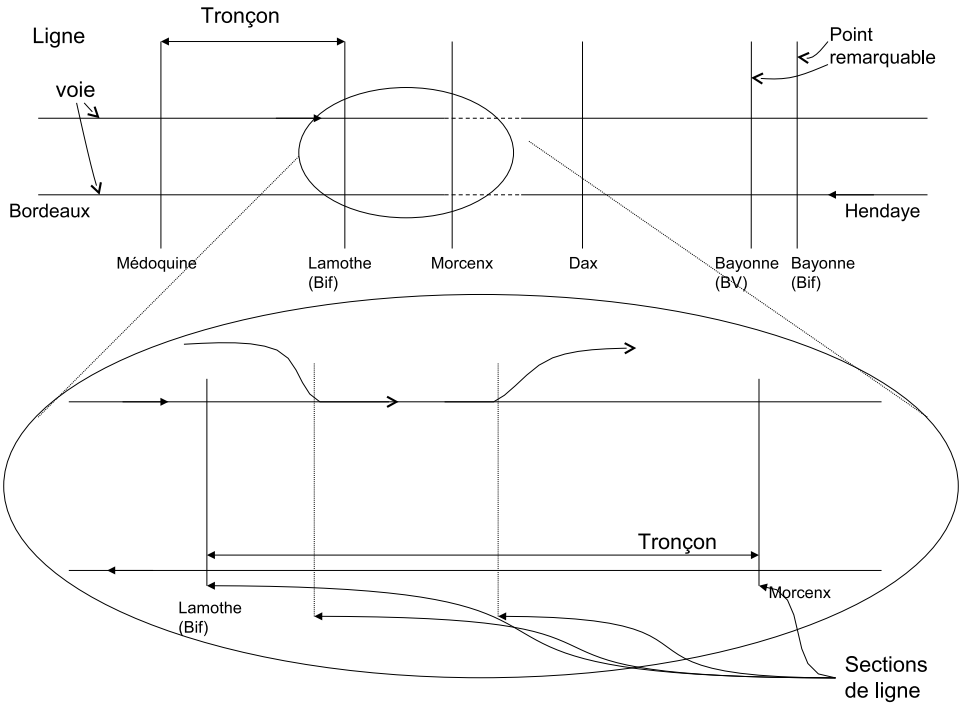


FIGURE 1.11 – Schéma ferroviaire

En raison de toutes les approximations et les pondérations arbitraires, cette méthode manque de précision. Comme nous l'avons dit la capacité représente le nombre de trains que l'on peut faire circuler sur la voie. Elle dépend du type de train qui est donc vu comme une unité. La méthode présentée ne permet pas de savoir quels types de trains peuvent être insérés, ce qui signifie que l'on obtient la valeur de la capacité sans posséder son unité.

b. Méthodes basées sur des formules probabilistes

Ces méthodes peuvent s'utiliser lorsque l'on ne connaît pas précisément la grille horaire. Elles se basent ainsi sur une évaluation probabiliste de la répartition des trains et font des hypothèses sur la distribution de la circulation. De la même manière que pour les formules analytiques, ces formules imposent des hypothèses trop contraignantes. Nous n'en développerons qu'une dans un but d'illustration.

Une des formules probabilistes est la formule DB (Deutsche Bahn) proposée par les Chemins de fer de l'Allemagne Fédérale au début des années 1970. Elle s'appuie sur l'hypothèse que la répartition du nombre de trains se présentant en un temps donné est distribuée suivant une loi de Poisson.

$$C = \frac{T_{circul}}{3 * T_{fm}}$$

où :

- T_{circul} est la partie du temps de référence T_{ref} pendant laquelle les trains peuvent effectivement circuler,
- T_{fm} est l'espacement moyen minimum entre deux trains,
- et "3" est le résultat de calculs qui sont basés sur la théorie de Poisson et qui dépendent fortement des hypothèses.

Cette méthode est facile à appliquer mais les hypothèses sont encore une fois trop réductrices et le résultat ne reflétera pas la réalité ferroviaire, selon les experts avec lesquels nous avons travaillé tout au long de ce projet.

c. Méthodes de construction d'horaires

Ces méthodes, partant d'une grille horaire donnée, utilisent des théories pour élaborer la grille la plus dense possible : cela correspond ainsi à la situation la plus saturée. Dans la partie suivante, nous distinguerons les systèmes qui font de la construction d'horaire pour évaluer la capacité (en utilisant la saturation) de ceux qui font de la construction d'horaire uniquement pour l'exploitation. Les systèmes existants les plus connus du monde ferroviaire ont été développés soit par des gestionnaires ou exploitants des réseaux ferrés comme la SNCF, CFF (Chemin de Fer Fédéraux Suisse), soit des universités (IVE (Institut für Verkehrswesen Eisenbahnbau und -betrieb de l'Université de Hanovre), EPFL (Ecole Polytechnique Fédérale de Lausanne), Université Technologique de Delft), soit des cabinets d'études (SYSTRA, SMA and associates. . .).

- CAPRES (système d'aide à l'analyse de la CAPacité des RESeaux ferroviaires) a été développé par les Chemin de Fer Fédéraux Suisse (CFF). Se basant sur les travaux d'Hachemane, CAPRES a pour but de fournir un système d'aide à l'évaluation de la capacité d'infrastructures complexes par construction d'horaires [Hachemane, 1997].
- DONS (Design Of Network Schedules) est un système de conception de grille horaire conçu par Railned (Pays-bas). Son objectif est d'aider à planifier une ou plusieurs grille(s) horaire(s) pour l'ensemble d'un réseau afin d'évaluer la capacité de différents projets d'aménagement à absorber un trafic prévu. Il est également pourvu d'un module qui permet de valider les circulations programmées lors du passage dans un nœud complexe [Zwaneveld et al., 2001].
- Le logiciel DEMIURGE, développé par la recherche SNCF [Labouisse and Djellab, 2001], permet d'évaluer la capacité d'un réseau à absorber de nouveaux trafics, d'optimiser les grilles horaires actuelles et futures et de calculer la capacité résiduelle d'une grille horaire par saturation de grille horaire.
- VIRIATO [SMA, 1999, SMA, 2002], développé par SMA et associates, permet la conception des grilles horaires pour un réseau particulier en intégrant le cadencement et en cherchant à optimiser les circulations de la grille horaire. Il se base sur un graphe représentant schématiquement le réseau et les horaires.
- RAILSYS [Radtke and Bendefeldt, 2001] développé par IVE (Institut für Verkehrswesen Eisenbahnbau und betrieb) , permet l'analyse, la planification et l'optimisation des grilles horaires. Il utilise une description "microscopique" du monde réel pour réduire le risque de mauvaises décisions basées sur une planification inexacte. Le calcul de la capacité est basé sur un processus itératif

qui consiste à planifier la grille horaire et la valider par simulation ; ce processus se fait jusqu'à obtention d'une mauvaise grille horaire ce qui signifie qu'à l'itération précédente la grille était à sa capacité maximale.

- Le système INFRAFER (INFRAstucture FERroviaire) est issu d'un projet PREDIT réunissant CORYS TESS, RFF et le LAMIH [Lepreux et al., 2001b, Lepreux et al., 2001a, Paulhac et al., 2002]. Ce logiciel intègre à la fois une simulation du trafic et un module de calcul de capacité. La simulation se base sur des coefficients d'accélération et de décélération des mobiles et sur une description fine du réseau alors que le module de calcul de capacité ne demande qu'une description détaillée de l'infrastructure et une grille horaire de base. Le module de calcul de capacité insère des sillons définis jusqu'à saturation pour obtenir la capacité résiduelle. Les sillons peuvent être définis : (1) sur la base de sillons existant, (2) sur la moyenne des sillons existant dans la grille horaire de base ou (3) peuvent être configurés entièrement par l'utilisateur.
- Delorme [Delorme, 2003] propose une modélisation linéaire du problème de capacité ferroviaire à l'échelle d'un nœud complexe ainsi que des algorithmes permettant de le résoudre dans le cadre d'un projet nommé RECIFE (REcherche sur la Capacité des Infrastructures FErroviaires). Ce projet a été mené par l'INRETS, le LAMIH et la SNCF ; il a visé à développer des outils d'aide à la décision pour l'étude de la capacité d'infrastructures ferroviaires à l'échelle d'une gare ou d'un nœud.

Nous avons rassemblé dans cette partie les systèmes de conception de grille horaire et de calcul de capacité car ils sont généralement indissociables. En effet, soit leur but principal est la conception de grille horaire et l'évaluation de la capacité est "faisable" ou, à l'inverse, ils sont prévus pour évaluer la capacité et peuvent être utilisés pour concevoir une grille horaire.

d. Méthodes de simulation

Ces méthodes informatiques qui ne font pas de calculs théoriques mais qui simulent la circulation des différents trains connus et les différents événements survenant sur le réseau. Les données issues de ces simulations telles que les temps de retour à voie libre permettent d'évaluer la capacité. On peut se rendre compte *de visu* du niveau de qualité et de robustesse d'une grille ; c'est pourquoi ces méthodes permettent également d'évaluer une grille horaire et seront présentés avec les systèmes d'évaluation présentés dans la section suivante.

1.5.2.2 Système d'aide à l'évaluation de la grille horaire et de simulation ferroviaire

Définition

L'évaluation de la grille horaire consiste à examiner comment "réagit" une grille horaire aux perturbations[6] ; on parle alors de robustesse de grille horaire. Cette évaluation peut par exemple être utilisée pour justifier d'un investissement qui aurait pour but d'augmenter la régularité[7] des circulations.

[6] Les perturbations peuvent être provoquées par des pannes, incidents voyageurs, accidents et autres causes non prévisibles.
[7] Capacité des circulations à arriver à l'heure prévue.

Méthodes d'évaluation de grille horaire basées sur la simulation ferroviaire

Nous avons regroupé les outils d'évaluation de grille horaire avec les simulateurs ferroviaires car ils sont le moyen le plus fréquemment utilisé pour évaluer une grille horaire. La simulation permet le plus souvent de vérifier la faisabilité d'une grille horaire ou d'analyser la robustesse d'une grille à supporter des perturbations. La simulation consiste à reproduire sur ordinateur le comportement d'un système de manière réaliste. Cést un moyen d'évaluer une situation avant sa mise en œuvre dans la réalité. Elle a également un aspect pédagogique intéressant. Il existe un certain nombre de simulateurs dans le domaine ferroviaire ; nous ne citerons que les plus connus :

- Le logiciel RAILSIM / Network Simulator développé par SYSTRA [SYSTRA Consulting, 2005] permet une modélisation précise de nombreux trains opérant sur le réseau ferroviaire et de nombreux types d'infrastructure. De plus, la simulation utilisée peut utiliser des méthodes stochastiques pour simuler la variabilité des scénarios ferroviaires. Cette simulation a un degré de précision élevé ce qui implique un long temps de saisie des informations nécessaires.
- Le logiciel SIMONE (SImulation MOdel for NEtwork) [Middelkoop and Bouwman, 2001] a été développé par Railned. Ce logiciel de simulation se base sur les temps de parcours déduits de la grille horaire, sans prendre en compte les temps de ralentissement des circulations, ce qui induit un manque de précision. Cependant, il permet d'insérer des retards pour évaluer la robustesse de la grille.
- Le logiciel SISYFE (SImulateur du SYstème FErroviaire) [Fontaine and Gauyacq, 2001] développé par la SNCF est une boîte à outils pour modéliser les différents éléments d'un système ferroviaire, tels que la signalisation, les types de train et les catégories de conducteur. Ces éléments sont utilisés pour simuler le fonctionnement du réseau ferroviaire.
- Le logiciel SAMURAIL [CORYS TESS, 2004], commercialisé par Corys TESS, est le produit successeur d'INFRAFER présenté dans la section précédente [Paulhac et al., 2002]. Cet outil est destiné à l'étude détaillée de la configuration de signalisation d'un réseau ferroviaire et des différents scénarios de circulations. SAMURAIL est innovant par la rapidité de la configuration de l'infrastructure même pour les infrastructures d'une complexité importante.
- Le logiciel FASTA développé par LITEP/EPFL [EPFL, 1991] simule la circulation des trains sur l'ensemble du réseau national suisse. Il permet d'analyser les retards et d'évaluer la stabilité des grilles horaires pour les réseaux ferroviaires complexes.
- DONS-Simulator [Hoogstriema and Teunisse, 1998] développé par Railned a pour but d'évaluer la robustesse de grilles horaires générées par DONS (logiciel de conception d'horaire, cf. 1.5.2.1). Ce logiciel permet d'étudier les effets des petites perturbations sur la ponctualité des circulations dans le réseau allemand.
- OPENTRACK est développé par ETH Zurich [ivt, 2005] depuis le milieu des années 90. Il prend en compte les caractéristiques du réseau, des locomotives et des grilles horaires. Il calcule le mouvement de tous les trains à la seconde près et reproduit le comportement exact des installations de sécurité. Cet outil adopte donc une très forte précision.

Nous pouvons remarquer que les simulateurs cités ont chacun leurs particularités : soit ils prennent en compte une description détaillée de l'infrastructure, soit une description plus macroscopique. Ils concernent uniquement les nœuds ferroviaires, les lignes, l'ensemble d'une ligne, ou l'ensemble d'un réseau. Ils concernent parfois un réseau particulier, comme FASTA qui n'est configuré que pour le réseau helvétique. Ils sont nombreux et disparates. L'inconvénient principal des simulateurs est le degré de précision. Pour obtenir des résultats satisfaisants, c'est-à-dire correspondant à la réalité, les simulateurs doivent prendre en compte de nombreux paramètres sur les infrastructures (signalisations, intersections…), les trains (vitesse, tonnage, coefficient d'accélération de décélération) et même parfois sur les comportements de conduite (freinage anticipé ou non, vitesse proche des limitations ou plus basse…). Dans les autres cas, la saisie des données est plus rapide mais la simulation n'en sera que moins réaliste.

Méthodes d'évaluation de grille horaire non basées sur la simulation ferroviaire

L'évaluation d'une grille horaire peut être faite par évaluation de la capacité, par simulation, mais aussi suivant d'autres méthodes comme le permet le système PETER (Performance Evaluation of Timed Events in Railways) [Goverde and Odijk, 2002], proposé par l'université de Delft (TUDelft), qui utilise une modélisation algébrique basée sur des événements discrets pour évaluer la robustesse d'une grille horaire. Il intervient principalement sur les paramètres de grille horaire.

Notons enfin qu'il existe un outil pour l'évaluation du réseau ferroviaire : NEMO (Network Evaluation MOdel) développé par IVE (Université de Hanovre) qui est un outil de planification stratégique pour l'évaluation de l'infrastructure [Kettner and Sewcyk, 2001]. Il utilise Railsys (cf. 1.5.2.1) pour la génération d'horaire et le calcul des temps de parcours. A partir des données de ce dernier, il détecte les points noirs de l'infrastructure entraînant des ralentissements, diminuant la capacité ou la robustesse de la ligne.

Le Tableau 1.2 récapitule les principales différences entre les outils selon les paramètres (infrastructure et/ou circulation), l'objectif (évaluer la capacité ou la robustesse) avec la (les) méthodes utilisées et le degré de précision. Soit le degré de précision est :

- Elevé ; ce qui implique une saisie longue des caractéristiques de l'infrastructure (voies, mais aussi signaux d'espacement entre les trains sachant, qu'en fonction du type de signalisation il peut y en avoir tous les deux kilomètres et qu'une ligne peut faire 300 km) pour obtenir un niveau microscopique,
- Intermédiaire ; ce qui signifie que la saisie ne comprend qu'un certain nombre d'éléments de précision ; le degré de précision sera plus faible qu'avec un niveau élevé.
- Bas ; il permet une saisie rapide des éléments essentiels de l'infrastructure, s'avérant suffisante pour évaluer la capacité ; ce niveau n'apparaît que dans le système INFRAFER. Il n'est pas compatible avec les méthodes de conception d'horaires qui demande plus de précision.

1.5.3 Conclusion sur la thématique liée au domaine ferroviaire

Une infrastructure ferroviaire est un investissement très important lors de sa mise en œuvre. Elle doit être capable de supporter les accroissements naturels du trafic fret et l'augmentation de l'offre voyageur sur plusieurs décennies. Les entreprises ferroviaires ont besoin d'outils d'analyse pour prendre des décisions en investissement. Deux types de systèmes sont nécessaires pour l'évaluation d'investissement : (1) les systèmes de conception d'horaire qui permettent aussi l'évaluation de la capacité, et (2) les systèmes d'évaluation de grille horaire dont les simulateurs font souvent partie. Certains outils ne se contentent pas d'évaluer les grilles horaires par rapport à l'infrastructure mais permettent de modifier les deux paramètres (infrastructure et circulation) pour trouver une adéquation entre les circulations voulues et les infrastructures existantes ou améliorées. Le Tableau 1.2 permet de comparer l'ensemble des outils par rapport aux critères paramétrables (infrastructure ou circulation) et de synthétiser les objectifs pour lesquels ils peuvent être utilisés. Ces outils sont disparates, parfois restant dans un milieu fermé. De plus ces outils ne sont pas interconnectables, ce qui entraîne un manque pour les entreprises gestionnaires qui ne disposent pas d'un ensemble cohérent d'aides dans leur processus de décision.

1.6 Conclusion

Nous avons dans ce chapitre présenté la problématique de nos travaux de recherche. Nous avons commencé par une présentation globale des systèmes d'aide à la décision, pour en souligner l'historique et en montrer certaines particularités. Nous axons nos recherches principalement sur les SIAD qui sont des systèmes complexes car interactifs, axés sur le processus de décision de l'utilisateur, basés sur les connaissances de l'entreprise (et de ses experts). C'est pourquoi nous avons ensuite rappelé les notions indispensables sur la gestion des connaissance, d'une part liées aux connaissances des experts du domaine et d'autre part liées aux connaissances acquises par l'entreprise concernant à la fois la gestion de projet et la gestion des acquis (tels que des systèmes d'analyse). Dans une troisième partie, nous avons rappelé le but de réutilisation aussi bien en terme d'idées et de connaissance qu'en terme de code logiciel. La notion de patron semble indispensable comme moyen permettant une meilleure réutilisation et ce quel que soit le domaine. L'intérêt de la programmation par composant pour la réutilisation de code logiciel a également été montré. Nous désirons intégrer la notion de patron et la programmation par composants aux SIAD de manière à réutiliser les acquis dans un domaine, en ayant une approche de réutilisation, à assurer l'évolution du SIAD avec l'entreprise et à utiliser un SIAD comme gestionnaire de connaissance pour l'entreprise.

Pour conclure ce chapitre, la quatrième partie a présenté les principaux systèmes existant dans le domaine ferroviaire au niveau de l'analyse du réseau. Nous avons constaté qu'il existe de nombreux systèmes et de nombreuses recherches dans ce domaine mais nous avons pu aussi remarquer qu'il n'existe pas d'outils permettant d'analyser le système ferroviaire suivant plusieurs critères simultanément. Or, pour qu'un décideur puisse prendre une décision, nous avons vu qu'il doit connaître la répercussion de ses choix ; il doit utiliser un ensemble d'outils de mesure lui permettant de connaître leurs répercussions. Un décideur a pour cela besoin d'utiliser un ensemble d'outils (existants ou non) pour mener à bien

Outils ferroviaires	Paramètres		Objectifs						Degré de précision des informations à saisir par l'utilisateur		
			Evaluer la capacité				Evaluer la robus-tesse				
	Infrastructure	circulation	Construction d'horaires	saturation	Simulation	Calcul	Simulation	Analyse	Elevé	Intermédiaire	Bas
CAPRES	X	X	X	X					X		
DONS		X	X						X		
VIRIATO		X	X							X	
INFRAFER, module de calcul de la capacité	X	X		X							X
RECIFE		X		X					X		
DEMIURGE	X				X					X	
RAILSIM NETWORK SIMULATOR	X	X			X				X		
RAILSYS		X	X		X		X		X		
INFRAFER, module de simulation	X	X			X		X			X	
SISYFE		X			X		X		X		
FASTA		X					X				X
DONS-SIMULATOR		X					X		X		
OPENTRACK	X	X			X		X		X		
SIMONE		X					X		X		
NEMO	X	X				X		X	X		
PETER		X	X	X				X		X	

TABLEAU 1.2 – Comparaison des systèmes d'évaluation du système ferroviaire

son étude. Il a également besoin d'un SIAD qui le guide tout au long de son processus de décision, notamment dans le choix, l'utilisation des outils utiles à son étude et pour l'interprétation des résultats des outils.

En résumé, les SIAD recouvrent plusieurs domaines (cf. Figure 1.12). Pour concevoir de tels systèmes, il faut respecter les règles de chacun des domaines qu'ils engendrent. Le chapitre suivant a pour but de présenter les approches de développement dans ces domaines afin de faire ressortir les éléments indispensables au développement de SIAD tel que nous l'avons défini : un SIAD est avant tout un système interactif. Il doit assister un décideur tout au long de son processus de décision par des interactions adaptées. Il est composé d'outils interactifs de mesure. Les interactions entre l'utilisateur, le SIAD et l'ensemble des outils doivent permettre à l'utilisateur de prendre une ou plusieurs décisions. Pour cela, les interactions entre le SIAD et l'utilisateur (décideur) doivent respecter les processus de prise de décision de celui-ci. Dans le domaine qui nous préoccupe, nous considérons donc un **SIAD comme un système interactif, permettant une gestion des connaissances, basé sur des composants et intervenant tout au long du processus de décision**.

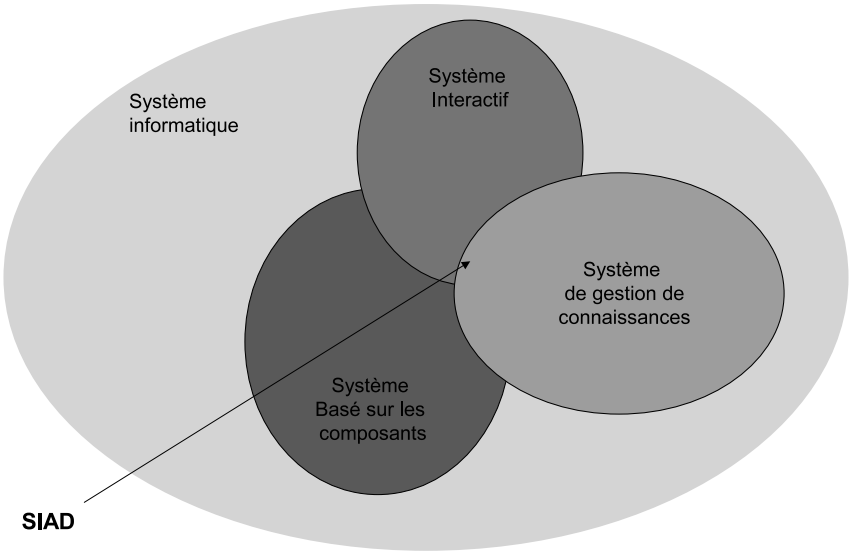


FIGURE 1.12 – Positionnement de la manière dont nous verrons les SIAD dans la suite du mémoire

Chapitre 2

Des approches classiques de développement de systèmes aux approches dédiées : modèles, méthodes et architectures

Sommaire

2.1 Introduction

Nous avons vu dans le premier chapitre que les SIAD étaient à l'intersection de plusieurs domaines. Ils sont avant tout des logiciels, doivent être interactifs, et peuvent avoir des caractéristiques communes avec les systèmes de connaissances. Dans un cadre de réutilisabilité, ils peuvent également être des systèmes à base de composants. Pour concevoir les SIAD tels qu'ils sont décrits précédemment, nous nous sommes intéressés aux méthodologies de conception existantes dans les cinq domaines suivants (Figure 2.1) : génie logiciel, Interaction Homme-Machine (IHM), gestion des connaissances, développement par composants et SIAD.

Ce chapitre est composé de cinq sections comprenant pour chacune une description de plusieurs processus de développement, des méthodes de conception et des architectures logicielles de chacun de ces domaines. A la fin de ce chapitre, les particularités des approches de chacun de ces domaines seront rappelées pour analyser les besoins en vue de la mise au point d'une approche de développement de SIAD orientée sur notre problématique.

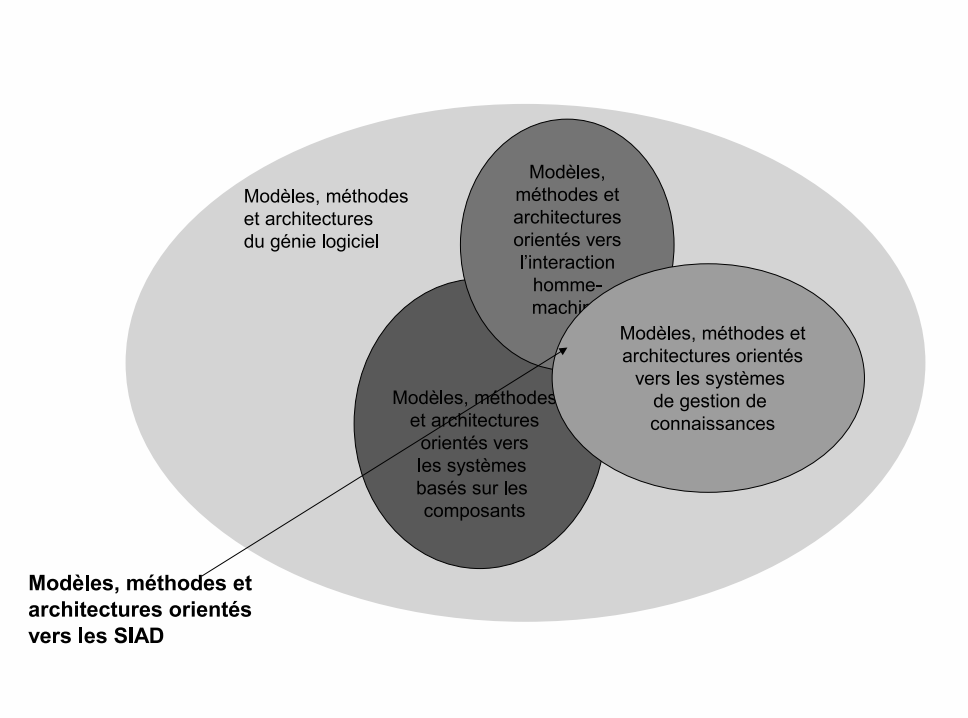


FIGURE 2.1 – Modèles de développement ayant un rôle dans le développement de SIAD

2.2 Modèles, méthodes et architectures dans le domaine général du génie logiciel

Avant de décrire des approches de développement spécifiques, nous allons commencer par présenter des approches classiques du génie logiciel.

2.2.1 Modèles de développement

Un modèle (ou cycle ou processus) de développement logiciel a pour objectif de préciser dans quel ordonnancement logique et temporel se déroulent ses étapes pour produire un logiciel. Nous assistons, depuis ces vingt dernières années, à une forte évolution des cycles classiques du génie logiciel comme les modèles en cascade (Waterfall), en V, spirale et incrémental (et leur variantes), vers des cycles intégrant la dimension humaine et favorisant fortement le prototypage.

2.2.1.1 Modèle en cascade

Le modèle en cascade, proposé par [Boehm, 1981], est un des premiers modèles apparus pour répondre aux besoins industriels en termes de productivité et de qualité logicielle. Il définit une réalisation séquentielle des étapes du processus de développement avec des retours possibles uniquement (en principe) vers l'étape précédente afin de prendre en compte les lacunes identifiées. Les notions d'analyse et de modélisation des tâches utilisateurs sont considérées de manière très informelle et par le bon sens des concepteurs les plus expérimentés. L'aspect utilisateur n'intervient le plus souvent, implicitement, que dans les étapes finales d'évaluation du produit réalisé. Ce sont pourtant des points qui nous intéressent particulièrement. Le modèle ne prévoit que des retours en arrière très restreints, ce qui peut représenter un handicap pour une conception itérative.

2.2.1.2 Modèle en V

Le modèle en V [McDermid and Ripkin, 1984] est utilisé dans de nombreuses entreprises et préconisé par des organismes de promotion de la qualité industrielle. Différentes variantes de ce modèle existent [Thayer and McGettrick, 1993, Jaulent, 1994, Laprie et al., 1995]. Il structure les étapes du cycle, qui restent globalement identiques à celles du modèle en cascade, en deux démarches : (i) descendante pour la spécification et la conception ; (ii) ascendante pour les validations et les tests. Dans chaque phase de la démarche descendante doivent être prévus le plan, les moyens et les méthodes permettant d'évaluer et de valider les résultats de la phase. Ce souci de prévoir le plus en amont possible l'évaluation du système, et ceci précisément vis-à-vis de chaque phase, constitue un point fort indéniable du modèle en V.

2.2.1.3 Modèle spirale

Contrairement aux deux premiers modèles, le modèle spirale introduit par [Boehm et al., 1984], constitue un processus itératif. Ce modèle est fort intéressant pour le développement de logiciels fortement interactifs étant donné que les besoins sont formulés progressivement, et les différents risques rencontrés sont analysés et résolus au fur et à mesure. Ceci présente l'avantage, contrairement aux modèles précédents, d'évaluer le risque avant d'entreprendre l'élaboration détaillée d'éléments logiciels de moindre risque, tant que les éléments à haut risque n'ont pas été résolus. L'état d'esprit du modèle spirale a été exploité dans nos travaux.

2.2.1.4 Modèle incrémental

Le modèle incrémental suit le modèle en cascade mais l'implantation se fait par incrément : à partir d'une phase donnée (généralement la spécification ou la conception architecturale), le processus est itéré plusieurs fois, et donne lieu à chaque fois, à la production d'un incrément. Chaque incrément correspond à un logiciel opérationnel s'approchant à chaque fois davantage du produit final par adjonction de fonctionnalités ; les évolutions entre les incréments sont guidées par l'expérience opérationnelle [ESA, 1991, Laprie et al., 1995].

2.2.1.5 Modèle transformationnel

Ce modèle indique une séquence d'étapes qui transforment graduellement une spécification en une implantation, Figure 2.2. De cette façon, chaque transformation est prouvée formellement. Ces preuves sont stockées dans un document. Ce processus favorise le développement de programme correct et documenté formellement mais exige une certaine expertise en méthodes formelles pour le développement de systèmes complexes. Ce processus est basé sur des outils d'aide à la preuve et permet de développer des composants réutilisables. Cette approche est surtout destinée aux systèmes à risques comme les systèmes embarqués et ne concerne pas nos travaux. Les méthodes supports à ce processus sont le plus souvent les méthodes/notations Z [Spivey, 1989], VDM (Vienna Development Method) [Jones, 1990] ou B [Abrial, 1996].

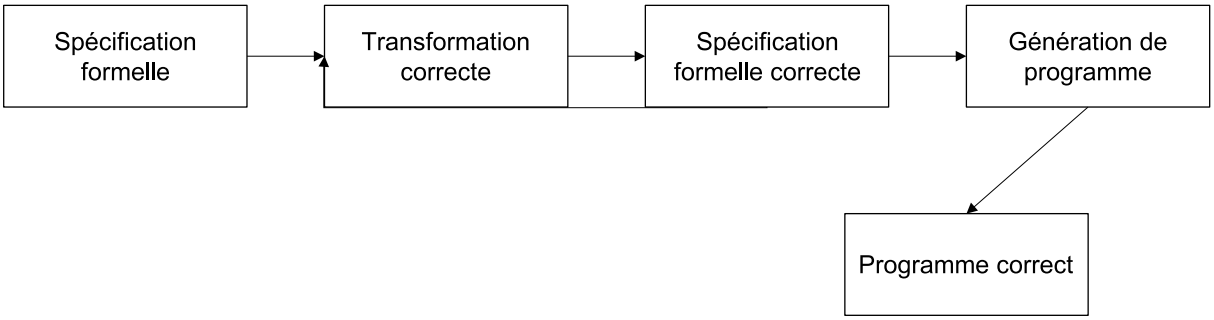


FIGURE 2.2 – modèle Transformationnel

2.2.1.6 Processus orienté objet, exemple de RUP basé sur UML

Le processus de développement orienté objets RUP (Rational Unified Process) a été crée par Rational Software [Rational Software, 2004]. Le processus unifié est piloté par les cas d'utilisation, centré sur l'architecture, itératif et incrémental [Jacobson et al., 1999]. RUP n'est pas une méthodologie standard de développement orienté objet. Il renforce six pratiques modernes de développement logiciel communes à de nombreuses méthodologies de développement : le développement itératif, la définition et la gestion des besoins, l'utilisation d'architectures à base de composants, la modélisation visuelle, la vérification de la qualité et la gestion des changements [Kruchten, 2000]. Les aspects de modélisation en analyse et conception se basent sur le langage UML (Unified Modelling Language) [Booch et al., 2000]. Ce pro-

cessus comprend quatre phases : (1) l'initialisation pour définir l'étendue du projet par l'utilisation de cas d'utilisation et l'étude de faisabilité, (2) l'élaboration pour définir les besoins et spécifier l'architecture, (3) la construction au cours de laquelle le logiciel est bâti au moyen de plusieurs itérations et de nombreuses versions du système et (4) la transition pour remettre le système aux utilisateurs finaux avec la mise en service et pour les former et les soutenir à l'utilisation.

2.2.1.7 Programmation Extrême (PE)

La programmation Extrême (Extreme Programing XP) est une approche de développement de système crée par Kent Beck [Beck, 1999] au milieu des années 1990. Elle combine des éléments d'autres approches comme le prototypage, le développement orienté objets et recommande une programmation par binôme de manière à motiver les programmeurs et à faciliter les échanges entre eux. L'approche XP est répartie sur trois niveaux (cf. Figure 2.3) : le système (l'anneau extérieur), la version (l'anneau central) et l'itération (l'anneau intérieur). Un système est livré à ses utilisateurs en plusieurs étapes appelées versions. Chaque version est un système entièrement fonctionnel exécutant un sous-ensemble des spécifications globales du système [Cloux, 2003]. D'après [Satzinger et al., 2002], le succès d'une telle méthode impose des équipes de spécialistes peu nombreux, elle n'est donc pas adaptée à des projets de grande envergure. Cette méthode est intéressante par le fait qu'elle intègre complètement les utilisateurs tout au long du cycle. Les cycles très courts permettent également la validation du système auprès des utilisateurs de manière très rapide ce qui limite les risques de rejets.

2.2.1.8 Cycle de développement par rétro-ingénierie

La rétro-ingénierie consiste à récupérer une partie des logiciels qui avec l'âge deviennent trop coûteux et difficiles à maintenir. D'après Printz [Printz, 1996], un logiciel peut être vu comme un assemblage de spécification, programmation et évaluation, dont un certain nombre de fonctions sont récupérables. Lors du développement d'un nouveau logiciel, tout ce qui était issu de la programmation et de l'évaluation d'un précédent programme devient une base pour la spécification du nouveau. Les éléments les plus facilement récupérables sont les documents d'architecture, le code source, les données dans le cas d'un logiciel utilisant des fichiers ou des bases de données, les tests, certains outils développés spécifiquement, et dans une moindre mesure, certaines analyses ou enquêtes préalables à l'expression des besoins. Dans le cas d'une politique de réutilisation, tous ces éléments doivent être archivés soigneusement, et facilement accessibles. Ce cycle nous montre à quel point une politique de réutilisation peut être intéressante. Selon Printz, en utilisant ce cycle, le coût du développement s'avère plus faible.

2.2.1.9 Conclusion

Les principaux cycles de développement ont été présentés. Nous pouvons remarquer que la tendance va aux processus itératifs (spirale, PE, RUP), mais aussi vers les processus qui tendent à intégrer la notion de réutilisation (RUP, PE, modèle transformationnel, développement par rétro-ingénierie). Nous pouvons constater que, même si les utilisateurs sont mentionnés pour les étapes d'analyse et de validation

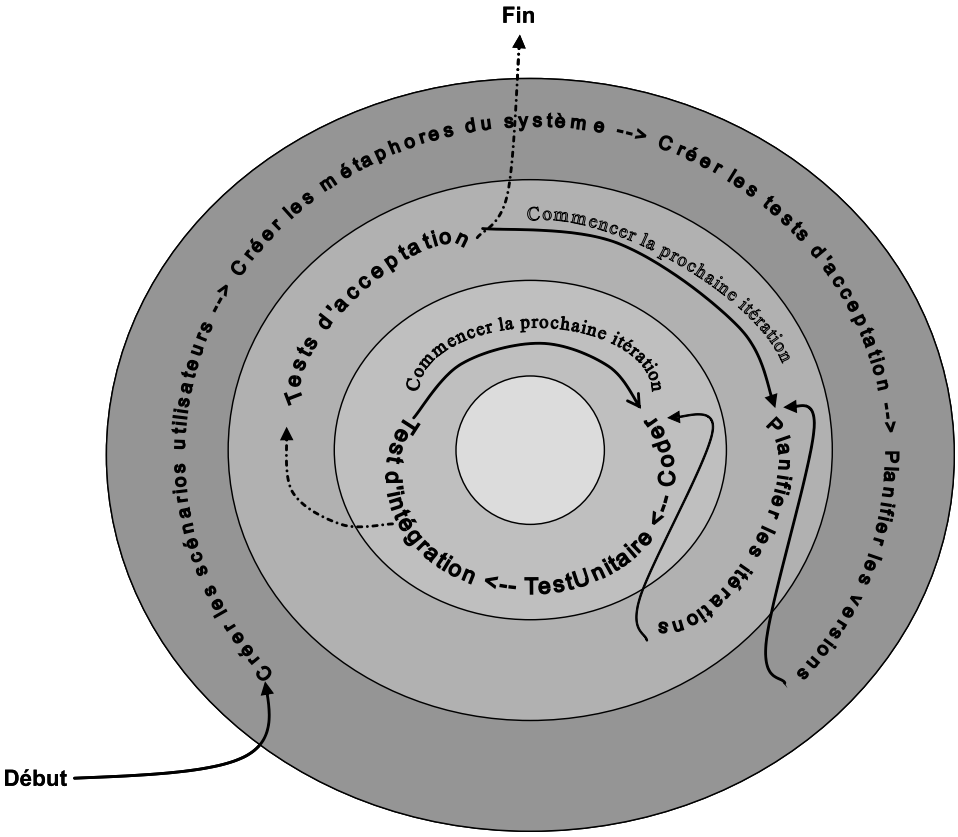


FIGURE 2.3 – Programmation Extrême

de prototype, les modèles et processus s'accompagnent le plus souvent de peu d'explications relatives à la prise en compte des utilisateurs ; la conception et l'évaluation des IHM est rarement spécifiée dans ces processus "génériques". Nous verrons quelles répercussions cela aura sur les processus de développement proposés dans le domaine des IHM.

La section suivante introduit les méthodes d'analyse et de conception et présente quelques unes de ces méthodes issues du génie logiciel.

2.2.2 Méthodes d'analyse et de conception

Selon [Silvestre and Verlhac, 1996], une méthode définit les informations à produire pour déclencher le processus de réalisation, le formalisme de présentation et les informations nécessaires au démarrage d'une tâche de conception. Il existe de nombreuses méthodes en génie logiciel, il ne sera donc pas possible de les énumérer toutes.

Quatre méthodes représentatives vont être présentées : la méthode SADT conçue en 1976 en tant que méthode cartésienne, la méthode MERISE conçue en 1978 en tant que méthode systémique, Objectory et UML comme méthodes orientées objet et la méthode RAD qui est apparue au début des années 90. Nous avons choisi de présenter ces méthodes car les deux premières sont à la base de beaucoup d'autres et ont connu beaucoup de succès, les troisième et quatrième car elles sont caractéristiques des méthodes

orientées objet et la cinquième car elle est récente et intègre l'utilisateur.

2.2.2.1 Méthodes cartésiennes : exemple de SADT

La méthode SADT (Structured Analysis and Design Technique) est une méthode créée en 1976 par D.T. Ross et est la propriété de la société Softech [Jaulent, 1994]. Elle suit une approche cartésienne qui décrit un système de manière descendante selon différents niveaux d'abstraction, modulaire, hiérarchique et structurée, en couvrant essentiellement les phases d'expression des besoins, de spécification et de conception. Elle s'appuie sur un modèle spécifique composé de datagrammes décrivant la transformation des données et d'actigrammes décrivant l'enchaînement des activités. Le datagramme est modélisé par des "boîtes" replacées dans le contexte des activités. L'actigramme est modélisé aussi par des "boîtes" identifiées par une activité, un événement ou un verbe. Ce modèle est représenté par des boîtes avec quatre types de liens : entrées, sorties, contrôles et mécanismes. L'ensemble des diagrammes est ordonné hiérarchiquement, avec au niveau le plus général, le diagramme de contexte. La méthode est basée sur un travail d'analyse orienté sur les flots de données qui permet de mettre en évidence les activités.

2.2.2.2 Méthodes systémiques : exemple de Merise

Merise a été mise au point en 1978 par un groupement formé par sept SSII et l'administration française [Tardieu et al., 1986, Tardieu et al., 2000]. Elle est la méthode systémique la plus connue et la plus employée en France en 1996 [Silvestre and Verlhac, 1996]. Un des points forts de Merise est de proposer trois niveaux avec des préoccupations complémentaires : (1) Le niveau conceptuel comprend le modèle conceptuel de données et le modèle conceptuel des traitements, (2) Le niveau logique (ou organisationnel) est représenté par le modèle logique des données et le modèle organisationnel des traitements, (3) le niveau physique (ou opérationnel) comprend le modèle physique des données et le modèle opérationnel des traitements.

Suivant cette méthode tout projet informatique suit trois cycles : abstraction, décision et vie. Le cycle de vie permet de décrire la vie du système. Il distingue les périodes qui vont de la conception à la maintenance. Le cycle de décision concerne les différentes décisions et les choix qui sont effectués tout au long du cycle de vie. Le cycle d'abstraction offre les concepts pour pouvoir décrire les différents éléments du monde réel qui seront représentés dans le système d'information. Merise a évolué au fur et à mesure des nouvelles technologies, par exemple en intégrant les notions objets dans une de ses versions récentes [Gabay, 2001].

2.2.2.3 Méthodes orientées objet : Objectory, UML

Objectory (contraction de l'anglais Object Factory for Software Development qui signifie une usine à objets pour le développement logiciel) gère le développement itératif d'un système sur toute la durée du cycle de vie. Cette méthode considère chaque itération comme la modification d'un système existant et supporte la chaîne entière, depuis les besoins jusqu'au système opérationnel [Jacobson et al., 1993]. Avec la technique des briques de base, un système est considéré comme un ensemble de blocs connectés,

chaque bloc représentant un service du système. Cette technique a été combinée avec une technique de modélisation conceptuelle et avec la programmation orientée objet.

UML (Unified Modeling Language) est apparue en 1995 comme méthode unificatrice des méthodes objets comme OMT, OOSE, OMT-2, Objectory. Cependant UML, même si elle est parfois considérée comme une méthode, n'est principalement qu'un langage de modélisation.

2.2.2.4 RAD

RAD pour (Rapid Application Development) est une méthode de conduite de projet qui a été proposée en 1991 par James Martin. Son principal but est comme son nom l'indique de permettre un développement rapide d'une application. Elle vise à intégrer réellement les utilisateurs en leur donnant des rôles opérationnels. Les dix principes de base résumant cette méthode, cités par Hugues et ses collègues [Hugues et al., 1996], sont :

- Confier l'expression des besoins aux utilisateurs ; la responsabilité et la description de ce que l'on attend du système est confié aux utilisateurs.
- Organiser l'expression des besoins ; RAD aide les utilisateurs pour organiser les besoins, elle permet des points de vue différents et contradictoires,
- Introduire une dimension temporelle dans l'expression des besoins ; RAD permet d'affiner progressivement les besoins pour prendre en compte les modifications,
- Ajuster les besoins ; RAD limite la durée de développement, le contenu du projet est adapté à cette limite,
- Raccourcir les circuits de décision ; RAD organise le projet de façon à y intégrer les décideurs,
- Structurer les problèmes selon la structure de décision ; le découpage du domaine se base sur la structure de décision de l'entreprise,
- Utiliser des techniques existantes, RAD s'appuie sur des techniques spécifiques tout au long du projet auxquelles on peut ajouter une technique pour l'organisation de la réutilisabilité,
- Travailler en session participative ; ce mode de travail réunit les utilisateurs et les informaticiens ; c'est un mode de travail collectif, intense et limité dans le temps,
- Anticiper ; toute réunion doit être préparée, la charge de préparation étant répartie entre les différents acteurs, et
- Utiliser des outils performants ; l'outil de prototypage s'accompagne d'une aide à la conception et d'un référentiel facilitant la documentation et la réutilisation.

Cette méthode s'appuie sur un cycle de vie qui combine une approche linéaire et une approche spirale, cf. Figure 2.4.

Cette méthode est intéressante et originale, elle prend en compte l'utilisateur tout au long du projet, mentionne la réutilisation comme principe permettant un développement plus rapide. Elle n'est, comme toutes les méthodes, pas universelle.

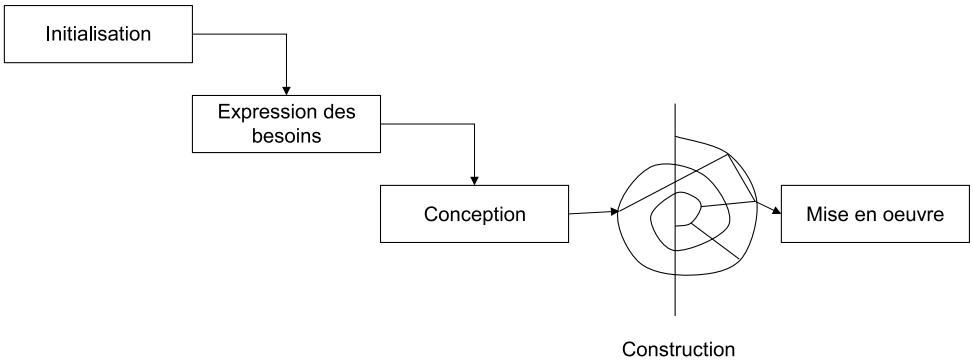


FIGURE 2.4 – Cycle de vie de RAD

2.2.2.5 Conclusion sur les méthodes issues du génie logiciel

Les deux premières méthodes présentées succinctement sont deux des plus anciennes. Elles ont l'inconvénient de ne pas faire intervenir les utilisateurs explicitement. Il existe des travaux dans le domaine du génie logiciel ou des interactions Homme-Machine qui ont permis de mettre en évidence certaines lacunes de nombreuses méthodes [Pascot and Bernadas, 1993, Kolski, 1997, Adam, 2000, Kolski, 2001, Abed, 2001]. Les méthodes comparées par ces auteurs étaient nombreuses :

- Les méthodes représentatives de celles basées sur l'analyse et la conception structurées dont SASD et JSD,
- Une méthode systémique MERISE,
- SSADM qui est une méthode dite structurée fortement accès sur la gestion de projet,
- IE (Information Engineering) influencée également par les méthodes structurées, et
- Les méthodes orientées objet dont OOA/OOD, OOD, OMT, Objectory, et UML, AXIAL visant le développement de systèmes d'information ont été retenues,
- REMORA qui a le souci de description de l'existant,
- La méthode structurée SADT,
- La méthode RAD (Rapid Application Development) axée sur le prototypage et
- Les méthodes concernant les organisations OSSAD et CISAD.

Les bilans sortant de ces études montrent que ces méthodes n'exploitent pas la diversité des modèles de développement. En effet, elles sont décrites autour d'un cycle de développement unique. Un autre constat concerne l'implication de l'utilisateur dans ces méthodes ; en effet, mises à part RAD et IE, aucune méthode actuelle ne fait vraiment intervenir l'utilisateur du début à la fin du projet. D'une manière générale, l'utilisateur reste trop peu impliqué. Nous verrons dans la partie suivante comment certaines d'entre elles ont été adaptées à ces notions d'IHM. De plus les notions de réutilisation sont peu intégrées dans ces méthodes. Elles ne comprennent souvent pas d'explication supplémentaire concernant l'analyse et la conception de systèmes spécifiques tels que les SBC et les SIAD. En résumé, ces méthodes restent très générales et sont difficiles à appliquer lorsqu'il s'agit de concevoir des systèmes particuliers.

2.2.3 Architectures logicielles

Il n'existe pas à ce jour de consensus pour définir une architecture logicielle. Cependant, il devient évident qu'une architecture peut servir de référence tout au long du cycle de vie d'un système [Gacek et al., 1995]. D'après Sanlaville, une architecture logicielle cherche à faciliter et à organiser l'implémentation, l'exécution, l'exploitation et la maintenance d'un logiciel afin de répondre aux besoins des différents acteurs [Sanlaville, 2002]. C'est pourquoi nous avons décidé de consacrer une section sur les architectures logicielles.

Nous avons dit qu'il n'existait pas de consensus quand aux définitions ; nous n'avons pas pour objectif de donner les différentes définitions et de les critiquer (pour cela les travaux de Sanlaville sont très intéressants [Sanlaville, 2002]), nous nous contenterons de donner une définition qui nous satisfait dans le cadre de nos travaux. L'architecture d'un système informatique est un ensemble de structures comprenant chacune : des composants, les propriétés extérieurement visibles de ces composants et les relations qu'ils entretiennent. Dans cette définition donnée par [Bass et al., 1998] et citée dans [Coutaz and Nigay, 2001] et dans [Sanlaville, 2002], un composant est une unité d'abstraction dont la nature dépend de la structure considérée dans le processus de conception architecturale. Ce peut être un service, un module, une bibliothèque, un processus, une procédure, un objet, une application, etc. Les propriétés extérieurement visibles de ces composants définissent son comportement attendu, elles traduisent les hypothèses que d'autres composants peuvent faire sur ce composant.

Le processus de développement architectural introduit par [Coutaz and Nigay, 2001] intègre plusieurs étapes, Figure 2.5. Il démarre du modèle de référence et des spécifications externes. La décomposition fonctionnelle consiste à exprimer les besoins fonctionnels du système en des unités plus simples. Cette activité aboutit à une structure appelée architecture conceptuelle. A partir de cette architecture, la décomposition modulaire définit une structuration statique du système, l'identification des processus et l'assignation de modules aux processus. Cette décomposition amène à l'architecture implémentationnelle en déterminant les modules qui serviront de lieu d'exécution aux processus et en allouant ces processus au processus qui répartit la charge de calcul sur plusieurs calculateurs.

Pour chaque domaine traité, des architectures dédiées au type de système seront présentées. Quand cela est possible, des modèles architecturaux plus génériques pourront être présentés. Sinon ces modèles sont inexistants ou inconnus ; nous donnerons alors des exemples d'architectures implémentationnelles. Nous n'avons pas de modèle à présenter dans le domaine du génie logiciel. Les architectures les plus connues sont les architectures pipeline, tableau noir, basée sur les événements, par couches et N-tiers.

2.2.4 Conclusion sur les modèles, méthodes et architectures issus du génie logiciel

Les approches présentées précédemment, sont certainement les plus connues de celles issues du génie logiciel. Il s'agit de rappeler que le but de cette section était de montrer globalement l'évolution dans le domaine du génie logiciel. Les parties suivantes montreront comment ces approches ont été adaptées dans les autres domaines. Nous nous intéressons dans ce mémoire aux points importants du génie logiciel qui peuvent ressortir de l'ensemble de ces approches et qui sont :

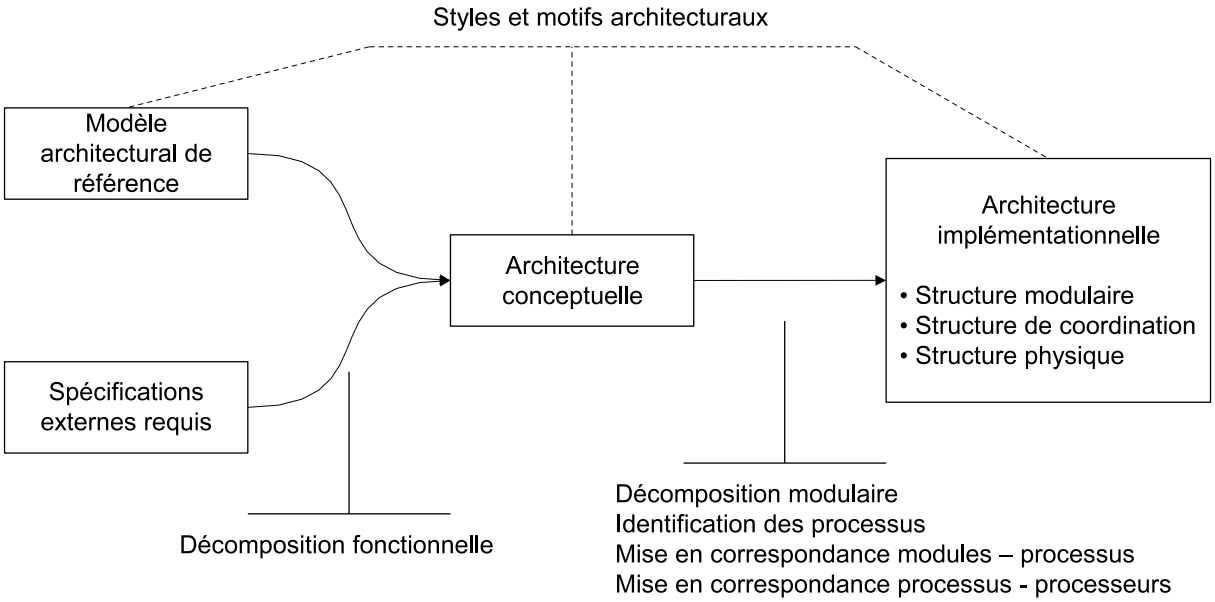


FIGURE 2.5 – Processus de développement architectural

– Pour les modèles : les phases de spécification et de conception doivent prévoir le plan, les moyens et les méthodes permettant leur évaluation et leur validation. Le processus itératif permet une formulation progressive des besoins, ainsi que l'analyse et la résolution des risques au fur et à mesure de l'avancement du projet. L'importance du prototypage pour permettre des validations intermédiaires en lien avec des utilisateurs est montrée par un ensemble de modèles.
– Les méthodes doivent suivre, si possible, l'ensemble du cycle de vie ou tout au moins une partie de celui-ci. Elles doivent être outillées.
– Les architectures ont également un rôle important dans plusieurs étapes du cycle de vie. On constate un manque de modèles à ce sujet.

2.3 Modèles, méthodes et architectures orientés vers l'Interaction Homme-Machine

Le manque d'implication des utilisateurs dans les modèles et méthodes du génie logiciel a engendré des recherches dans le domaine des Interactions Homme-Machine de manière à mieux intégrer l'utilisateur dans les projets. Les principaux résultats de ces recherches sont présentés dans cette partie dans le but d'analyser les démarches qu'utilisent les spécialistes des interactions homme-machine pour intégrer l'utilisateur.

2.3.1 Modèles de développement enrichis sous l'angle de l'interaction homme-machine

Durant la présentation de certains modèles dédiés à l'interaction homme-machine, les modifications apportées aux modèles classiques du génie logiciel seront soulignées pour faire ressortir des caractéris-

tiques importantes dues aux interactions homme-machine.

2.3.1.1 Modèle PRODUSER

Le modèle PRODUSER (PROcess for Developing USER interfaces) proposé par James [James, 1991] s'est inspiré du modèle spirale, cf. figure 2.6. Un aspect important de ce modèle est d'incorporer la gestion du risque dans son processus pour améliorer la qualité des interfaces utilisateur. Il propose l'utilisation du prototypage comme moyen pour minimiser ce risque. Il se veut applicable à toutes tailles de projet et suffisamment flexible pour aider les équipes de projet qui ne désirent pas utiliser l'ensemble des capacités de ce processus. Le modèle PRODUSER est composé de quatre parties : les besoins, les spécifications abstraites et les spécifications concrètes et l'évaluation et le contrôle du risque. Elles sont détaillées suivant les quatre plans : planification de l'étape, définitions des objectifs, alternatives et contraintes, évaluation des alternatives permettant de gérer le risque et développement et vérification. Les phases sont centrées sur l'interface utilisateur.

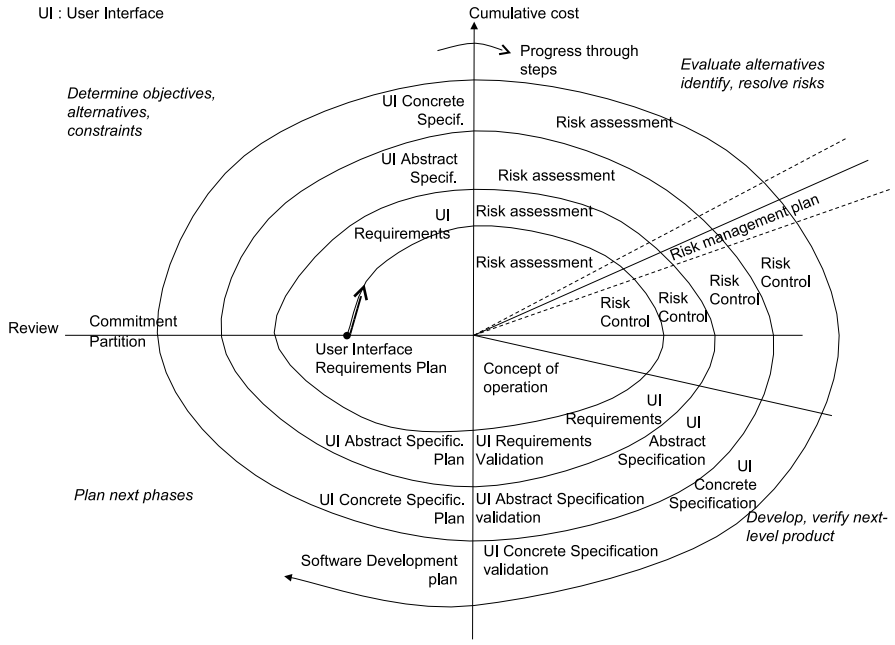


FIGURE 2.6 – Modèle PRODUSER [James, 1991]

2.3.1.2 Modèle de Curtis et Hefley

Le modèle de Curtis et Hefley [Curtis and Hefley, 1994], est intéressant par le fait qu'il situe le travail à réaliser pour chacune des étapes classiques en Génie Logiciel, dans la partie gauche du modèle autour des interactions homme-machine, dans sa partie droite autour d'aspects liés habituellement au développement de logiciel, cf. figure 2.7. Il précise donc les tâches complémentaires à effectuer tout au long du projet, ce qui peut s'avérer fort utile pour les chefs de projet.

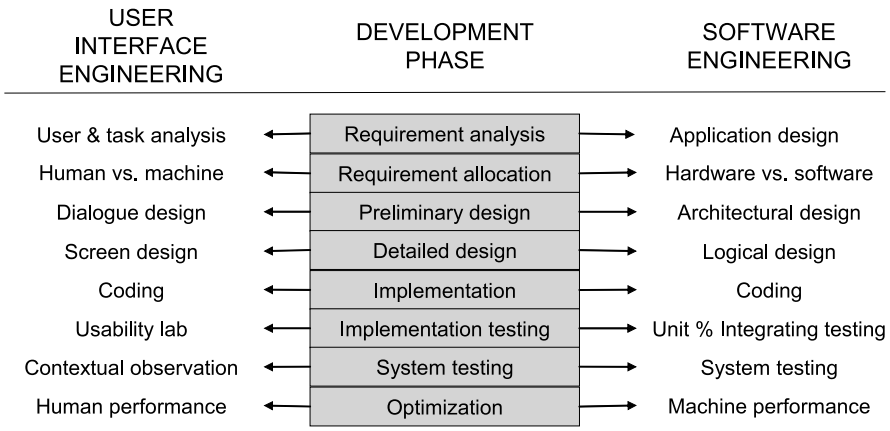


FIGURE 2.7 – Modèle de Curtis et Hefley [Curtis and Hefley, 1994]

2.3.1.3 Modèle en étoile

Le modèle proposé par [Hix and Hartson, 1993], appelé aussi modèle en étoile, situe l'évaluation au centre même du cycle complet, montrant ainsi des interactions/itérations possibles entre chacune des autres étapes, cf. figure 2.8. L'étape d'évaluation est vue comme une étape intermédiaire permettant de protéger l'équipe de développement d'un rejet terminal, à l'allure de sanction. Même si ce modèle se situe assez loin d'un modèle classique, cette idée le rend intéressant. Il n'impose pas a priori d'ordre dans l'accomplissement des étapes du processus, bien qu'en pratique les activités de développement soient reportées en fin de cycle. Notons qu'il sous-entend une conception participative visant la détection précoce de problème d'utilisabilité, requérant une forte adhésion de l'utilisateur par cette implication centrale [Hix and Hartson, 1993, Poltrock and Grundin, 1995].

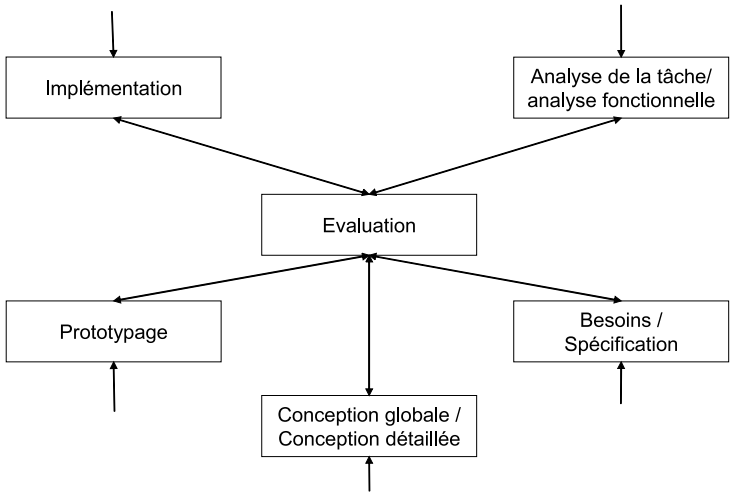


FIGURE 2.8 – Modèle en étoile (traduit de [Hartson and Hix, 1989])

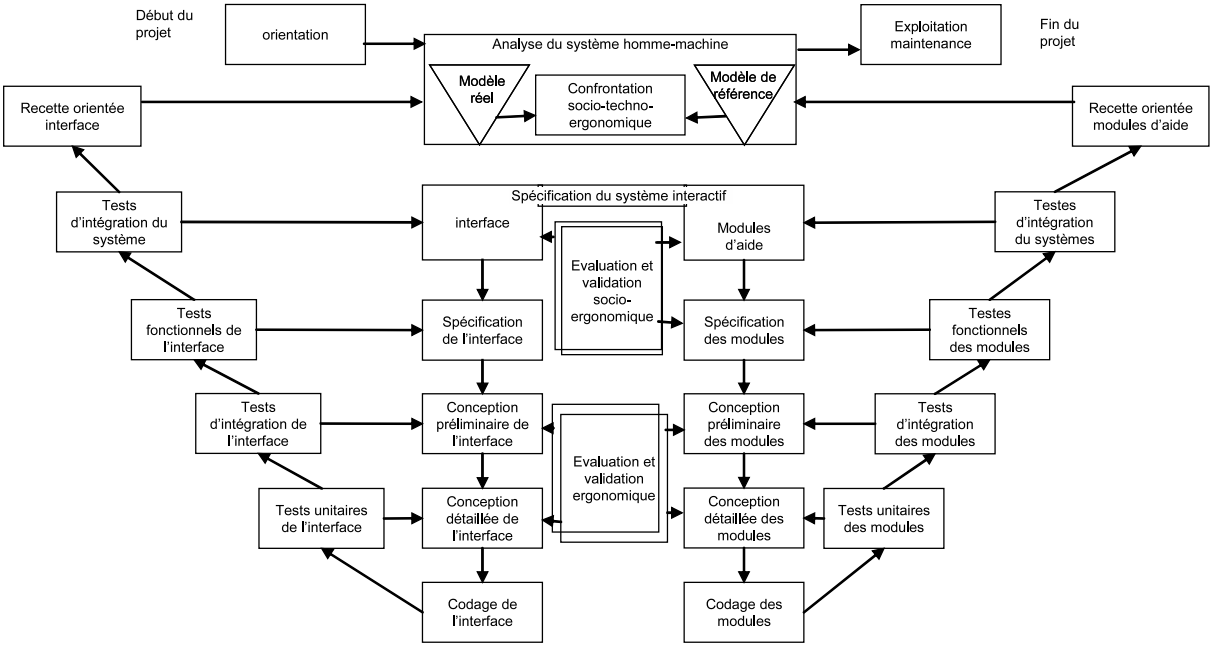


FIGURE 2.9 – Modèle ∇ [Kolski, 1995, Kolski, 1997]

2.3.1.4 Modèle ∇

Le modèle (prononcer nabla) [Kolski, 1995, Kolski, 1997], construit selon un double cycle en V, situe les différentes étapes du génie logiciel nécessaires pour développer un système interactif, tout en différenciant l'interface proprement dite (partie gauche du modèle) des modules d'aide (ou applicatifs) éventuellement accessibles à partir de ceux-ci (partie droite), cf. figure 2.9. Nabla se base sur une confrontation progressive entre un modèle réel et un modèle de référence, où le modèle de référence correspond à celui d'un système homme-machine dit idéal, en considérant les points de vue et besoins des différents intervenants concernés par le système homme-machine visé. Le résultat de cette confrontation conduit à identifier les données pertinentes pour spécifier un système interactif adapté aux besoins informationnels des utilisateurs, ainsi qu'aux besoins en mode de coopération utilisateur-modules d'aide. L'ensemble des spécifications est ensuite évalué et validé d'un point de vue socio-ergonomique, afin de vérifier la pertinence de l'intégration des solutions nouvelles dans le système homme-machine visé. L'évaluation se situe au centre du projet suggérant une démarche itérative aussi bien dans les parties gauche que droite. Elle se termine par l'étape de recette qui se différencie symboliquement en une recette orientée interface et une recette orientée modules d'aide.

2.3.1.5 Modèle en U

Une des caractéristiques marquantes du modèle en U est de positionner des étapes – inexistantes dans les modèles classiques du génie logiciel, qui restent très généraux – où les facteurs humains devront être considérés par l'équipe de développement [Millot and Roussillon, 1991, Abed et al., 1991, Abed, 2001, Lepreux et al., 2003a]. Il est structuré en deux phases (cf. figure 2.10) : (i) une phase descendante de

modélisation du système homme-machine, et qui aboutit à sa mise en œuvre, (ii) une phase ascendante consistant en l'évaluation du système global, selon des critères d'efficacité du système mais également des critères strictement humains. La validation consiste à confronter le modèle du système de la phase descendante avec le modèle du système en phase ascendante. Le résultat de cette confrontation permet soit de valider le système homme-machine soit de mettre en évidence ses carences et à affiner progressivement celui-ci, particulièrement au niveau des interfaces homme-machine et des outils d'aide. Le modèle final résultant de la confrontation permet ainsi de généraliser des comportements spécifiques de l'homme dans des conditions particulières de travail, réutilisable dans des situations des systèmes similaires.

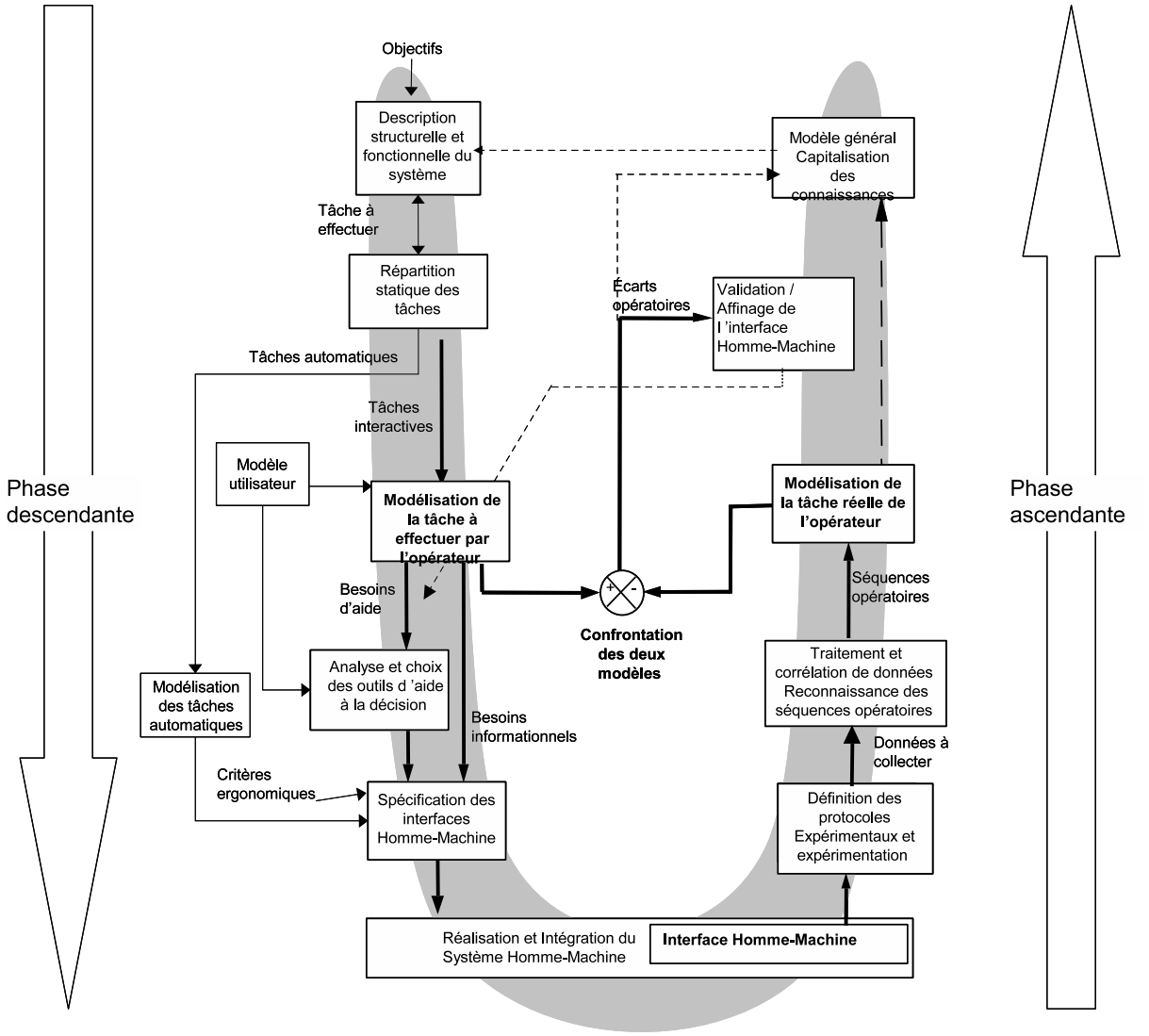


FIGURE 2.10 – Modèle en U [Abed, 2001]

2.3.1.6 Conclusion sur les modèles orientés vers les interactions homme-machine

Les modèles présentés montrent les évolutions apportées par le domaine des IHM au génie logiciel. Parfois, les cycles présentés sont difficilement utilisables car ils ne sont pas suffisamment complet (comme le cycle Etoile) mais ils montrent le manque dans les modèles du génie logiciel. Ces modèles mettent l'accent sur des idées essentielles pour le développement de systèmes interactifs comme :

- fixer les activités pour les différents intervenants,
- placer l'évaluation au centre du processus,
- modéliser les activités humaines, les interfaces homme-machine et le système et
- confronter les modélisations des activités à effectuer du début du cycle avec les modélisations de ces activités réelles en fin de cycle.

2.3.2 Méthodes d'analyse et de conception de systèmes interactifs

Les méthodes d'analyse et de conception de systèmes interactifs sont nombreuses [8]. Nous en présenterons plusieurs parmi les plus intéressantes dans le cadre de notre travail. Nous ne nous intéressons pas ici aux méthodes permettant uniquement d'analyser et modéliser les tâches humaines (voir à ce sujet le handbook de [Diaper and Stanton, 2004]). Les méthodes qui vont être présentées sont DIANE+, SADT/Petri et TOOD. Les deux premières méthodes sont basées sur MERISE pour DIANE+ et comme son nom l'indique SADT et les réseaux de Petri pour SADT/Petri. TOOD est une évolution de SADT/Petri qui comme Merise utilise un ensemble de modèles pour couvrir plusieurs étapes du cycle de vie d'un système interactif.

2.3.2.1 Méthode DIANE+, influencée par les méthodes systémiques

Diane+ [Tarby and Barthet, 1996, Tarby and Barthet, 2001] étend la méthode Diane [Barthet, 1986] qui est née dans les années 1980 pour pallier les insuffisances de la méthode MERISE quand à la prise en compte des caractéristiques de l'utilisateur et la conception de l'interface homme-machine. En effet, Diane+ est une méthode centrée sur l'utilisateur. Diane+ est basée sur des concepts suffisamment abstraits pour être indépendante des évolutions technologiques. Elle est orientée vers la conception participative. Par expérience, on sait qu'un formalisme, aussi puissant soit-il, dès qu'il est trop rigoureux ou trop contraignant, peut être rejeté par certains concepteurs ou des utilisateurs ayant à participer à des validations à partir de modèles, c'est pourquoi Diane+ n'est pas formelle. Diane+ peut être considérée comme une méthode d'accompagnement à la conception. Elle sert de base à la génération de code de l'application (données, traitements, IHM, aide).

[8] Le lecteur intéressé en trouvera dans [Kolski, 2001] qui résume les méthodologies MUSE [Lim and Long, 1994], et d'autres dans TRIDENT [Bodart et al., 1995]. Dans [Tabary, 2001] est proposé une synthèse sur les approches dites à base de modèles [Szekely, 1996], telle : MOBI-D (Model-Based Interface Designer) [Puerta and Maulsby, 1997], ALACIE (Atelier logiciel d'aide à la conception d'Interfaces Ergonomiques) [Gamboa-Rodriguez, 1998].

2.3.2.2 Méthode SADT/Petri, influencée par les méthodes cartésiennes

La méthode SADT/Petri se base sur le modèle en U (cf. 2.3.1.5). Cette méthode s'applique à l'analyse, la spécification et l'évaluation des systèmes interactifs. Le principal intérêt d'associer SADT et les réseaux de Petri est de permettre la structuration fonctionnelle du système en termes de tâches par SADT, et de décrire son comportement dynamique par les réseaux de Petri. Les données enregistrées durant l'activité sont structurées en séquences opératoires et formalisées en modèles de la tâche effectuée ou réelle. Ces modèles sont traduits dans le même formalisme que les modèles de la tâche à effectuer (ou prescrite) obtenus par les outils SADT et Réseau de Petri [Abed et al., 1991, Abed et al., 1995, Abed et al., 2001]. Une confrontation entre ces deux modèles de la tâche permet de mettre en relief les corrections à apporter au système lors de sa conception.

2.3.2.3 Méthode TOOD, influencée par les méthodes objets

La méthode TOOD [Mahfoudi, 1997, Abed and Tabary, 1998, Tabary, 2001] se base sur la transformation d'une série de modèles pour couvrir le cycle de développement, de l'analyse de la tâche à l'implémentation. Elle fait partie des méthodes à base de modèles (model-based development [Szekely, 1996]). TOOD s'appuie :

- Dans l'étape de spécification, sur un Modèle de la Tâche et un Modèle des Objets du Domaine ;
- Dans l'étape de conception, sur un Modèle Opérationnel, un modèle de l'utilisateur et un Modèle Local de l'Interface ;
- Dans celle de réalisation, sur un Modèle d'Implémentation.

Le modèle de la tâche spécifie "le quoi" et montre "le quand" les traitements principaux sont réalisés. Il définit également les données d'entrée, les données de sortie, ainsi que les ressources (humaine et système) nécessaires à l'accomplissement de la tâche. Deux types de modèles sont issus de cette étape de spécification : un modèle statique pour exprimer la structure de la tâche, l'autre dynamique pour représenter le comportement de celle-ci vis-à-vis de son environnement. En se basant sur l'approche orientée objet, le Modèle Statique de la Tâche fait apparaître les Classes-Tâches représentant les traitements attendus du système. Le Modèle Dynamique de la Tâche utilise les Réseaux de Petri Objet pour décrire comment la tâche gère et traite les ressources qu'elle manipule en fonction de son état interne. Dans TOOD les données sont appelées Objets du Domaine. Ils sont représentés dans le Modèle des Objets et formalisés de manière similaire aux tâches.

L'étape de conception affine chaque tâche terminale jusqu'à aboutir à des traitements élémentaires (appliqués aux Objets du Domaine) clairement détaillés. Elle contribue à l'expression du lien entre la partie applicative et la partie IHM du système en se basant sur le modèle de l'utilisateur et le modèle local de l'interface. En effet, l'objectif du modèle de l'utilisateur est de comprendre et de formaliser le comportement de l'utilisateur et en conséquence d'établir une conception de l'interface centrée utilisateur. Le Modèle Local de l'Interface décrit le comportement des Objets Interactifs qui aident l'utilisateur dans l'accomplissement de sa tâche en mettant à sa disposition un ensemble de services.

La dernière étape de la méthode consiste à implémenter les éléments du modèle de l'interface issus de l'étape de conception vers une architecture orientée-agents de type PAC (Présentation, Contrôle Abstraction) [Coutaz, 1987, Coutaz and Nigay, 2001](cf. 2.3.3.3).

2.3.2.4 Conclusion sur les méthodes d'analyse et de conception de systèmes interactifs

Les méthodes présentées ont montré comment des méthodes du génie logiciel pouvaient être adaptées pour l'intégration explicite de l'Interaction Homme-Machine lors de l'analyse et la conception, soit par une meilleure prise en considération de l'utilisateur tout au long du projet, soit par une modélisation de celui-ci ou de ses actions. Notons aussi que, suivant un nouveau courant de recherche, d'autres approches basées sur la théorie de l'activité [9] s'appuient sur le fait que des besoins naissent avec l'utilisation d'outils pour la réalisation d'une activité ; ces approches ont pour but d'intégrer l'utilisateur comme concepteur dans l'évolution d'un système [Bourguin, 2000]. D'une manière générale l'évolution des besoins doit être prise en compte en permanence dans la conception d'un système et le suivi de celui-ci. L'aspect de réutilisation est parfois évoqué mais n'est pas le point central des méthodes présentées. Nous reviendrons sur celui-ci plus loin (cf. 2.4).

2.3.3 Architectures des systèmes interactifs

Les principaux modèles d'architectures proposés pour la conception des interfaces Homme-Machine vont être présentés. Les modèles les plus anciens, dont le modèle de Seeheim [Pfaff, 1985] fait partie, ont mis en valeur la séparation entre l'application et les fonctions chargées d'assurer l'interaction avec l'utilisateur. De ces modèles ont découlé des modèles plus précis et plus orientés vers l'implémentation comme le modèle Arch [Bass et al., 1991]. D'autres modèles fondés sur des approches réparties ou à base d'agents, comme PAC [Coutaz, 1987, Coutaz, 1990, Coutaz and Nigay, 2001], MVC [Krasner and Pope, 1988], ALV [Hill, 1992] ont été développés pour répondre aux exigences de modularité, de parallélisme, d'encapsulation/affinement et de dépendance fonctionnelle.

Dans un but d'illustration, nous présenterons dans cette section le modèle de Seeheim qui est le plus ancien et qui en a inspiré d'autres, comme le modèle Arch et le modèle PAC qui seront également présentés.

2.3.3.1 Le modèle de Seeheim

Le modèle de Seeheim [Pfaff, 1985] tient son nom du lieu où il a été défini à l'occasion d'un workshop sur les systèmes de gestion des interfaces utilisateurs. Il découpe la structuration de l'interface en trois composants logiques, Figure 2.11 :

– La présentation correspond aux aspects lexicaux du dialogue en entrée et en sortie. Ce composant prend en charge les composants qui apparaissent sur l'écran et les différentes interactions physiques.

[9] Parmi les travaux sur la théorie de l'activité notons l'article de Stary [Stary, 2003] concernant ses travaux sur TADEUS (Task Analysis / Design / End User Systems).

Il doit convertir les actions de l'utilisateur en unités de dialogue plus abstraites, adaptées au besoin des autres composants.

– L'interface avec l'application s'occupe de l'invocation des fonctions de l'application. Ce composant représente la sémantique de l'application indépendamment de l'interface utilisateur.
– Le contrôleur de dialogue s'occupe des aspects syntaxiques de l'interaction. Ce composant gère la coopération et le dialogue entre l'utilisateur et l'application (le noyau fonctionnel). Il sert d'intermédiaire entre la présentation et l'interface avec l'application.

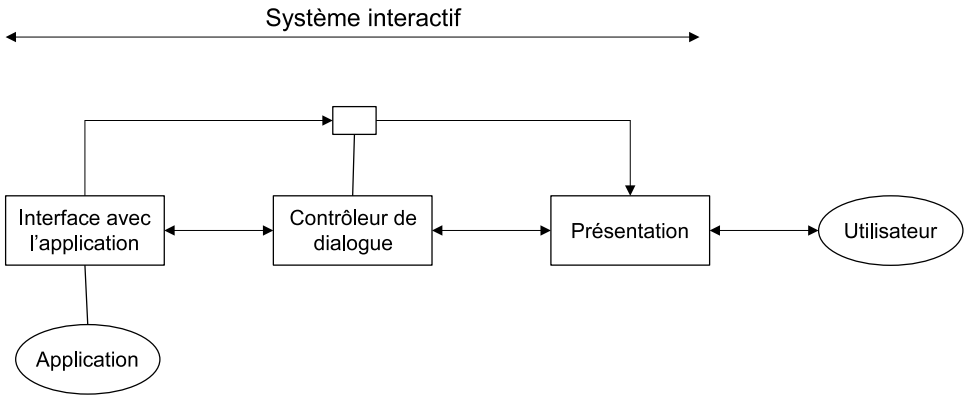


FIGURE 2.11 – Modèle de Seeheim

Coutaz [Coutaz, 1990] a identifié les apports de ce modèle se résumant en :

– la définition d'un cadre de pensée intégrant les notions de lexique, de syntaxe et de sémantique permet de jeter les bases structurelles du système à construire ;
– la conception itérative des interfaces est possible grâce à la modularité des interfaces qui distingue les aspects lexicaux, syntaxiques et sémantiques ;
– la généricité est due au fait que le modèle Seeheim ne fait aucune hypothèse sur la nature des applications et n'est lié à aucune technique de réalisation ; et
– la généralité est possible car le modèle n'est pas seulement applicable à la construction des systèmes interactifs mais aussi à leur conception et à leur réalisation automatisée.

Autrement dit ce modèle est en principe applicable à tout système interactif. Il pourrait être vu comme incarnant une norme architecturale de tels systèmes. De ces quatre apports cités par Coutaz, celui qui nous semble le plus intéressant est le second relatif à une conception itérative car désormais la majorité des conceptions de systèmes interactifs sont itératives pour permettre une meilleure prise en considération des besoins et de l'évolution de ces besoins.

2.3.3.2 Le modèle ARCH

Le modèle ARCH [Bass et al., 1991] est un modèle révisé de celui de Seeheim. Il affine le contrôleur de dialogue du modèle de Seeheim par l'introduction des trois composants centrés sur la portabilité et la réutilisabilité. Il divise une application interactive en cinq composants, Figure 2.12 :

– Les pieds de l'arche sont constitués d'une part du *composant noyau fonctionnel* réunissant les don-

nées et les traitements propres à l'accomplissements de tâches du domaine, et d'autre part du *composant d'interaction*, en liaison avec les boîtes à outils, implémentant les objets physiques de l'interaction.

– Le *composant de présentation* assure la médiation entre le composant de dialogue et le composant d'interaction. Il propose au composant de dialogue des objets de présentation conceptuels (abstraits) indépendants des boîtes à outils par lesquels ils sont réalisés. Cette fonction permet d'isoler l'application des environnements particuliers.
– Le *composant de dialogue* est chargé du séquencement des tâches et de la traduction de données entre le noyau fonctionnel et la présentation ainsi que leur formalisme respectif.
– Le *composant adaptateur de domaine* assure l'interfaçage entre le contrôleur de dialogue et le noyau fonctionnel, et garantissant une totale indépendance entre eux. Il permet ainsi de déclencher les procédures nécessaires à la tâche de l'utilisateur, de détecter et réparer les erreurs sémantiques, d'implémenter les protocoles de communication entre dialogue et application, etc.

Ce modèle a été critiqué en raison de la lourdeur imposée par l'introduction des composants adaptateur de domaine et de présentation. De ce fait, le méta-modèle Slinky[10] a été introduit pour mettre l'accent sur le fait que les architectures logicielles doivent être dimensionnées en fonction d'un compromis entre plusieurs contraintes et critères selon le cas traité.

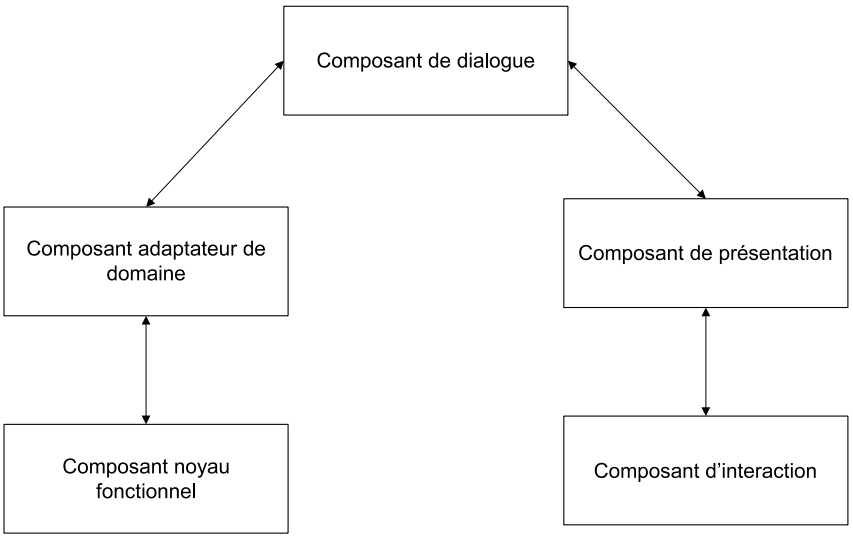


FIGURE 2.12 – Modèle ARCH

2.3.3.3 Le modèle PAC (Présentation, Abstraction, Contrôle)

Proposé par Coutaz [Coutaz, 1987], le modèle PAC prend appui sur une approche à base d'agents. C'est pourquoi Coutaz parle d' "agent PAC". Un agent PAC se modélise selon trois perspectives : la Présentation, l'Abstraction et le Contrôle. La présentation définit le comportement perceptible de l'agent auprès d'un agent humain. L'abstraction définit la compétence de l'agent indépendamment des considé-

[10] Slinky est le nom d'un jouet qui une fois mis en mouvement, voit sa masse se déplacer dynamiquement.

rations de présentation. Le contrôle a un double rôle : il sert de pont entre les facettes Présentation et Abstraction de l'agent et il gère les relations avec les autres agents PAC. Le fonctionnement d'un agent PAC est illustré par l'exemple du thermomètre dans la Figure 2.13. PAC structure récursivement un système interactif sous forme d'une hiérarchie d'agents. Il permet de distinguer les services abstraits des techniques d'interaction en introduisant un intermédiaire explicite : le Contrôle. Cette propriété d'indépendance présente plusieurs avantages dont la satisfaction du critère de réutilisabilité des constituants de l'interface et la modification de l'interface à moindre coût.

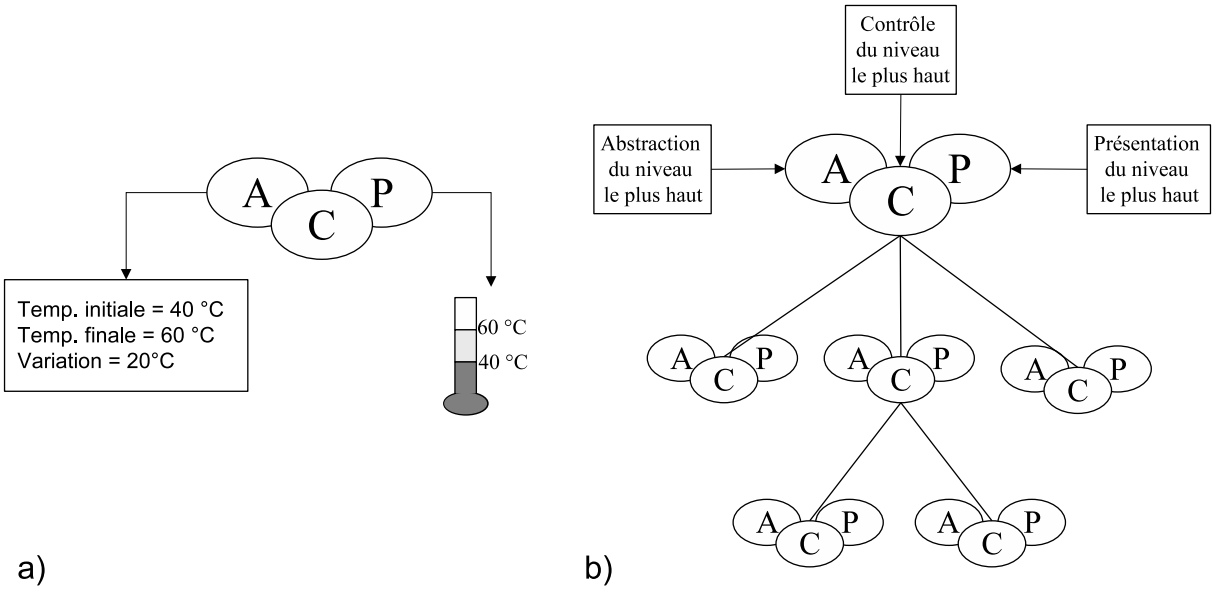


FIGURE 2.13 – a) un exemple d'agent PAC b) Le modèle d'architecture PAC [Coutaz, 1987]

2.3.3.4 Conclusion sur les architectures proposées dans le domaine des interactions homme-machine

Les trois modèles d'architectures présentés sont basés sur le même principe qui est la séparation de l'interface utilisateur des modules fonctionnels par un module de contrôle. Cette règle est devenue une règle de base pour la conception de systèmes interactifs. Son importance a encore été renforcée par le développement des interactions multimodales [Nigay, 1994, Coutaz and Nigay, 2001]. En effet, pour que les systèmes soient compatibles avec une grande variété des supports, sans être obligés d'être développés de manière spécifique à chaque support, ils doivent adopter une architecture respectant les modèles présentés.

2.3.4 Conclusion sur les modèles, méthodes et architectures orientés vers l'interaction homme-machine

Les approches de développement issues du domaine des IHM ont montré que les processus du génie logiciel avaient besoin de plus prendre en compte l'utilisateur et mieux les interactions homme-machine

en mettant l'accent sur le prototypage, l'évaluation, le positionnement explicite des activités des acteurs dans le processus de développement et l'analyse de l'activité. Les méthodes quand à elles ont évolué vers une conception participative et ont précisé l'importance de la modélisation de l'activité humaine.

2.4 Modèles, méthodes et architectures orientés vers la réutilisation

Nous allons dans une première partie présenter les approches adaptées à la réutilisation puis nous en mettrons plusieurs en évidence particulièrement axées vers une programmation par composants.

2.4.1 Modèles de développement adaptés à la réutilisation

La réutilisation a modifié les démarches classiques de développement pour en distinguer deux types : les processus de développement *pour* la réutilisation et les processus de développement *par* la réutilisation. L'objectif du premier est d'identifier et de développer des composants réutilisables tandis que l'objectif du second est d'utiliser les composants réutilisables issus du processus précédent pour concevoir et réaliser un système. Dans cette partie, les modèles en X, Nabla adapté à la réutilisation, en Y et IPM vont être présentés. Le cycle en X est à la fois *pour* et *par* la réutilisation. Les modèles en Nabla et en Y sont un processus de développement par réutilisation. IPM est lui aussi spécifique à la réutilisation en proposant deux phases correspondant à un développement *pour* la réutilisation et à un développement *par* réutilisation.

2.4.1.1 Le cycle en X

Le cycle en X a été proposé par Ralph Hodgson [Hodgson, 1991]. Il permet d'introduire la réutilisation dans le développement en V. Coulange [Coulange, 1996] a adapté ce cycle pour introduire les besoins liés à la mise en œuvre d'une politique de réutilisation, cf. Figure 2.14. Les phases hautes du cycle ont la même signification que dans le cycle en V classique.

Une première différence majeure avec le cycle en V est que la fin de la phase de validation ne signifie pas la fin des travaux sur le logiciel que l'on veut réutiliser. Il reste à traiter les phases d'acquisition et la phase d'archivage des composants. La phase d'acquisition est celle durant laquelle on va identifier les parties du projet réalisé qui sont réutilisables. Tandis que la seconde phase consiste à stocker les composants identifiés pendant la phase d'acquisition afin de les retrouver ultérieurement. Ces deux phases concernent les composants d'analyse, de conception et de code.

L'autre différence majeure avec le cycle en V est la prise en compte de la réutilisation dans les phases de spécification, conception générale, conception détaillée et codage. Au fur à mesure de l'avancement dans ces phases, les librairies correspondant à l'étape de développement sont consultées pour déterminer si les constituants sont déjà connus et réutilisables.

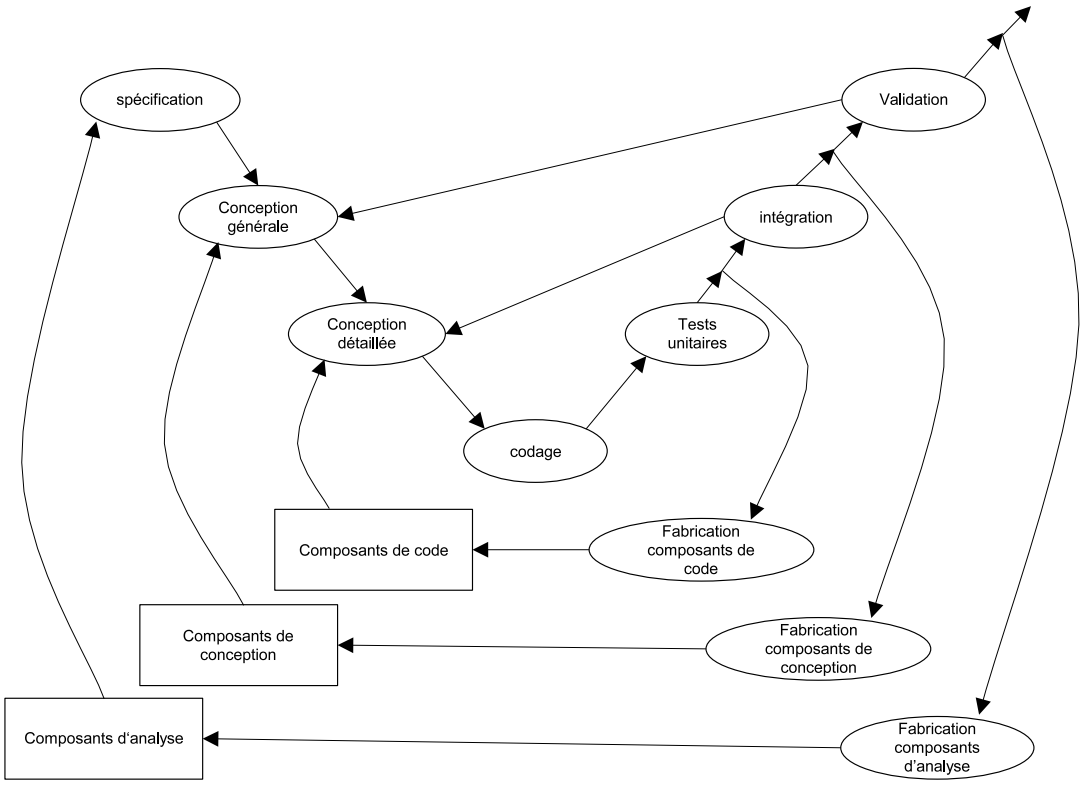


FIGURE 2.14 – Cycle en X adapté par Coulange

2.4.1.2 Nabla adapté pour la réutilisation

De la même manière que pour le modèle précédent, le modèle Nabla, étant basé sur un principe de double cycle en V (cf. 2.3.1.4) [Kolski, 1997], a été transformé pour prendre en compte la réutilisation. Le principe de réutilisabilité peut être envisagée selon des aspects liés à l'interface homme-machine, et selon des aspects liés aux modules d'aide éventuellement disponibles à partir de l'interface homme-machine. La réutilisabilité des interfaces homme-machine (cf. Figure 2.15a) peut être envisagée selon trois niveaux : le premier vise la centralisation de composants de base réutilisables tels qu'une courbe, le second rend disponibles des services tels qu'une fenêtre affichant des messages d'erreurs tandis que le dernier, de niveau le plus haut, rend disponibles des architectures d'imageries ayant fait leur preuve dans le domaine concerné tels que les architectures d'IHM de supervision dans le domaine des systèmes industriels. La réutilisabilité des modules d'aide (cf. Figure 2.15b), respecte les niveaux de la méthode KADS (cf. 2.5.2.1) liées aux systèmes à base de connaissance qui sont au nombre de quatre dans une de ses versions initiales : le niveau "domaine" correspond à la théorie du domaine et comprend des connaissances statiques, le second appelé "inférence" concerne les inférences qui peuvent être réalisées sur les entités du niveau précédent, le troisième niveau nommé "tâche" décrit comment effectuer et séquencer les inférences afin d'atteindre un but, et le dernier intitulé "stratégie" comprend des plans de résolution sous forme d'enchaînement de différentes tâches.

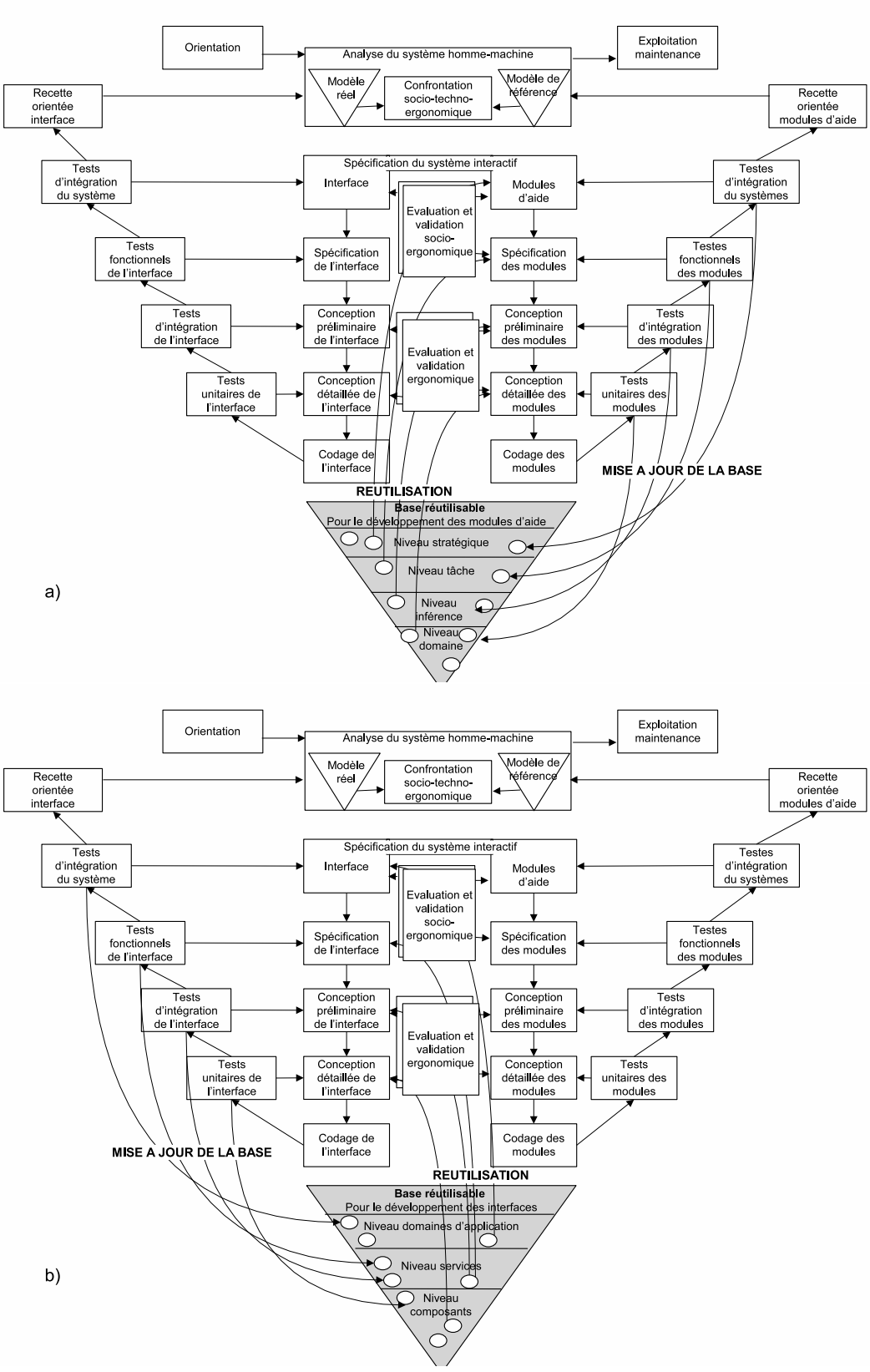


FIGURE 2.15 – Modèle ∇ et principe de réutilisabilité a) pour le développement de l'interface homme-machine b) pour le développement des modules d'aide

2.4.1.3 Modèle en Y

Le modèle en Y, proposé par [André, 1993] et cité et adapté dans [Larvet, 1994], a inspiré Hassine et ses collègues [Hassine et al., 2002b] pour aboutir au modèle présenté en Figure 2.16. Cette figure met en évidence que les composants métier (CM) (resp. composants techniques ; CT) sont identifiés et analysés dans la "branche fonctionnelle" (resp. technique). Dans la branche centrale du "Y" (conception, codage, etc.) les composants conceptuels (CM et CT) sont progressivement transformés en composants logiciels. Le principe de ce modèle est d'apporter une aide à la maîtrise du développement et à l'optimisation des coûts et des délais en parallélisant les tâches d'analyse et de conception.

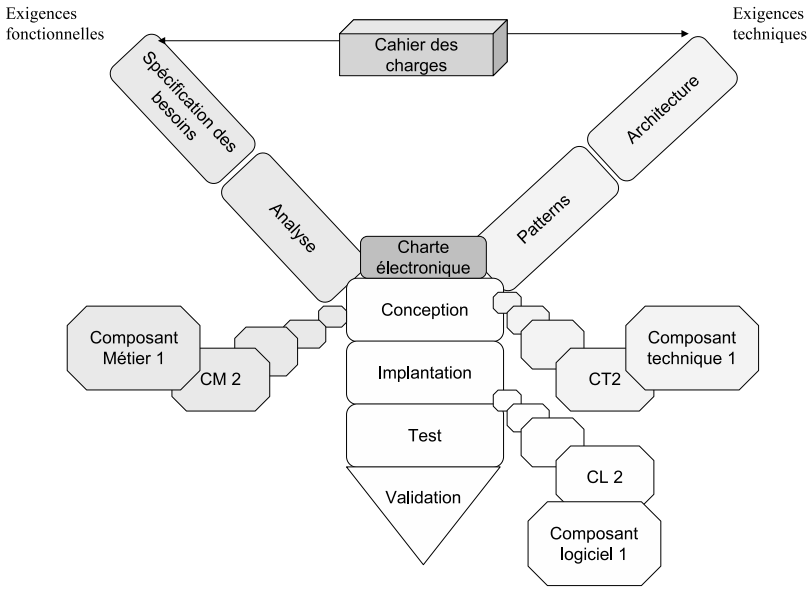


FIGURE 2.16 – Modèle en Y [Hassine et al., 2002b]

2.4.1.4 IPM : Incremental Process Model

IPM est un modèle de processus incrémental [Almeida et al., 2003]. Il est composé de deux phases (Figure 2.17) : une phase de conception de composants distribués qui sont stockés dans un dépôt, et une phase de conception d'application distribuée utilisant les composants existant dans son dépôt. La première phase comprend quatre étapes : définition du problème, spécification des composants, conception des composants et implémentation des composants. La seconde phase comprend trois étapes : la spécification de l'application, la conception de l'application et l'implémentation de l'application.

Ce modèle rassemble le développement pour la réutilisation en proposant un moyen de fabriquer et stocker des composants et un développement par réutilisation en donnant le moyen de rechercher et de réutiliser les composants déjà conçus. De notre point de vue, ce modèle ne montre pas qu'il est possible de concevoir des composants en fonction des besoins de l'application. D'après le schéma, les besoins de l'application impliquent des besoins en composants. Ceux-ci sont recherchés dans la base mais dans l'optique où un composant n'existe pas encore, aucune information ne permet de dire comment il sera

conçu et intégré au dépôt.

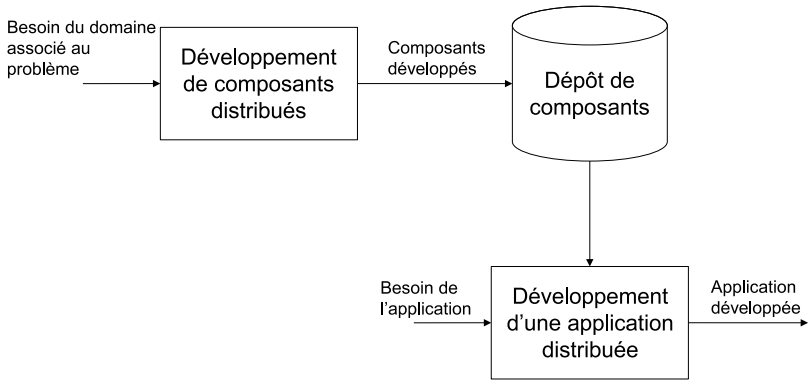


FIGURE 2.17 – IPM pour le développement de logiciel basé sur des composants distribués

2.4.1.5 Conclusion sur les modèles adaptés à la réutilisation

Les approches les plus anciennes ne prenaient pas en compte les deux aspects de développement : pour réutiliser et par réutilisation. Les trois types de composants à savoir du domaine (ou métier), de conception (ou technique) et logiciels (ou codage), sont par contre apparus dès les premiers modèles proposés. La principale critique que l'on peut apporter aux processus de développement axés sur la réutilisation est qu'ils ne prennent pas en compte l'utilisateur sauf dans le cas de Nabla qui est issu du domaine de l'interaction homme-machine et qui présente quelques notions de réutilisation. Il manque donc un processus qui soit axé aussi bien sur les interactions homme-machine que sur la réutilisation.

2.4.2 Méthodes axées réutilisation proposées dans le domaine de l'ingénierie des composants

Pour satisfaire le besoin en réutilisation, nécessaire au développement rapide et plus sûr, les systèmes ont évolué en intégrant la notion de composants, appelés systèmes à base de composants. Cette évolution a une répercussion sur les méthodes de conception dites "classiques" du génie logiciel car il faut désormais prendre en compte la conception des composants, dite "conception pour réutilisation", et la conception du système "composé" dite "conception par réutilisation". Quelques méthodes seront décrites succinctement.

2.4.2.1 Catalysis

Dans la méthode Catalysis [D'Souza, 2003], un composant est "un paquetage cohérent de logiciel qui peut être développé indépendamment et livré comme une unité, et qui définit les interfaces par lesquelles il peut être composé avec d'autres composants pour fournir et utiliser des services". Le processus de développement que propose Catalysis se décompose en quatre grandes étapes : l'analyse des besoins, les spécifications du système, la conception de l'architecture puis la conception interne des composants.

Cette méthode utilise un dictionnaire pour la mise en relation avec la conception des interfaces utilisateur et la conception de la base de données. C'est une méthode dite "par réutilisation" et "pour la réutilisation".

2.4.2.2 Symphony

Symphony [Hassine et al., 2002a, Hassine et al., 2002b] est un processus de développement logiciel élaboré et commercialisé par la société Umanis. Cette démarche s'appuie sur le langage unifié UML et est construite autour d'un certain nombre de pratiques. Symphony est une démarche itérative, orientée utilisateur, orientée composants, pilotée par les cas d'utilisation et adoptant le modèle en Y (cf. 2.4.1.3). Symphony fait partie des méthodes dites "par réutilisation". Symphony est une méthode à base de composants métier dont l'objectif est l'utilisation systématique de composants tout le long du cycle de développement et leur identification dès la phase d'expression des besoins.

2.4.2.3 Select Perspective

Les premières versions de Select Perspectives [Frost, 1995], cité dans [Allen and Frost, 1998], ont été créées en 1995. Elles sont conduites par les cas d'utilisation et basées sur une architecture logicielle orientée-métier. Elles évoluent ensuite pour intégrer UML. Select Perspective fournit la modélisation des composants orientés métier, la modélisation des composants de conception et la modélisation du déploiement. La méthode est basée sur une architecture s'appuyant sur les services. Le processus de développement est scindé en deux processus : le processus solution et le processus composant. Le processus solution vise le développement de solutions en terme de services utilisateurs pour fournir un maximum de besoin des utilisateurs. Tandis que le processus de composant vise à développer des composants supportant le métier et/ou les services plus spécifiques. Ces deux processus sont incrémentaux. Le processus Perspective suit donc les développements *par* et *pour* la réutilisation.

2.4.2.4 MDA (Model Driven Architecture)

MDA a été conçu par l'OMG [OMG, 2003] pour guider les spécifications de composants au moment où il y avait une multiplication des intergiciels [11]. En effet, l'OMG avait auparavant travaillé sur CORBA pour qu'il devienne un standard en intergiciels, mais la multiplication des plateformes (J2EE, .NET, Web...) et des modèles de composants a conduit à des conceptions spécifiques à la plateforme et a introduit des problèmes d'intégration. MDA propose alors un modèle stable indépendant de l'intergiciel, basé sur les standards de modélisation de l'OMG (UML (Unified Modeling Language), CWM (Common Warehouse MetaModel) et MOF (Meta Object Facility)). Cette modélisation indépendante de la plateforme (PIM, Plateform Independant Model) permet d'aboutir, grâce aux modèles, à une modélisation spécifique de la plateforme (PSM, Plateforme Specific Model) qui est ensuite utilisée pour générer le code d'implémentation. L'intérêt de MDA est de montrer qu'il est possible de spécifier des composants sans être dépendant d'une plateforme, la plateforme n'étant qu'un support final pour le développement.

[11] L'intergiciel est une couche logicielle qui s'appuie sur les systèmes d'exploitation et qui fournit des abstractions pour simplifier le développement d'applications réparties sur des réseaux d'ordinateurs [Demeure and Najm, 2002].

2.4.2.5 Conclusion sur les méthodes axées réutilisation proposées dans le domaine de l'ingénierie des composants

Les méthodes qui ont été présentées évoluent toutes vers des méthodes supportant les deux types de cycles de développement (*par* et *pour* la réutilisation). Elles tentent d'intégrer les composants métier dès le début du cycle de vie. Elles visent également à être indépendantes de la plateforme de développement. UML est un langage souvent utilisé dans ces méthodes. Elles proposent généralement une architecture de composants.

2.4.3 Architectures des systèmes à base de composants

L'architecture proposée dans Select Perspective vise à atteindre une architecture 3-tiers. L'évolution des architectures est représentée par la Figure 2.18. Elle montre la tendance à séparer les modules spécifiques. Dans Select Perspective, l'accent est mis sur l'architecture orientée métier. Celle-ci sépare les logiques métier locales (spécifique à un service) ou globales à l'entreprise.

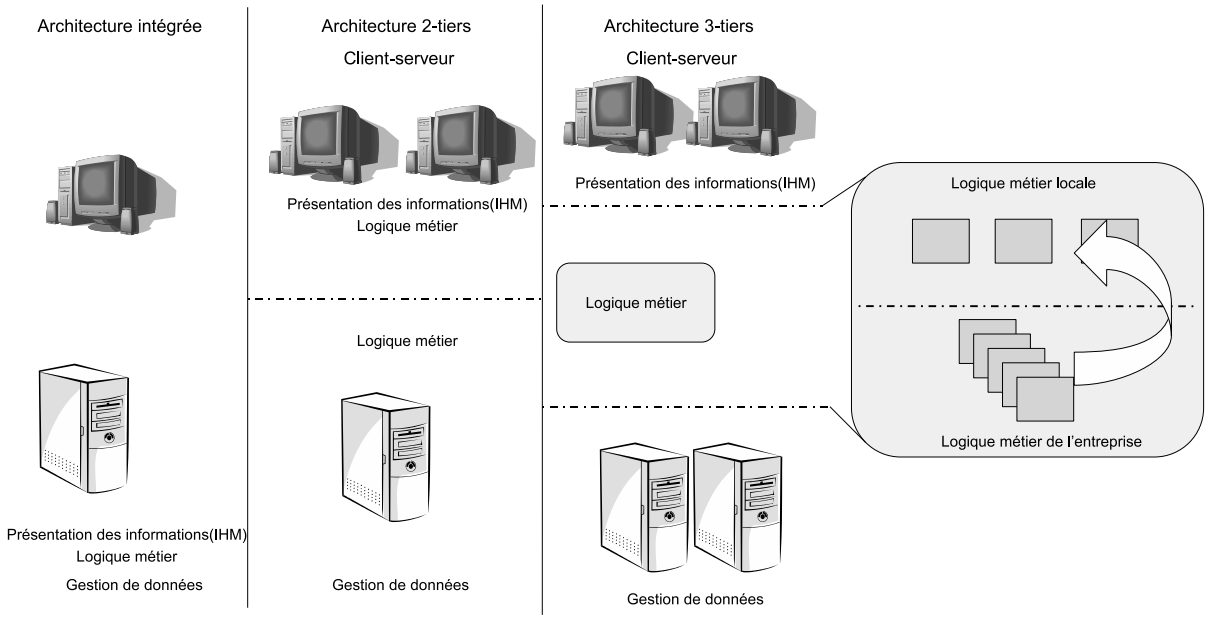


FIGURE 2.18 – Evolution des architectures vers une architecture logicielle orientée métier, traduit de [Allen and Frost, 1998]

Un exemple d'architecture implémentationnelle, issue des travaux de [Barais and Duchien, 2004], est visible en Figure 2.19. Elle représente la structure d'une station service. La station est composée d'une pompe, d'une caisse et d'un pistolet. Chacun de ces composants possèdent des ports. Ils sont liés par leur port pour former le composant composite station (assemblage). D'autres liens, tels que la délégation, permettent des interactions avec l'extérieur (le client et la banque). Cet exemple permet de montrer que les caractéristiques essentielles des architectures des systèmes à base de composants sont les interfaces qui permettent de faire le lien entre les composants.

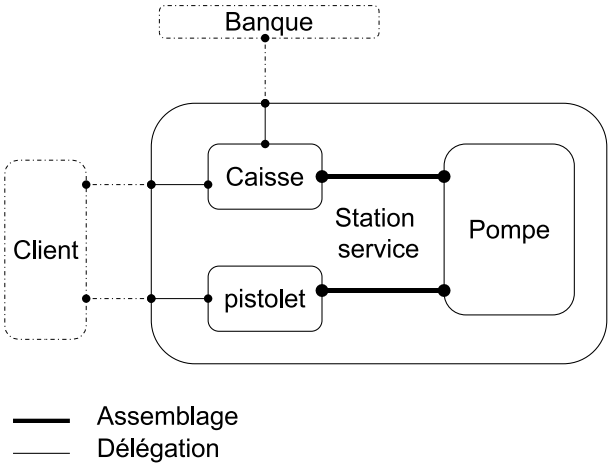


FIGURE 2.19 – Exemple d'architecture proposé par [Barais and Duchien, 2004]

2.4.3.1 Conclusion sur les modèles, méthodes et architectures orientés vers la réutilisation

Le développement axé vers la réutilisation devrait maintenant être intégré au développement classique de logiciel. Cependant, les nouveaux cycles de développement ne sont pas encore très connus et sont à leur tour spécifiques à leur domaine. Les notions inhérentes aux logiciels interactifs qui avaient été ajoutées aux modèles classiques sont négligées dans ces modèles. Les particularités de ces modèles sont qu'ils prennent en compte les composants sur l'ensemble du cycle. Les étapes de conception de composants et de logiciel sont distinguées. Il y a apparition des modèles pour le développement de composants et pour le développement de logiciel par utilisation de composants. Quelques méthodes se distinguent déjà dans ce type de processus. Par contre, il n'y a pas vraiment de modèle d'architecture particulier spécifique dans ce domaine.

2.5 Modèles, méthodes et architectures orientés vers les systèmes de connaissance

2.5.1 Modèles de développement adaptés aux systèmes de connaissance

Les cycles de vie des systèmes basés sur la connaissance différent sensiblement de ceux du génie logiciel classique par le fait que les problèmes liés à la connaissance sont trop complexes et trop globaux pour pouvoir être définis rigoureusement au départ ou spécifiés complètement avant la réalisation de tout système [Ermine, 1993]. C'est pourquoi certains cycles s'orientent vers une approche itérative utilisant le prototypage, en gardant les phases de spécification et conception classiques mais en les incluant dans un cycle incrémental de développement [Houriez, 1994].

A titre d'exemple, le modèle MODESTI (Méthodologie de développement de systèmes à base de connaissances) [Duribreux-Cocquebert, 1995] fournit le cadre d'une méthodologie de développement de SBC interactive et centrée sur l'utilisateur, cf. Figure 2.20. Elle propose une approche multi-modèles qui fait cohabiter des acteurs, des techniques et des méthodes issues de disciplines différentes mais

complémentaires telles que l'informatique, l'ingénierie des connaissances, la psychologie et l'ergonomie cognitives.

D'une manière générale les processus de développement des systèmes de connaissance n'ont pas de grandes différences par rapport aux processus classiques. Les principales différences se situent au niveau de la terminologie employée comme le précise Ermine [Ermine, 1993] : le couple marché/produit a tendance à céder la place au triplet savoir-faire/problème/solution, Tableau 2.1.

Niveau	**Système classique**	**Système à base de connaissance**
Conceptuel (quoi ?)	Besoin	Problème
Organisationnel (qui, où, quand combien, etc. ?)	Tâche fonctionnelle	Tâche cognitive
Physique (comment ?)	Service rendu	Aide à la résolution de problème

TABLEAU 2.1 – Analyse des besoins vs. Analyse du problème (d'après Ermine [Ermine, 1993])

2.5.2 Méthodes d'analyse et de conception de systèmes de connaissance

Les méthodes dans ce domaine ont évolué dans le temps. Trois méthodes parmi les plus connues composeront cette section : la méthode KADS qui a évolué vers CommonKADS, KOD vers KODMS et MOISE vers MASK.

2.5.2.1 De KADS à CommonKads

KADS (Knowledge Acquisition and Design System) [Hickman et al., 1989, Wielinga et al., 1992a] est le nom d'un projet ESPRIT qui a eu pour but de produire une méthodologie pour le développement commercial de systèmes à base de connaissances. Il définit un modèle de cycle de vie pour ces systèmes, identifie et décrit un ensemble de techniques et méthodes pour la construction de bases de connaissances. Le cycle de vie de KADS comporte une étape de spécification qui aboutit à un modèle indépendant de tout formalisme d'implémentation. Cette étape est essentielle car, dans KADS, le développement d'un système à base de connaissances est vu comme un processus de modélisation. D'après Ermine [Ermine, 2000], le modèle de la tâche de KADS est établi à partir de la décomposition d'une tâche en sous-tâches. A chaque tâche est associée une définition intégrant le quoi et le comment de celle-ci. Cette étape de spécification est suivie d'une étape de conception qui établit un modèle tourné vers l'implémentation. Au cours du temps KADS a évolué et est devenu CommonKads. Dans la méthodologie CommonKADS (Common Knowledge Acquisition and Design Support) [Wielinga et al., 1993] le cycle de vie est essentiellement décrit en termes d'activités de modélisation. Le modèle principal est le modèle de l'expertise qui caractérise le comportement de résolution de problèmes d'un agent en termes de connaissances appliquées pour résoudre une certaine tâche [Caulier, 1997, Caulier, 2001]. De plus, CommonKads est doté d'une bibliothèque de modèles réutilisables. Cette bibliothèque doit aider les

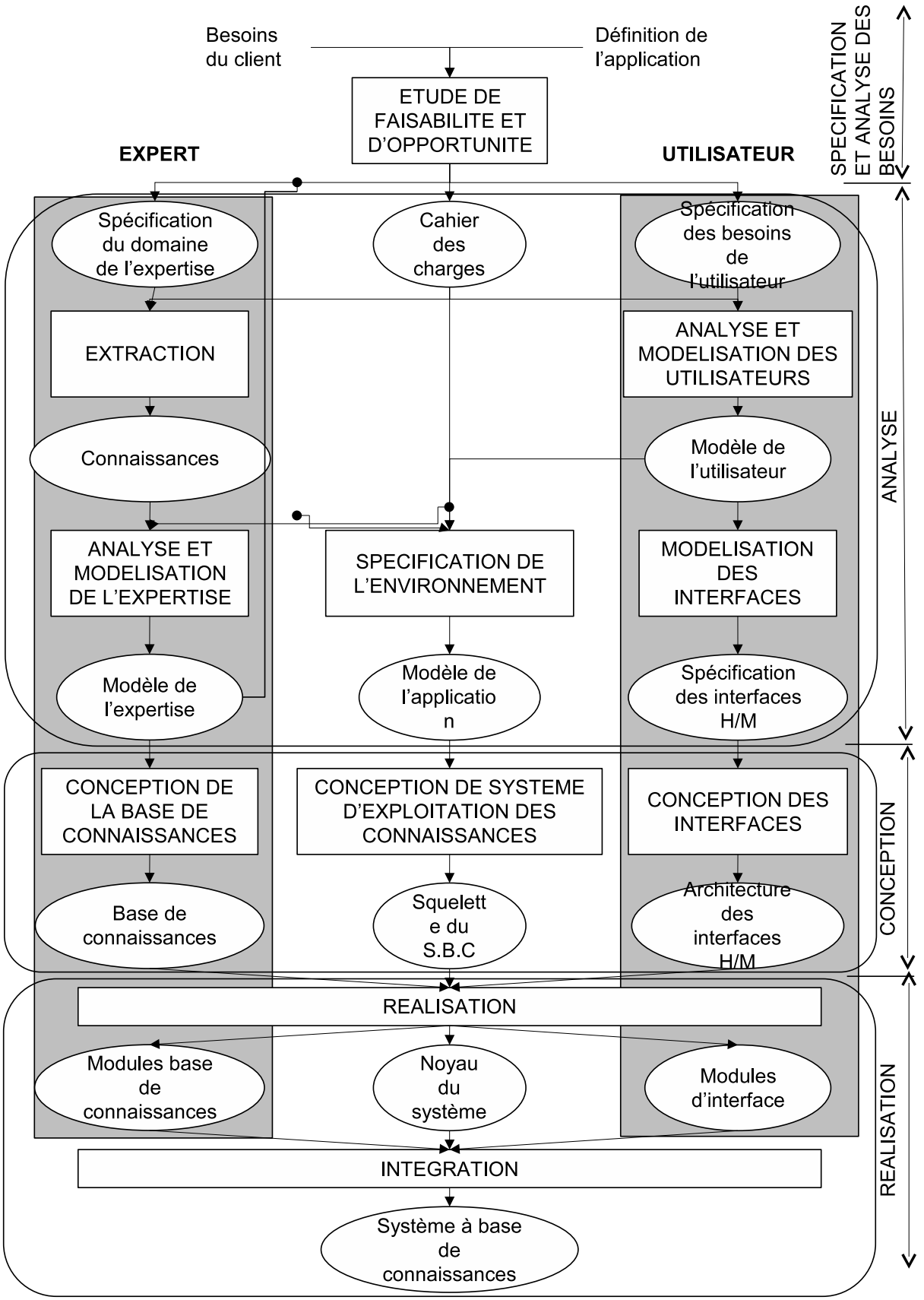


FIGURE 2.20 – Modèle MODESTI

ingénieurs de la connaissance en leur fournissant des modèles génériques et des blocs de construction réutilisables, et intègre ainsi l'expérience en modélisation de la communauté [Wielinga et al., 1992b]. CommonKADS vise à devenir un standard européen.

2.5.2.2 De KOD à KODMS

KOD (Knowledge Oriented Design) [Vogel, 1988] ne cherche pas à couvrir le cycle de vie complet d'un système à base de connaissances, mais est essentiellement orientée vers la spécification, cf. figure 2.21. C'est une méthode structurée qui utilise à la fois des techniques pour sa réalisation (interviews, recueil d'expertise...) et des représentations théoriques sophistiquées, fortement imprégnées des sciences humaines telles que l'anthropologie et la linguistique. Le modèle de spécification de KOD est de type mixte : (1) il décompose à la fois des objets et des fonctions associées à ces objets et assujettit les fonctions aux objets et (2) il associe l'information traitée aux conditions d'énonciation de cette information.

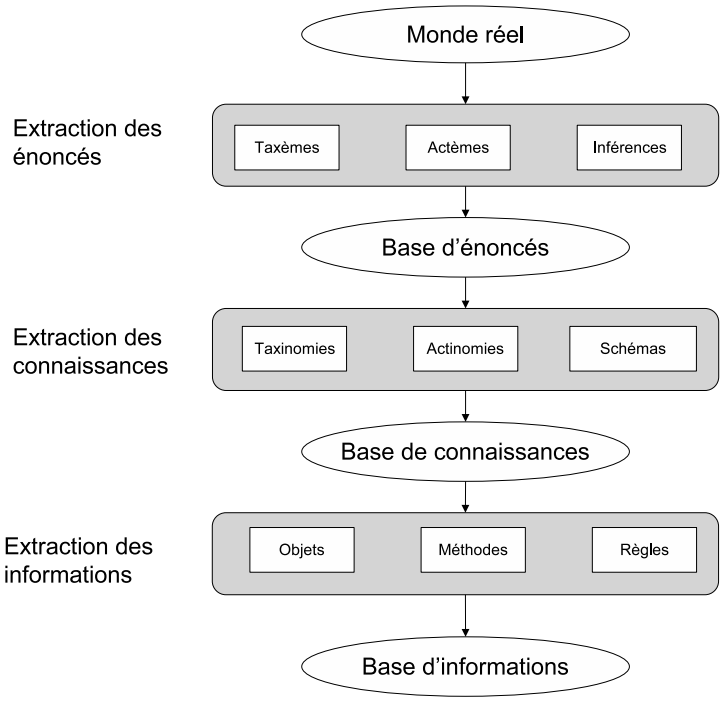


FIGURE 2.21 – Les trois niveaux de KOD [Vogel, 1988]

La méthodologie KODMS propose une extension de la méthode KOD vers les phases aval du développement en s'appuyant sur la méthode de conception HOOD (Hierarchical Object Oriented Design) [Lai, 1991]. Pour chacune des phases, KOD est recommandée pour la modélisation des connaissances pour l'aspect statique (connaissances du domaine) alors que HOOD est mieux adaptée pour la modélisation des traitements pour l'aspect dynamique (raisonnement). Cette méthode ne semble plus être utilisée.

2.5.2.3 De MOISE à MASK

La méthode MOISE (Méthode Organisée pour l'Ingénierie des Systèmes Experts) [Ermine, 1993] est une méthode de génie cognitif. Son but est de construire un modèle formalisé de l'ensemble des connaissances qui interviennent dans le problème qui est traité [Ermine, 2000]. Elle propose des modèles spécifiques à chaque phase du cycle dont notamment le modèle S2D2 signifiant "Static, Semiotic, and Dynamic Design" pour l'analyse et la spécification des connaissances.

Ensuite est apparue MKSM (Method for Knowledge Systems Management) (marque déposée du CEA (Commissariat à l'énergie atomique)) qui est une méthode de gestion des connaissances [Ermine, 1996, Ermine, 2000] qui dépasse et englobe celle du génie cognitif. Elle consiste à modéliser les connaissances selon différents points de vue des sources de connaissances de l'entreprise. La méthode MKSM conduit à la réalisation d'un livre de connaissances. La connaissance est structurée suivant les trois points de vue de l'information, de la signification et du contexte : elle est modélisée suivant cinq modèles :

- Le modèle du domaine participe à la mise en contexte des connaissances du domaine de connaissances concerné en déterminant les phénomènes généraux qui sont à la base du savoir.
- Le modèle d'activité correspond à la mise en contexte des connaissances, sous la forme d'analyse de l'activité du système qui produit ou utilise les connaissances. Ce modèle permet de replacer les connaissances du domaine (décrites par les phénomènes) dans le cadre d'une utilisation opérationnelle.
- Le modèle des concepts représente l'aspect "statique" de la connaissance, en traduisant la structuration conceptuelle d'un expert, d'une personne habituée à travailler dans un domaine précis.
- Le modèle des tâches décrit la connaissance dynamique représentant le savoir-faire utilisé pour réaliser tout ou partie de certaines activités identifiées dans le modèle d'activités.
- Les modèles d'évolution prennent en compte l'adaptation des organisations à leur environnement et la problématique de l'innovation.

MASK est une méthode d'analyse d'un patrimoine de connaissance donné. MASK peut signifier "Méthode d'Analyse et de Structuration des Konnaissances" , "Méthode d'analyse de Systèmes de Konnaissances" ou "Method for Analysing and Structuring Knowledge" [Ermine, 2001]. Cette méthode a été élaborée au CEA. Son but est la capitalisation du savoir d'experts partant à la retraite ou d'équipes de spécialistes redéployés, la structuration de corpus d'information et/ou de documents, l'intégration de savoir-faire dans des procédés industriels ou des processus d'entreprise pour améliorer leur productivité et leur compétitivité, et la diffusion de connaissances des meilleurs experts à travers des outils variés (l'hypermédia, l'aide à la décision, les livres, la formation, etc.). MASK est une méthode basée sur l'explicitation de la connaissance à l'aide de modèles de connaissances hérités des tous premiers modèles élaborés par l'ingénierie des connaissances [Ermine, 2003] et présentés ci-dessus. Elle respecte les points de vue de la méthode MOISE (information, signification et contexte) et reprend les modèles de MKSM pour former un cycle de modélisation.

2.5.2.4 Conclusion sur les méthodes d'analyse et de conception des systèmes à base de connaissance

Les aspects importants ressortant de ces méthodes sont l'importance de la modélisation de l'activité experte et de la connaissance dans son contexte. Ces méthodes se distinguent des méthodes classiques du génie logiciel par le fait qu'elles sont centrées sur la connaissance et son exploitation. Elles ne prennent pas réellement en compte les aspects "informatiques" comme dans les méthodes plus classiques vues précédemment.

2.5.3 Architectures des systèmes à base de connaissance

Il n'y a pas de modèles d'architectures connus dans ce domaine mis à part les modèles génériques de systèmes experts ou systèmes à base de connaissance proposés dans la littérature depuis les années 80. A titre d'exemple, nous prenons celui que Bernard Houriez [Houriez, 1994] a proposé pour les systèmes à base de connaissance, cf. Figure 2.22. Cette architecture est centrée sur la base de connaissances. Les modules d'interfaçage avec les utilisateurs ou avec l'environnement sont impliqués lors de la consultation du SBC. Ils permettent de saisir des faits instrumentés ou non et de leurs communiquer les résultats de l'inférence. Le module d'explication joue un rôle important car il permet à l'utilisateur de comprendre les résultats sortis. L'utilisateur peut ou non être un expert. Dans le premier cas, le module d'explication doit lui expliquer comment ont été obtenu les résultats mais peut être aussi ce qu'ils signifient. Dans le cas où l'utilisateur est lui aussi expert, il voudra vérifier le cheminement de l'inférence ayant permis de fournir les résultats. Dans cette architecture, on remarque également qu'il y a un mécanisme d'apprentissage automatique qui permet d'enrichir la base de connaissance.

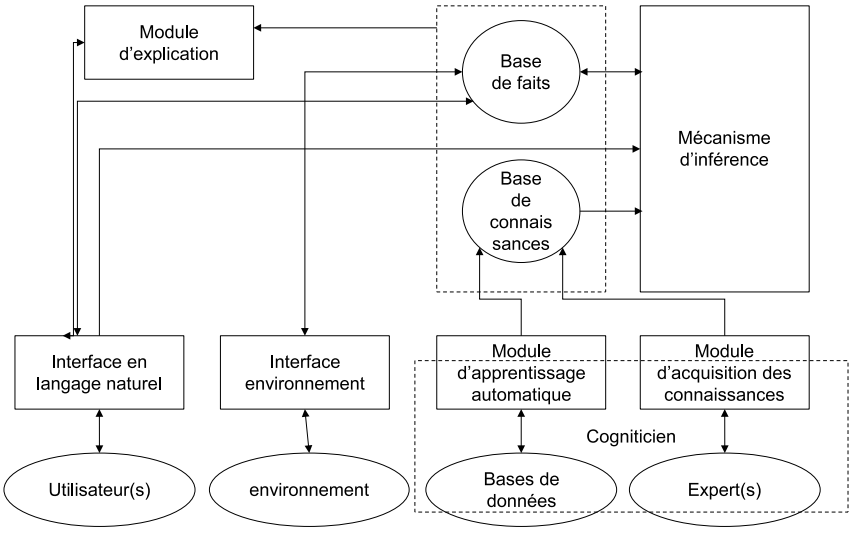


FIGURE 2.22 – Architecture d'un Système à Base de Connaissances (SBC) [Houriez, 1994]

2.5.4 Conclusion sur les modèles, méthodes et architectures orientés vers les systèmes de connaissance

Après avoir abordé les approches classiques du génie logiciel, un ensemble d'adaptations faites sous l'angle des IHM, et les évolutions pour intégrer la réutilisation, les approches de développement de Systèmes à Base de Connaissance ont été présentées. Dans ce domaine, la difficulté de l'extraction de la connaissance a abouti à un processus itératif visant un cycle de vie court. Les méthodes du domaine se distinguent des méthodes précédemment présentées par les aspects spécifiques de la modélisation de la connaissance et la place privilégiée de l'expert. L'architecture orientée, même si elle est centrée sur la base de connaissance, n'a pas omis l'utilisateur. La réutilisation autre que celle des connaissances stockées dans la base n'a pas véritablement été abordée dans ce domaine.

2.6 Modèles, méthodes et architectures de développement de SIAD

Cette section présente les processus de développement, les méthodes de conception et les architectures qui sont dites spécifiques aux SIAD.

2.6.1 Modèles de développement de SIAD

Les modèles présentés dans cette section sont basés sur les cycles classiques provenant du Génie Logiciel. Nous en résumons deux : le premier est influencé par le modèle spirale, le second par le modèle cascade.

2.6.1.1 Démarche itérative

Selon Checroun [Checroun, 1992], Courbon et ses collègues [Courbon et al., 1981] ont été les premiers à proposer une démarche itérative comme une méthodologie basée sur la conception progressive d'un SIAD, en utilisant des cycles aussi courts que possible, où les versions successives du système en construction sont utilisées par l'utilisateur final. Les étapes de cette démarche peuvent être résumées en quatre étapes : (1) identification d'un sous problème important, (2) développement d'un petit système utilisable par le décideur pour évaluer son adéquation au problème posé (maquette) (3) reprise, développement et modification progressive de ce "noyau" pour aboutir au SIAD spécifique à ce problème, (4) évaluation permanente du système.

Ce cycle ayant abouti à la réalisation d'un sous-système satisfaisant, on le reproduit au niveau d'autres sous problèmes, puis au niveau du problème global, de manière à obtenir progressivement l'ensemble des applications et l'outil de développement et de maintenance de l'ensemble.

2.6.1.2 Processus de développement "habituel" d'après Nykänen

Nykänen [Nykanen, 2000] a cité un processus de développement en le considérant comme "classique". Ce cycle de vie comprend les principales phases suivantes :

- La phase de planification intègre l'évaluation des besoins, le diagnostic du problème, la définition des objectifs et des buts du système ;
- La phase de recherche consiste à identifier l'approche utilisée pour adresser les besoins de l'utilisateur et les ressources disponibles et à définir l'environnement du SIAD ;
- La phase de conception comprend la sélection de la meilleure approche de conception, la définition des ressources requises, l'étude de faisabilité ;
- La phase de conception détaillée cible la conception des composants, de la structure, des interfaces, du dialogue, des bases de données et de la gestion des connaissances du système ;
- La phase de construction correspond à l'implémentation technique et à l'intégration ;
- La phase de développement intègre les tests et évaluations, les démonstrations, les entraînements, et le déploiement ;
- La phase de maintenance, de documentation et d'adaptation interviennent pour la gestion de la documentation, pour adapter le système lors d'un changement éventuel d'environnement et permet de le faire évoluer selon les besoins des utilisateurs.

Ce cycle est classique par le contenu des phases. Il parait peut spécifique à un système particulier, tel qu'un SIAD. Les seuls traits caractéristiques concernent (1) les notions de problème qui n'apparaissent pas dans d'autres modèles, (2) la gestion des connaissances mais seulement dans la partie conception détaillée.

2.6.1.3 Conclusion sur les modèles de développement de SIAD

Les deux modèles présentés ici montrent qu'il n'y a pas eu de grandes modifications des cycles du génie logiciel pour la conception de SIAD. Les quelques modifications concernent la terminologie ; on retrouve les notions de problème, sous-problèmes, la notion de décision. On peut également remarquer que le second processus présenté mentionne la gestion des connaissances ce qui confirme notre hypothèse qu'un SIAD doit être capable de gérer celle-ci.

2.6.2 Méthodes d'analyse et de conception de SIAD

Nous n'avons trouvé aucune méthode spécifique à l'analyse et la conception de SIAD. Nos travaux tenteront de pallier ce manque, dans la mesure où le chapitre trois de ce mémoire proposera une telle méthode, tout en y intégrant des notions essentielles provenant des autres domaines parcourus dans ce chapitre deux.

2.6.3 Architectures de SIAD

Pour Bonczek et ses collègues [Bonczek et al., 1981a, Bonczek et al., 1981b] un SIAD est un système logiciel intégrant trois composants : (1) un système de langage, i.e. un mécanisme permettant une communication entre l'utilisateur et les autres composants du SIAD, (2) un système de connaissance, i.e. le référentiel de la connaissance dans le domaine du problème, (3) un système de traitement du problème, i.e. le lien entre les deux autres composants, contenant une ou plusieurs capacités de manipulation du

problème demandé pour la prise de décision. En 1987, Sprague a présenté une architecture similaire, cf. Figure 2.23. On peut dire que cette architecture n'a pas évolué beaucoup car pour Shim et ses collègues [Shim et al., 2002] vingt ans plus tard, les SIAD classiques comprennent toujours trois composants : (1) un composant destiné à la gestion des bases de données, (2) un composant comprenant des fonctions de modélisation puissantes accédées par système de gestion de modèle, (3) un composant de conception d'interfaces utilisateur simples et puissantes qui permettent des requêtes, des rapports et des fonctions graphiques.

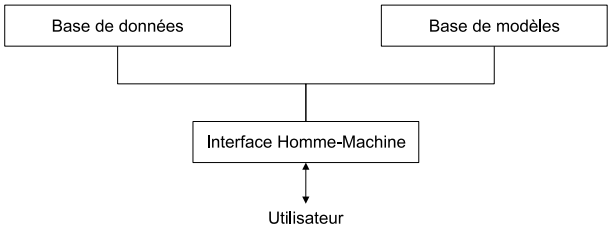


FIGURE 2.23 – Architecture générale d'un SIAD [Sprague, 1987]

Selon Vahidov et ses collègues [Vahidov and Kersten, 2004], le noyau du SIAD est composé des entités habituelles : base de données, base de modèles et base de connaissance correspondant à un domaine de problème, mais aussi d'un composant appelé manager. Ce manager permet au SIAD d'être actif et autonome pour la réalisation de certaines tâches. Les autres entités montrées en Figure 2.24 telles que le "Sensors" (traduit par détecteur) et l'"Effectors" (traduit par effecteur) sont plus spécifiques au type de système que traite l'auteur. Le détecteur capture les données pertinentes au domaine du problème à partir de sources variées. Tandis que les effecteurs sont les outils utilisés par le système de décision pour envoyer les signaux à l'environnement du problème. On remarque simplement que la structure des SIAD correspondant au noyau dans cet exemple n'a pas évolué durant toutes ces années.

2.6.4 Conclusion sur les modèles, méthodes et architectures spécifiques au développement de SIAD

Cette section a présenté quelques démarches de développement représentatives orientées vers les SIAD. On peut remarquer que les démarches rapportées sont très générales. Elles traitent la conception de SIAD sans réellement prendre en compte les spécificités de ce type de systèmes. Les quelques spécifications qui peuvent être citées concernent la terminologie qui diffère par rapport aux méthodes classiques du génie logiciel. D'autres démarches beaucoup plus spécifiques existent mais elles sont tellement spécifiques à un type de SIAD particulier qu'elles n'apportent souvent pas beaucoup d'intérêt à ces travaux.

2.7 Conclusion

Les modèles classiques du génie logiciel ont progressivement évolué au cours du temps. Ils ont également servi de base aux approches spécifiques à d'autres domaines et ont subi des évolutions différentes

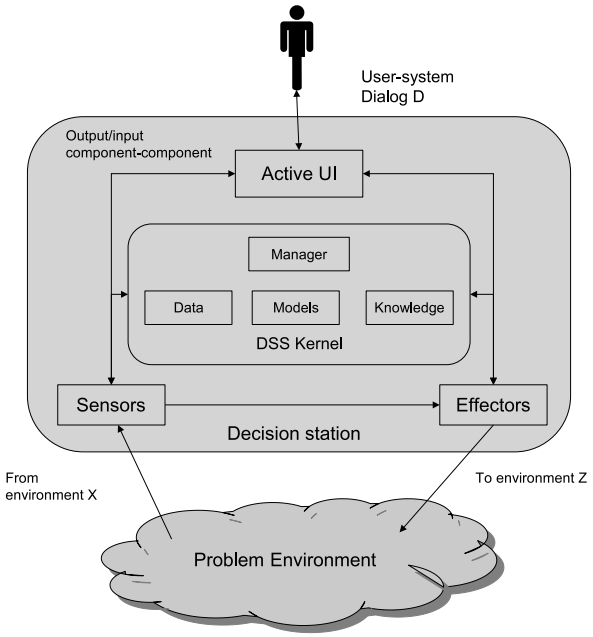


FIGURE 2.24 – Architecture générique pour un SIAD [Vahidov and Kersten, 2004]

par domaine. Ce chapitre a montré comment chaque domaine a adapté les modèles à ses contraintes. Il est possible de s'inspirer du modèle de Curtis et Hefley (cf. §2.3.1.2) pour mettre en parallèle les spécificités de chacun des domaines par rapport aux phases classiques de développement du Génie Logiciel. Nous avons adapté ce modèle pour y intégrer les spécificités du développement des systèmes à base de composants et des systèmes de connaissance (cf. Figure 2.25). Le tableau comparatif (Tableau 2.2) recense les modèles présentés dans ce chapitre. Il a pour but de montrer pour chaque modèle si il y a eu prise en compte de l'utilisateur, intégration des notions de réutilisabilité et de gestion des connaissances puisque nous avons défini que les trois étaient nécessaires pour la conception de SIAD. Dans la colonne "prise en compte de l'utilisateur" nous avons utilisé une notation plus complète que OUI, NON. Nous avons ajouté l'étape dans laquelle l'utilisateur a été pris en compte si il ne l'a pas été dans tout le processus. De même, nous avons précisé si une modélisation a été proposée ou non. Dans certains modèles l'utilisateur peut être pris en compte mais ce n'est pas stipulé clairement dans ce cas nous utilisons "non mentionné". En conclusion, le Tableau 2.2 montre qu'il n'existe pas un processus pour le développement d'un SIAD tel qu'on l'a défini dans le premier chapitre, permettant de guider un concepteur en visant quatre objectifs : centré utilisateur, axé vers une réutilisation, permettant la gestion des connaissances et gérant un problème de décision.

Le chapitre 3 a pour but de proposer une approche centrée sur la satisfaction de ces quatre principaux objectifs de manière à obtenir un modèle d'analyse et de conception de SIAD qui réunisse les caractéristiques de ces cinq types de systèmes : Systèmes d'information, d'aide à la décision, basés sur la connaissance et à base de composants, centré sur l'utilisateur (dans notre cas le décideur) et interactifs.

Phases de développement	Génie Logiciel	Ingénierie des IHM	Ingénierie des Systèmes à base de Composants	Ingénierie des systèmes de connaissances
Analyse des besoins	Analyse des besoins	Analyse de l'utilisateur et de la tâche	Acquisition des composants de domaine	Caractérisation du problème
Allocation des besoins	Matériel / logiciel	Humain / machine	Système / composants	Expert / Intelligence Artificielle
Conception préliminaire	Conception de l'architecture	Conception du dialogue	Acquisition des composants de conception	Spécification des connaissances
Conception détaillée	Conception logique	Conception des écrans	Acquisition des composants de code	Conception de la base de connaissance
Implémentation	codage	codage	intégration	codage
Tests d'implémentation	Tests unitaires et d'intégration	Tests utilisabilité	Fabrication des composants de code	Tests d'opérationnalité des traitements sur les connaissances
Test du système	Test du système	Observation contextuelle	Fabrication des composants de conception	Comparaison résultats produits / attendus par l'expert
Optimisation	Performance de la machine	Performance humaine	Fabrication des composants d'analyse	Performance du système d'aide

FIGURE 2.25 – Intégration des domaines dans les phases de développement, modèle adapté de Curtis & Hefley [Curtis and Hefley, 1994]

Domaine	**Processus**	**Prise en compte de l'utilisateur**	**Intégrant la réutilisabilité**	**Gestion des connaissances**	**Méthodes associées au processus**
Génie Logiciel	Modèle Cascade	Non Mentionné	NON	NON	MERISE, SADT...
	Modèle V	Non mentionné	NON	NON	OSSAD, MKSM, TOOD
	Modèle spirale	Sans modélisation	NON	NON	RAD, UML...
	Modèle incrémental	Sans modélisation	NON	NON	
	Modèle transformationnel	NON	NON	NON	Méthodes formelles
	Rétro-ingénierie	NON	OUI	NON	
	RUP	En phase finale	OUI	NON	UML, Objectory, RUP
	Extreme Programming	A chaque étape	NON	NON	
IHM	Modèle PRODUSER	Sans modélisation	NON	NON	
	Modèle Valentin et al.	Par l'analyse de l'activité	NON	NON	
	Modèle Curtis & Hefley	OUI	NON	NON	
	Modèle étoile	OUI	NON	NON	
	Modèle Nabla	OUI	NON	NON	
	Modèle en U	OUI	NON		SADT/PETRI
Réutilisation	Modèle en X	Non mentionné	OUI	NON	
	Nabla adapté à la réutilisation	OUI	OUI	NON	
	Modèle en Y	NON	OUI	NON	Symphony
	IPM	NON	OUI	NON	Catalysis
Système de connaissance	MODESTI	OUI	NON	OUI	
SIAD	Conception itérative	NON	NON	NON	
	Processus "habituel"	NON	NON	NON	
Notre Objectif		OUI	OUI	OUI	

TABLEAU 2.2 – Récapitulatif des Processus de développement par domaine éétudiés dans ce chapitre

Chapitre 3

ADESIAD : Approche de Développement d'un SIAD centré décideur, axée vers la réutilisation et la gestion des connaissances

Sommaire

3.1 Introduction

Après avoir analysé les SIAD, nous les avons défini comme étant des systèmes interactifs, permettant une gestion des connaissances, basés sur des composants et intervenant tout au long du processus de décision. Comme on a pu le constater, le génie logiciel a proposé des approches qui ont évolué dans le temps. Ces approches ont servi de base à différents domaines qui les ont adaptées en fonction de leurs besoins spécifiques (par exemple la prise en compte de l'expert est plus marquée dans les systèmes de connaissance, le développement par composants a été intégré pour des besoins de réutilisabilité...). Au cours du temps, chaque domaine a fait évoluer ses propres approches le plus souvent indépendamment des autres domaines ce qui a eu pour conséquence de multiplier les approches. Beaucoup de ces approches se ressemblent mais aucune n'est réellement adaptée au développement de SIAD tel que nous l'avons défini.

Le chapitre 2 a présenté les modèles, méthodes et architectures issus de différents domaines qui nous intéressent. Nous avons souligné que le modèle donne le principe de développement du système dans son ensemble ; tandis que la méthode est plus finement décrite, le plus souvent outillée, pour permettre une réelle mise en œuvre de l'étape ou des étapes du modèle qu'elle recouvre. L'architecture peut être liée au modèle de développement, comme dans RUP (cf. § 2.2.1.6) qui est un processus centré sur l'architecture [Jacobson et al., 1999]. Par exemple, dans le cadre d'un développement par réutilisation, nous retrouverons dans l'architecture du système développé des composants tandis que l'architecture des systèmes à base de connaissance intègre la plupart du temps une base de connaissance et un moteur d'inférence.

Le but de nos travaux est de proposer une approche qui satisfasse le maximum des contraintes de l'ensemble des domaines couverts par les SIAD. Cette approche, intitulée *Approche de DEveloppement de SIAD* (ADESIAD), est spécifique au SIAD, tel que nous l'avons défini, au sens où elle sera centrée sur les décideurs, et par voie de conséquence sur les experts du domaine et le processus de décision. Elle a pour but d'intégrer les spécificités des approches de développement des différents domaines que nous avons décrits et qui sont couverts par les SIAD.

De manière à ce qu'ADESIAD soit la plus complète possible, elle sera composé d'un modèle accompagné d'une méthode et d'une architecture qui intégreront les particularités importantes permettant l'analyse et la conception des SIAD tels que nous les avons définis. La première partie de ce chapitre présentera les particularités des modèles présentés qui devront être intégrées au modèle de développement d'ADESIAD ; la vue globale de ce modèle sera présentée. Ensuite, nous proposerons une méthode d'analyse et de conception possible, s'articulant sur le modèle précédent. Cette méthode sera composée d'étapes issues de méthodes présentées dans le chapitre 2 et d'étapes nouvelles issues de nos travaux. La troisième partie du chapitre concernera l'architecture proposée en concordance avec le modèle et la méthode. Nous comparerons notre approche avec celles des différents domaines, vues dans le chapitre précédent, avant de conclure sur les avantages et les inconvénients qu'elle procure *a priori*.

3.2 ADESIAD : le modèle

ADESIAD propose un modèle pour le développement de SIAD. Ce modèle correspond au cycle de développement du système. Les étapes du cycle de vie précédant ou suivant le cycle de développement telles que la faisabilité (cahier des charges), la recette et l'exploitation et la maintenance ne seront pas traitées en détail. Les principes adoptés dans les modèles étudiés précédemment ainsi que les particularités associées aux domaines qui nous préoccupent seront rappelés succinctement. Ces principes et particularités constitueront la base de création du modèle d'ADESIAD.

3.2.1 Particularités des modèles des domaines concernés

Nous avons pu relever, lors de l'état de l'art des modèles de développement (cf. chapitre 2), des particularités associées à chacun des domaines. Les principales sont synthétisées dans le Tableau 3.1. Le modèle d'ADESIAD a pour objectif de satisfaire le maximum de ces particularités.

3.2.2 ADESIAD : vue globale du modèle

ADESIAD se base sur le modèle en V car c'est un modèle répandu, qui a déjà inspiré l'ensemble des domaines qui nous intéressent et qui intègre des étapes classiques du développement logiciel ; il est donc aisément "aménageable". Pour prendre en compte les principes associés à l'interaction Homme-Machine les particularités du modèle en U [Abed et al., 1991, Lepreux et al., 2001a, Lepreux et al., 2003a] seront intégrées. Il intègre le principe de confrontation de deux modèles, respectivement le *modèle idéal* et le *modèle réel*. Ces principes sont visibles dans la partie "développement de SIAD", "phase descendante de conception" et "phase ascendante d'évaluation" de la Figure 3.1 représentant le modèle global d'ADESIAD. De même, le modèle en étoile servira de base pour placer l'évaluation au centre du processus de développement. Le modèle en X, également basé sur le cycle en V, présenté en §2.4.1.1, qui est orienté sur la réutilisation donnera l'exemple pour intégrer au maximum la gestion des composants ; le résultat est représenté par les deux formes extérieures gauche et droite intitulées "recherche/conception de composants" et "validation des composants" sur la Figure 3.1. Dans le modèle en X, les recherches de composants se font à chaque étape du développement du projet (phase d'acquisition) mais le stockage des composants réutilisables ne se fait qu'en fin de cycle (phase d'archivage). Dans le cas d'un cycle incrémental, il serait intéressant de stocker les éléments réutilisables dès qu'ils ont été validés de manière à en faire profiter les incréments successifs. Le dépôt de composant est représenté sous les parties de recherche et de validation de composant avec lesquelles il est en liaison. La méthode MODESTI (cf. § 2.5.1) est aussi une source d'inspiration pour intégrer les spécificités des modèles de développement de systèmes de connaissances et notamment concernant la place des acteurs dans le développement de système faisant intervenir des experts. Nous appréhenderons cet aspect de manière plus détaillée dans la section suivante.

Le modèle d'ADESIAD s'organise suivant un cycle itératif et incrémental, où les développements d'entités indépendantes peuvent se dérouler en parallèle. Il se fera de manière itérative ; il devra être parcouru plusieurs fois avant d'aboutir à un système complet, accepté et répondant complètement aux

Particularités	**Génie logiciel**	**Domaine des IHM**	**Domaine de la réutilisation**	**Domaine de la gestion des connaissances**	**Domaine des SIAD (modèles actuels)**	**Domaine des SIAD (modèles recherchés)**
Préparation de l'évaluation et de la validation en début de cycle	X					X
Processus itératif	X	X	X	X	X	X
Processus incrémental	X	X	X		X	X
Processus parallèle			X			X
Prototypage	X	X	X	X	X	X
Etablissement des activités pour les différents intervenants, en particulier pour les utilisateurs		X				X
Etablissement des activités pour les différents intervenants, en particulier pour les experts				X		X
Modélisation des activités	X	X		X	X	X
Modélisation de l'utilisateur		X		X	X	X
Modélisation des interfaces homme-machine		X			X	X
Modélisation du système	X	X		X	X	X
Modélisation des démarches de l'expert				X		X
Confrontation des modèles du système faits en début et fin de cycle		X				X
Conception séparée des composants et de l'architecture			X			X
Composants de trois types : métier, de conception et logiciel (de code)			X			X

TABLEAU 3.1 – Synthèse des particularités des modèles par rapport au domaine

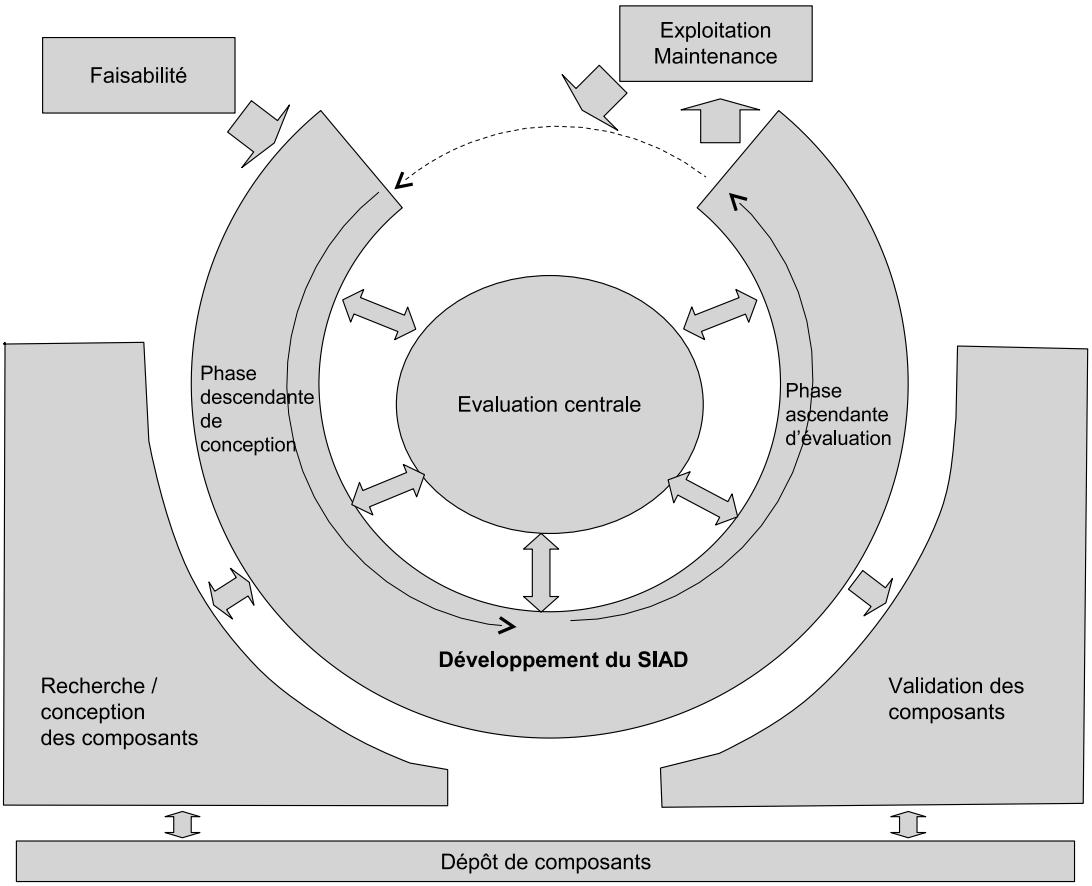


FIGURE 3.1 – ADESIAD : le modèle, vue globale

besoins. Ce cycle est incrémental car il permet l'enrichissement des incréments par adjonction de fonctionnalités notamment grâce à l'intégration progressive de composants. De plus, la possibilité de faire évoluer les incréments une fois qu'ils sont en exploitation permet une évolution guidée par l'expérience opérationnelle. Il est parallèle par la conception séparée des composants et du système qui se déroulent en parallèle. Une des caractéristiques de ce modèle est la séparation de la conception du système et des composants qui l'intégreront. En effet, pour que le système conçu soit évolutif, notamment pour s'adapter à l'utilisateur, et permette la réutilisation, l'approche de conception s'est tout d'abord axée sur la particularité du développement à base de composants. Le principe de ces développements est de concevoir séparément les composants du système. De plus les composants métier renferment des connaissances de type métier ; réutiliser ces composants permet la transmission d'un savoir-faire [Lepreux et al., 2004a].

Le contenu des diverses parties va être détaillé. Le schéma représenté par la Figure 3.2 met l'accent sur les phases de développement du modèle d'ADESIAD. Il montre également les interactions entre les étapes du cycle et l'étape d'évaluation centrale. La phase descendante comprend les étapes classiques des modèles du génie logiciel : l'analyse des besoins, la spécification, la conception, la conception détaillée et le codage. La phase ascendante intègre les évaluations terminales qui sont différentes des évaluations centrales. Ces dernières permettent des retours en arrière plus forts que dans les cycles classiques et

permettent d'impliquer les utilisateurs et experts en plus à chaque fin d'étape pour validation. Ces étapes vont être présentées succinctement dans cette partie et feront l'objet d'une description plus détaillée au cours de la présentation de la méthode.

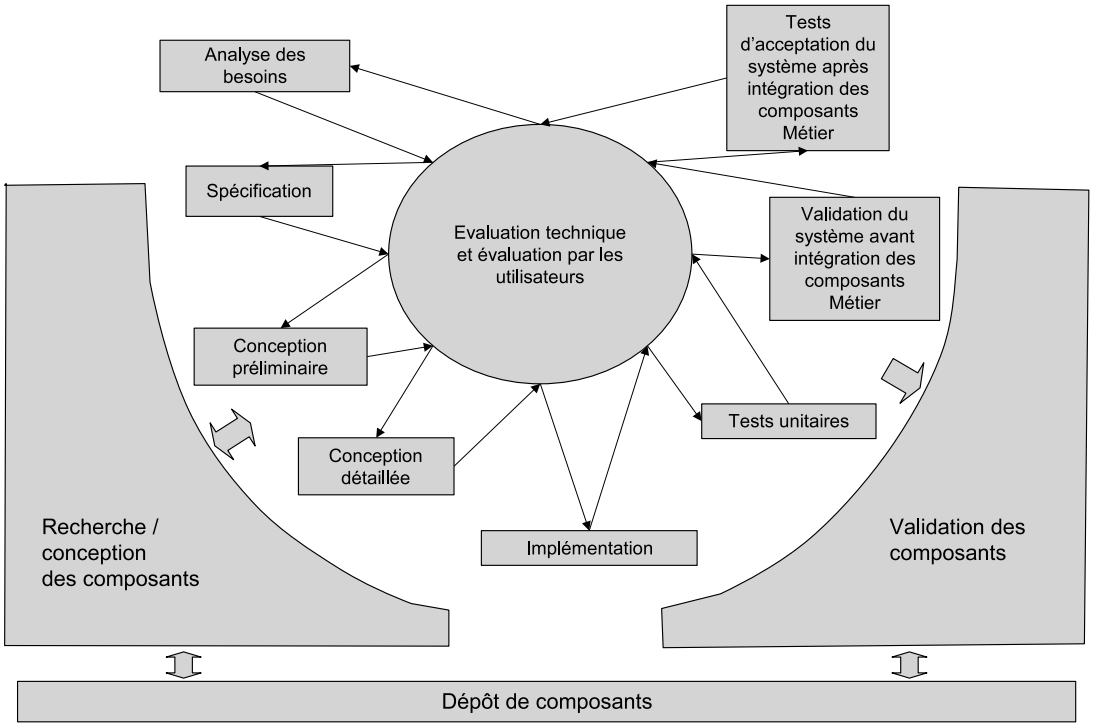


FIGURE 3.2 – ADESIAD : le modèle, vue des étapes de développement du SIAD

3.2.3 ADESIAD : vue détaillée du modèle

La première étape du cycle est celle d'analyse des besoins. Elle consiste en l'analyse des besoins des utilisateurs, aussi bien en termes de besoins fonctionnels qu'en besoins relatifs à l'IHM. Cette étape s'appuie sur l'intégration des connaissances expertes. Elle permet une modélisation du SIAD en partenariat avec les experts et les utilisateurs. Elle intègre également une partie originale, spécifique au SIAD, consistant, à partir des démarches et des connaissances des experts, à décomposer le problème en sous-problèmes pour permettre l'identification des besoins spécifiques au problème. L'analyse de ce problème aboutit à la mise en évidence des besoins en outils nécessaires au traitement du problème et qui seront considérés dans le développement comme des composants métier.

L'étape de spécification consiste à modéliser l'utilisateur et les tâches interactives pour spécifier les interfaces homme-machine, à spécifier les tâches automatiques, le but final étant le maquettage/prototypage du système. L'autre but est de spécifier les composants métier issus de l'étape d'analyse.

Les phases de conception préliminaire et détaillée se concentrent sur la mise en application des spécifications issues des étapes précédentes. Elles consistent à faire des choix en architecture et en conception d'où la mise en évidence des besoins en terme de composants de conception puis par la suite de composants de code.

La dernière étape de la phase descendante (la première de la phase ascendante) est l'étape d'implémentation. Elle a pour but d'intégrer les composants au système (sauf les composants métier) et à coder les parties non réutilisables.

La phase ascendante intègre les évaluations terminales. Elle comprend les tests unitaires, les tests d'intégration et la validation du système (ou plutôt du sous-système puisque le cycle est itératif) avant et après intégration des composants métier, puis la recette du système avant exploitation. Les étapes de validation ont pour but d'analyser la cohérence entre le système (ou sous-système) demandé en phase d'analyse et le système (ou sous-système) obtenu en fin d'itération.

Nous nous sommes inspirés du cycle en étoile qui montre l'importance d'une évaluation centrale. C'est pourquoi les évaluations se retrouvent à la fois dans le disque central du modèle et dans la phase ascendante. Le but de ces deux types d'évaluations est différent. Les évaluations et tests de la partie ascendante sont des évaluations terminales qui arrivent en fin d'itération du cycle. Elles sont utiles aux validations finales. Tandis que les évaluations centrales sont soit techniques soit correspondent à des évaluations par les utilisateurs et les experts. Elles sont des évaluations "moins lourdes" mais qui interviennent dès le début et tout au long de la phase descendante (de conception). Elles permettent la validation des étapes et les retours arrière moins restreints qu'avec le cycle en V "classique". Ces évaluations s'inscrivent dans une démarche participative où les utilisateurs doivent réellement s'impliquer et valider les étapes au fur et à mesure. Si on voulait faire une comparaison de ces deux types d'évaluations dans le système scolaire, on pourrait dire que les évaluations de la phase ascendante représentent les examens et les évaluations centrales le contrôle continu.

Le schéma visible en Figure 3.3 est axé sur les parties concernant la gestion des composants vue du système à concevoir. Il est centré sur la réutilisation de composants existant et la conception des composants qui n'ont pas été trouvés ; cette partie correspond à un développement pour la réutilisation (selon le principe décrit en § 2.4.1). Pour faciliter la réutilisation, le modèle se base sur trois types de composants sur quatre niveaux différents : les composants métier (ou de domaine) au niveau analyse et spécification, les composants de conception au niveau conception et les composants de code au niveau conception détaillée.

Les étapes d'analyse et de spécification permettent de mettre en évidence la spécification des composants métier. Ceux-ci sont recherchés soit en interne soit chez des fournisseurs externes. Cette recherche se fait en deux étapes : la sélection du composant qui est possible grâce aux spécifications et le test. Une fois que les composants sont sélectionnés, ils doivent être évalués selon deux aspects : techniques et non techniques. L'évaluation technique comprend l'intégration, la validation et la vérification [Christiansson et al., 2002]. La validation permet de vérifier que le composant correspond à la demande [12] , tandis que la vérification permet de s'assurer que le composant est correct. Dans le cas où le composant n'a pas été trouvé, il faut le développer ; c'est le développement pour la réutilisation. Ce développement peut être fait en interne à l'entreprise ou en externe. Si le développement se fait en interne, cela demandera des ressources supplémentaires et certainement plus de temps. Dans le cas où le

[12] D'après les définitions de Boehm [Boehm, 1981] : "Validation : Are we building the right product ?" et "Verification : Are we building the product right ?".

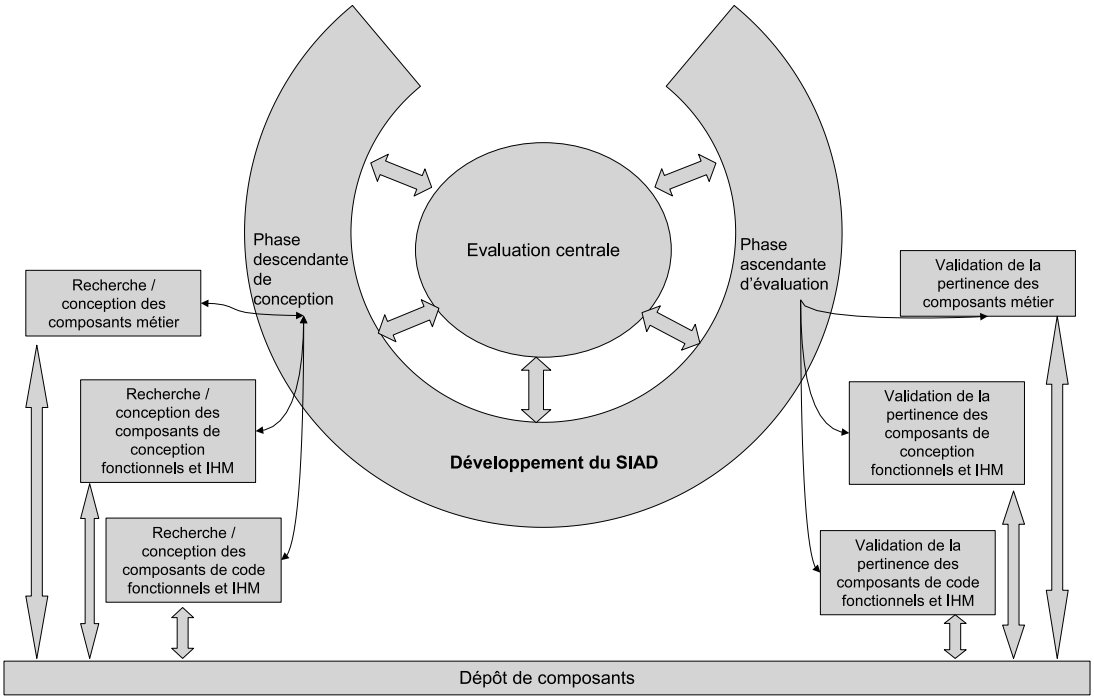


FIGURE 3.3 – ADESIAD : le modèle, vue de la réutilisation

composant doit être développé, il faut regarder s'il est indispensable au fonctionnement du SIAD ou s'il peut être ajouté tardivement, par exemple lors d'une autre itération. Une fois que les composants sont disponibles, ils pourront être intégrés.

De la même façon, l'étape de conception permet de spécifier les composants de conception (d'une part fonctionnels et d'autre part IHM) nécessaires à la conception du SIAD ; ils seront à leur tour recherchés, éventuellement conçus, et intégrés. Ensuite, la phase de conception détaillée met en avant les besoins en terme de composants logiciels qui suivront le même processus de recherche, conception et intégration que les autres types de composants.

La séparation dans le cycle entre le système et les composants qui l'intègrent est explicite (cf. Figure 3.4). Cette séparation permet de concevoir un SIAD indépendamment des composants métier. Dans la phase ascendante des évaluations finales, cette séparation sera encore marquée. En effet, les tests d'implémentation mettront en évidence les défauts des composants de code. Les défauts détectés peuvent être de type fonctionnel, en rapport avec le choix (ce n'est pas le bon composant qui a été choisi), en rapport avec la performance (le composant ne respecte pas son contrat[13] ; il ne correspond pas à la description qu'il donnait)... Les défauts peuvent aussi apparaître après la composition sous la forme de problème de compatibilité. Dans ce cas, la définition du composant à rechercher est revue ou validée et le composant

[13]Le développement contractuel ou à base de contrats développé dans les années 90 et décrit par [Meyer, 2000] consiste à intégrer la notion de contrat entre celui qui utilise le code développé et celui qui le développe : le client doit satisfaire certaines conditions pour utiliser les fonctionnalités du logiciel et sous ces conditions, le fournisseur assure que certaines autres conditions seront remplies après l'exécution [Petit, 2003].

désiré est recherché une nouvelle fois. De même les tests du système avant intégration des composants métier a pour but de valider le SIAD "seul" et notamment d'évaluer les composants de conception tandis que les tests du système après intégration mettront en évidence les défauts des composants métier.

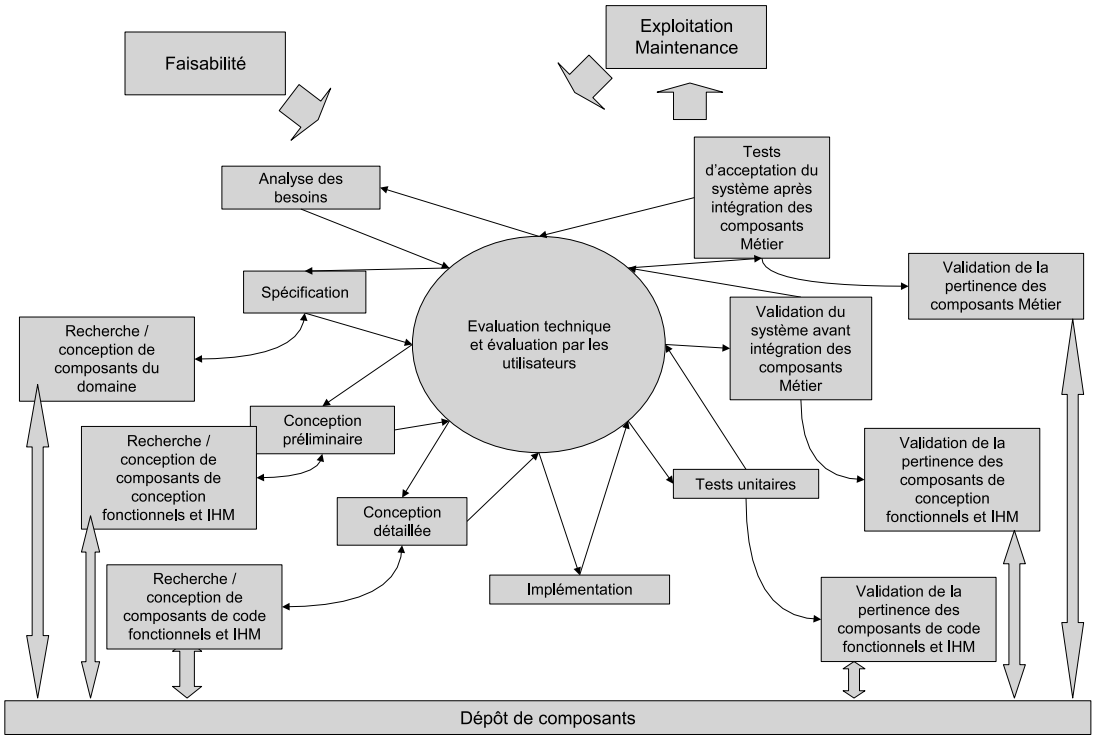


FIGURE 3.4 – ADESIAD : le modèle, vue complète

3.2.4 ADESIAD : les acteurs

En terme d'acteurs agissant dans ce cycle, nos travaux se basent sur ceux de Duribreux-Cocquebert et de Caulier [Duribreux-Cocquebert, 1995, Caulier, 1997], cf. figure 3.5. Ces auteurs, dans leurs travaux concernant la conception de Système à Base de Connaissances, proposent trois groupes de travail dont chacun englobe un ou plusieurs individus et a une mission bien définie :

- Le groupe de pilotage (ou comité directeur) oriente, définit les finalités et arrête des stratégies pour l'ensemble des aspects du projet. Ce groupe a une fonction de contrôle de projet d'un point de vue administratif et a une fonction de coordination d'un point de vue technique.
- Le groupe de développement comporte en principe quatre types de spécialistes : un cogniticien (ou ingénieur de la connaissance) dont les compétences sont proches des sciences humaines, un informaticien qui maîtrise les méthodes, techniques et outils de l'informatique (dite souvent avancée), un ergonome qui analyse et anticipe les modalités d'interaction homme-machine et un bibliothécaire qui gère la rédaction et le classement des documents[14]. Le cogniticien recueille et analyse les connaissances expertes, élabore les modèles. La réalisation est à la charge de l'informaticien qui

[14]Ces documents sont indispensables dans les démarches de qualité.

définit les orientations pour la réalisation physique et concrétise le système en implémentant des modules et en les intégrant. L'ergonome analyse les besoins de l'utilisateur, suit leurs évolutions et gère les adaptations jusqu'à l'exploitation du système. Le bibliothécaire est chargé de fournir les documents aux autres intervenants et de classer les nouveaux documents établis de manière à avoir un suivi et un archivage de toutes les décisions et actions.

– Le groupe d'application englobe l'expert et l'utilisateur. L'expert est le détenteur du savoir alors que l'utilisateur est la personne qui va utiliser ce savoir. Les utilisateurs doivent exprimer leurs attentes et apprécier dans la mesure du possible les incidences des changements que va introduire la mise en place du système.

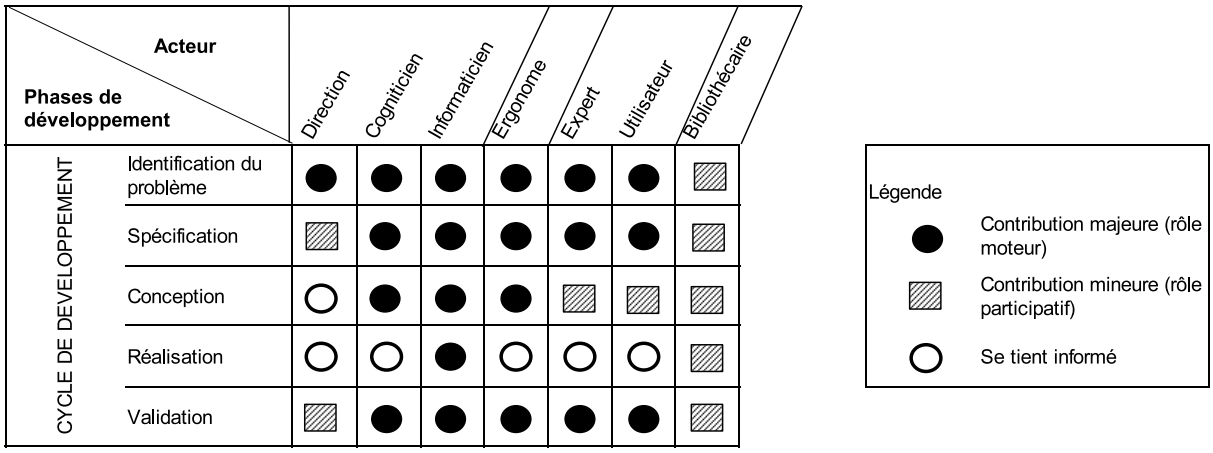


FIGURE 3.5 – Les intervenants et les phases du cycle de développement adapté de [Duribreux-Cocquebert, 1995]

Nos travaux s'appuient sur ceux de [Duribreux-Cocquebert, 1995, Caulier, 1997], qui viennent d'être présentés, car nous sommes globalement d'accord avec ces idées. Le seul point divergeant concerne l'utilisation des connaissances expertes. En effet, dans notre cas le système va exploiter les connaissances de l'expert mais pas tout à fait au même sens qu'un système à base de connaissance pour lequel ces travaux ont été proposés. Pour nous, un expert fournit des connaissances qui sont utilisées pour la conception du système. Ces connaissances sont utiles comme contenu d'une base et comme un ensemble de règles pour la mise au point d'un moteur d'inférence mais aussi pour initier les autres acteurs du développement aux problèmes de décision ; cette initiation permet de les sensibiliser pour mieux structurer l'aide à apporter aux utilisateurs. Dans le cas d'un SIAD, les utilisateurs peuvent être ou non des experts.

Nous avons ajouté aux travaux de Duribreux-Cocquebert et Caulier le rôle du bibliothécaire qui doit être présent lors de chacune des étapes pour présenter les documents des étapes antérieures et doit également archiver tous les documents liés à l'étape en cours.

Nous pouvons remarquer également que dans ces travaux, les auteurs identifient l'étape d'analyse des besoins que l'on trouve habituellement en début de cycle de développement à une étape d'identification du problème. Pour ces auteurs, cette étape comprend l'identification des objectifs, la caractérisation des services ou fonctions et la description de l'environnement opérationnel dans lequel le système sera

intégré.

3.2.5 Conclusion sur le modèle proposé

Le modèle d'ADESIAD que nous avons présenté est innovant par le fait qu'il intègre à la fois les principes généraux du génie logiciel avec certaines particularités de l'interaction homme-machine, des systèmes à base de composants et aussi des systèmes de connaissance. Nous avons sélectionné les caractéristiques communes et les plus marquantes (importantes) de chacun des domaines. La majorité de ces caractéristiques sélectionnées ont pu être intégrées dans le modèle, cf. Tableau 3.2. Les étapes du modèle vont être détaillées dans la méthode supportant ce modèle et présentée dans la partie suivante.

Particularités	**D**omaine des SIAD (modèles recherchés)	**Modèle d'ADESIAD**
Préparation de l'évaluation et de la validation en début de cycle	X	Modèle idéal et réel du système
Processus itératif	X	OUI
Processus incrémental	X	OUI
Processus parallèle	X	OUI
Prototypage	X	Dès la fin de l'étape de spécification
Etablissement des activités pour les différents intervenants, en particulier pour les utilisateurs	X	Dans la partie concernant les acteurs
Etablissement des activités pour les différents intervenants, en particulier pour les experts	X	Dans la partie concernant les acteurs
Modélisation des activités	X	Etape d'analyse
Modélisation de l'utilisateur	X	Etape de spécification
Modélisation des interfaces homme-machine	X	Etape de spécification
Modélisation du système	X	Etape d'analyse
Modélisation de l'expert	X	Etape d'analyse
Confrontation des modèles du système faits en début et fin de cycle	X	Modélisation faite en étape d'analyse confrontée à celle faite en étape ascendante de validation du système sans composant métier
Conception séparée des composants et de l'architecture	X	Oui, deux processus distincts
Composants de trois types : métier, de conception et logiciel (de code)	X	Oui, sur quatre niveaux (analyse, spécification, conception préliminaire et conception détaillée)

TABLEAU 3.2 – Synthèse des particularités des modèles par rapport au domaine

3.3 ADESIAD : la méthode

Cette section a pour but de détailler la méthode proposée dans l'approche ADESIAD. La méthode suit l'ensemble des étapes du modèle proposé auparavant. Pour chacune des étapes, les actions à effectuer et le rôle des acteurs seront définis.

3.3.1 Etape d'analyse

L'étape d'analyse réunit plusieurs activités, cf. figure 3.6[15]. Plusieurs sont originaires du modèle en U et d'autres ont été adaptées du modèle MODESTI. L'analyse du processus de décision et la décomposition du problème en sous-problèmes sont des étapes spécifiques aux SIAD qui sont des parties novatrices et font partie de notre contribution ; ces activités sont grisées sur le diagramme. De manière à familiariser une équipe de conception avec un domaine, autour d'un problème dans le cas des SIAD, l'étape d'analyse débute par le recueil des connaissances des experts et l'analyse de l'existant et du besoin des utilisateurs. Le processus de décision correspond à l'activité à analyser. Une fois que ces analyses aboutissent, le SIAD sera modélisé (par exemple à l'aide d'UML) et les tâches pourront être réparties en deux catégories : tâches automatiques (recherche d'un élément déterminé dans une base) et tâches humaines ou interactives (choix d'un élément à rechercher).

Nous avons vu au cours du premier chapitre que les patrons étaient utilisés pour la gestion des connaissances et comme moyen de réutilisation. Comme nous l'avons mis en évidence dans le chapitre précédent, ils ont également une place importante dans le domaine des interactions homme-machine [Borchers, 2001, Nanard, 2002]. Pour toutes ces raisons, les patrons serviront d'appui à la méthode proposée ; nous verrons dans la suite de quelle manière ils seront mis en oeuvre.

Pour la conception d'un système centré sur l'utilisateur et sur les connaissances expertes, la première activité concerne le recueil des connaissances expertes [Lepreux et al., 2003a]. En effet, celle-ci permet de familiariser l'équipe de conception au domaine et d'intégrer au système les connaissances recueillies. Le diagramme de cas d'utilisation (provenant d'UML) est représenté par la Figure 3.7. Cette étape d'extraction des connaissances comprend deux étapes :

- La première étape permet d'acquérir un niveau de compétence de dialogue avec les experts du domaine. Elle doit commencer par la mise au point d'ontologies du domaine et une acquisition bibliographique des connaissances. Les systèmes similaires développés en industrie ou en laboratoire sont également étudiés, ce qui permet une familiarisation des concepteurs avec les principes de fonctionnement, les méthodes de calcul utilisées et la présentation d'information liés au domaine considéré.
- La seconde étape concerne l'acquisition de connaissances auprès des experts lors de réunions de travail régulières en s'appuyant sur des techniques d'entretien, des analyses de traces écrites et de documents, des études de cas. Cette connaissance disparate est décomposée en concepts et en règles élémentaires. Cette acquisition donne lieu à une modélisation globale des activités en utilisant un langage de modélisation. Il est par exemple possible d'utiliser pour cette modélisation le langage

[15] Le diagramme respecte la notation d'UML.

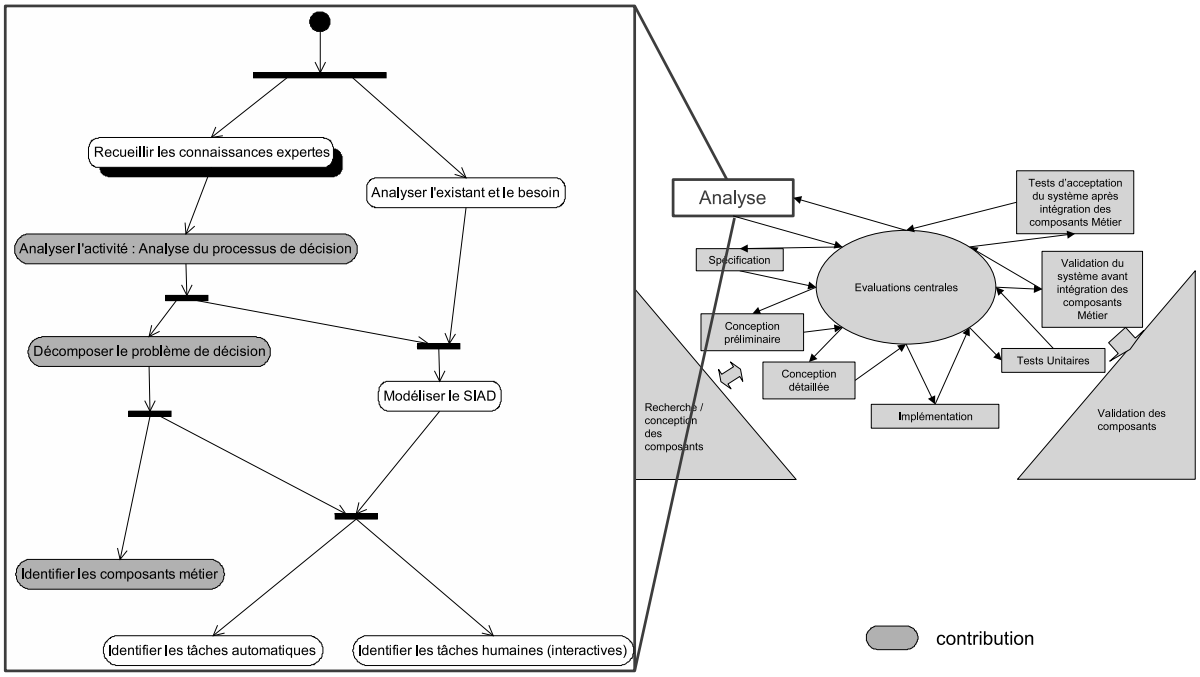


FIGURE 3.6 – Diagramme d'activité de l'analyse des besoins

UML (particulièrement les diagrammes de cas d'utilisation et d'activités). Le bibliothécaire doit archiver les documents utilisés et complétés, les diffuser et les présenter lors des réunions suivantes.

Une fois cette étape avancée, l'analyse de l'existant et du besoin peut débuter. Cette analyse se focalise plus sur la situation de travail que sur les solutions techniques. Elle donne une orientation du travail à réaliser à partir des systèmes existants ou de référence (objectifs, besoins, contraintes, organisation...). Les informations peuvent provenir de techniques de recueil (traces écrites, recueil d'expertise, questionnaires, incidents critiques, arbres de causes, monitoring...) [Cooke, 1994], de procédures écrites et d'experts du domaine. Ces données doivent être transposées, si possible, dans un modèle "source" unique décrivant la tâche et le rôle de l'utilisateur dans son contexte de travail. Les rôles des acteurs du projet informatique sont montrés par un diagramme de cas d'utilisation, cf. Figure 3.8. L'utilisateur décrit sa situation de travail pour mettre en évidence ses besoins. Les experts décrivent également leurs situations de travail en soulignant leur rôle dans celles-ci. Le cogniticien et l'ergonome analysent les données pour extraire les besoins. L'analyste participe également à ces analyses et étudie les systèmes existants. Le bibliothécaire archive l'ensemble des données (brutes et analysées).

L'autre analyse à mener en parallèle concerne celle de l'activité ; elle permet de faire ressortir les besoins fonctionnels et les besoins en IHM. L'activité dans le cadre d'un SIAD correspond au processus de prise de décision. Si on considère le processus de décision de Balasubramanian et al. décrit dans le chapitre 1, on distinguera cinq étapes dans l'activité [Balasubramanian et al., 1999]. Ces étapes seront représentées par la suite sous la forme de sous-tâches de la tâche de résolution d'un problème, cf. Figure 3.14. En effet, pour ces auteurs, le processus de décision débute par une phase de définition du contexte et du but de la décision. Le but parent peut être décomposé en sous-objectifs plus faciles à résoudre. La seconde

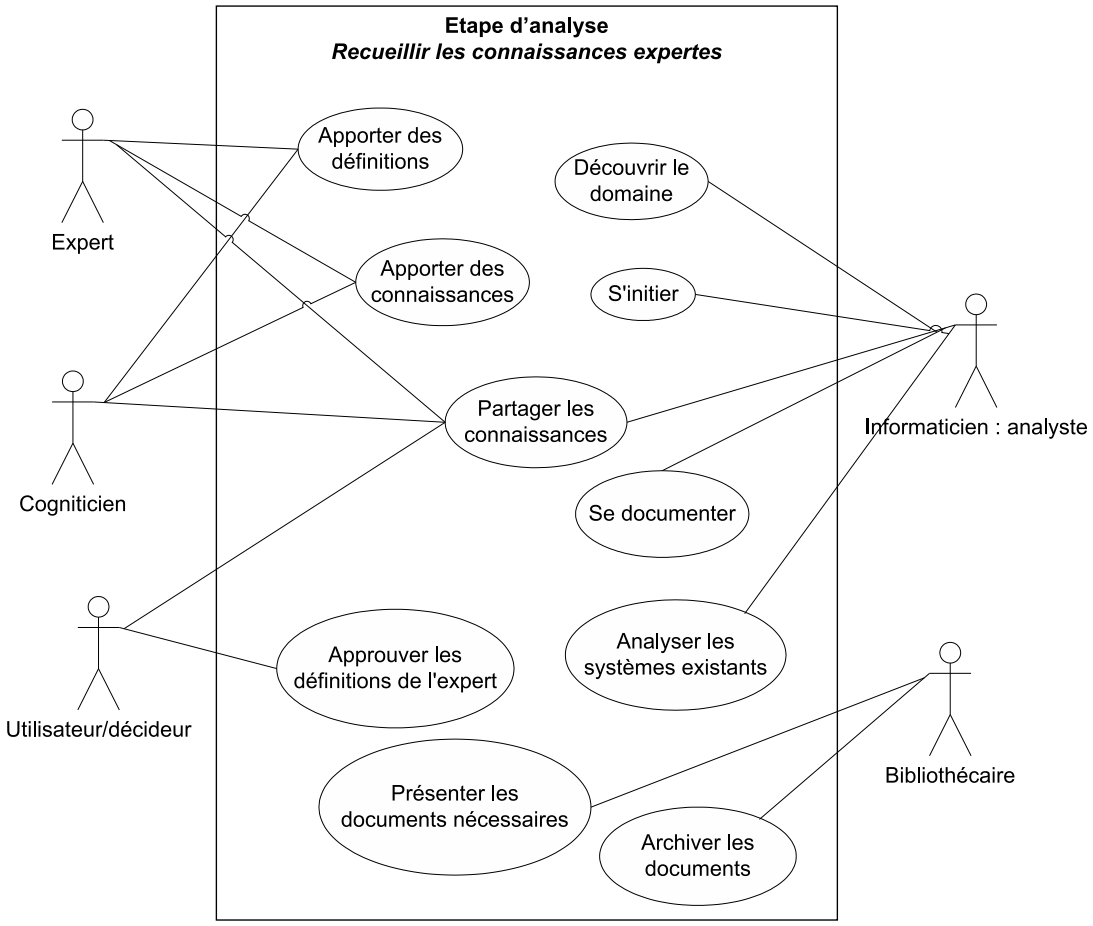


FIGURE 3.7 – Diagramme des cas d'utilisation visant le recueil des connaissances expertes

phase du processus consiste à identifier et générer les options à considérer. Ensuite, le décideur spécifie les facteurs, suppositions, raisons et autres informations pertinentes à considérer. Une étape d'évaluation des options par rapport aux facteurs pertinents, suppositions et autres variables pour la prise de décision est alors mise en œuvre, afin de proposer les résultats analysés et la décision. Ces cinq étapes ne sont pas obligatoirement réalisées par une seule personne. La (les) personne(s) prenant la (les) décision(s) n'étant pas obligatoirement la même (les mêmes) que celle (celles) qui procèdent aux analyses.

Les activités et les acteurs sont présentés en Figure 3.9. Cette analyse de l'activité doit permettre de préciser quel type d'utilisateur serait amené à conduire ces étapes du processus de décision. Pour cela les utilisateurs et décideurs doivent décrire leurs activités, leurs méthodes de travail. A l'aide de ces données, l'ergonome et l'informaticien proposent un processus de décision qui sera validé par l'utilisateur/décideur. L'expert et le cogniticien analysent les activités expertes. Le bibliothécaire doit participer à toutes ces activités pour recueillir les documents, les archiver et les présenter lors de futures réunions.

L'analyse du processus de décision fait ressortir le besoin de décomposer le problème, car c'est la première étape de ce processus. L'objectif est, à partir des énoncés du problème général fournis par les experts, d'aboutir à une décomposition de ce problème pour arriver à des sous-problèmes pour lesquels

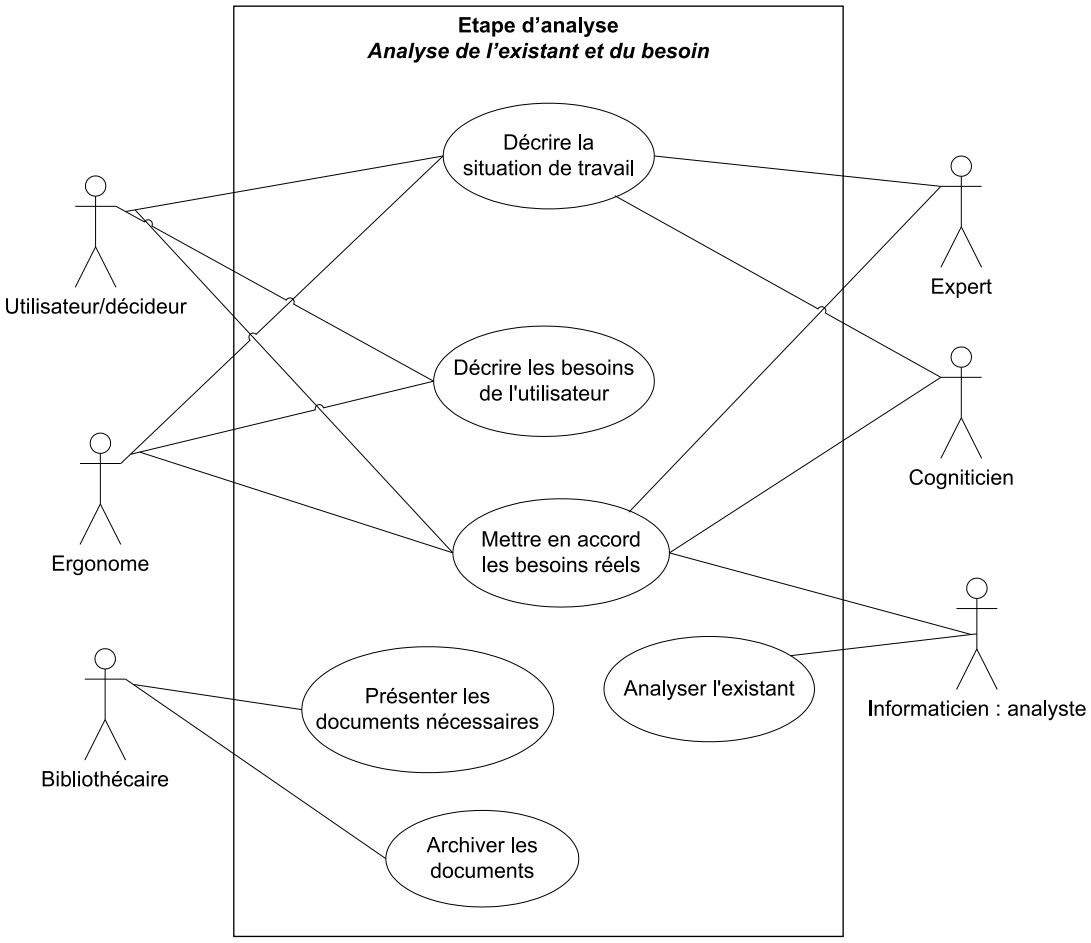


FIGURE 3.8 – Diagramme des cas d'utilisation de l'analyse de l'existant et du besoin

les moyens d'évaluation et les solutions sont connus. A partir de ces sous-problèmes, il est possible de mettre en évidence un certain nombre d'outils qui permettrait d'évaluer ces sous-problèmes. Ces outils sont dans le processus de développement sous la forme de composants métier [Lepreux et al., 2003b]. Le diagramme d'activités correspondant à cette étape est montré en Figure 3.10. Pour mettre en œuvre cette étape, le cogniticien et l'informaticien doivent supporter l'expert afin de sélectionner un problème récurrent du domaine. A partir de ce problème, ils devront faire apparaître les sous-problèmes en proposant une démarche pour résoudre le problème et en énonçant les solutions connues possibles. Cette décomposition est modélisée à l'aide de patrons de problème (définis par la suite). Ils serviront également à structurer, partager et réutiliser cette connaissance liée au problème. Ces patrons permettent de former un diagramme de patron. L'ensemble des documents utilisés et sortis de ces études est archivé par le bibliothécaire.

La Figure 3.11 propose un patron permettant de formaliser un problème [Lepreux, 2003]. Pour proposer ce modèle, nous nous sommes inspirés de ceux présentés dans le catalogue de [Gamma et al., 1994]. De manière à utiliser et partager cette connaissance, chaque problème doit porter un nom, une définition doit être associée pour permettre à d'autres personnes non-initiées de comprendre le problème. Ensuite,

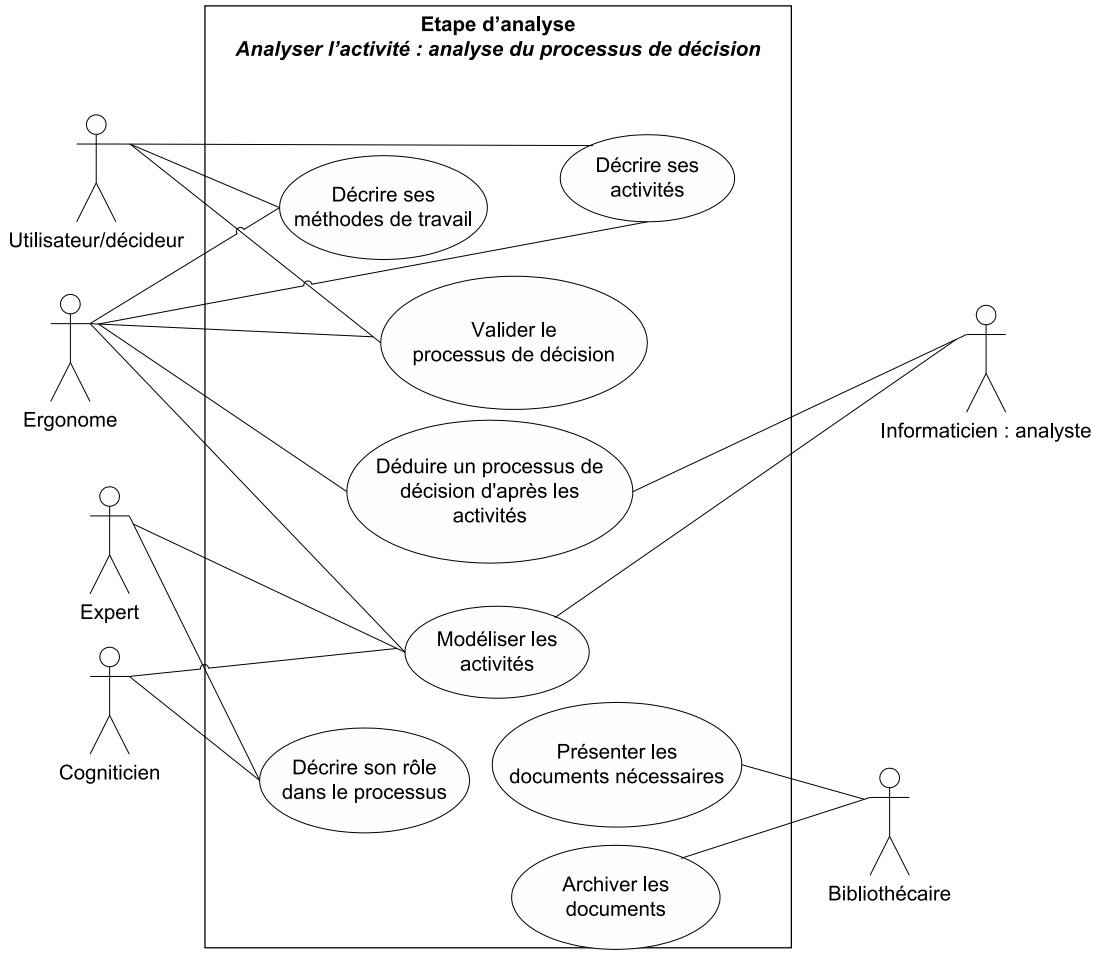


FIGURE 3.9 – Diagramme des cas d'utilisation de l'analyse des activités, du processus de décision

un problème est toujours lié à des entités, celles-ci doivent être précisées dans le champ "attribut". Le but de la formalisation est d'arriver à relier les problèmes entre eux ; pour cela, nous avons ajouté le champ "problèmes père" qui permet de faire le lien entre un sous-problème et son problème père. De la même façon, il y a un champ "sous-problèmes" qui permet de faire le lien dans l'autre sens. Il peut exister d'autres relations entre deux problèmes. Par exemple un problème peut être toujours relié à un autre ; ce champ ne sera peut-être pas utile en fonction du type de problème à traiter et du domaine, il est donc optionnel. Un champ "analyse" permet de renseigner les méthodes de traitement qui permettent d'analyser ou de résoudre le problème. Si des solutions existent, elles apparaissent dans le champ "Solutions".

En résumé, un patron de problème contient les champs suivants :

- "Nom" qui doit être le plus représentatif possible aux yeux des experts et des utilisateurs,
- Une "Définition" qui permet d'expliciter le nom et de capitaliser les idées des experts concernant ce problème,
- Le champ "Attribut" correspond aux notions liées au problème. Par exemple pour un problème d'emploi du temps, le champ attribut pourra contenir : professeurs, salles, horaires, etc.
- Les champs suivants "Problème père", "sous problème" et "relation avec d'autres problèmes" per-

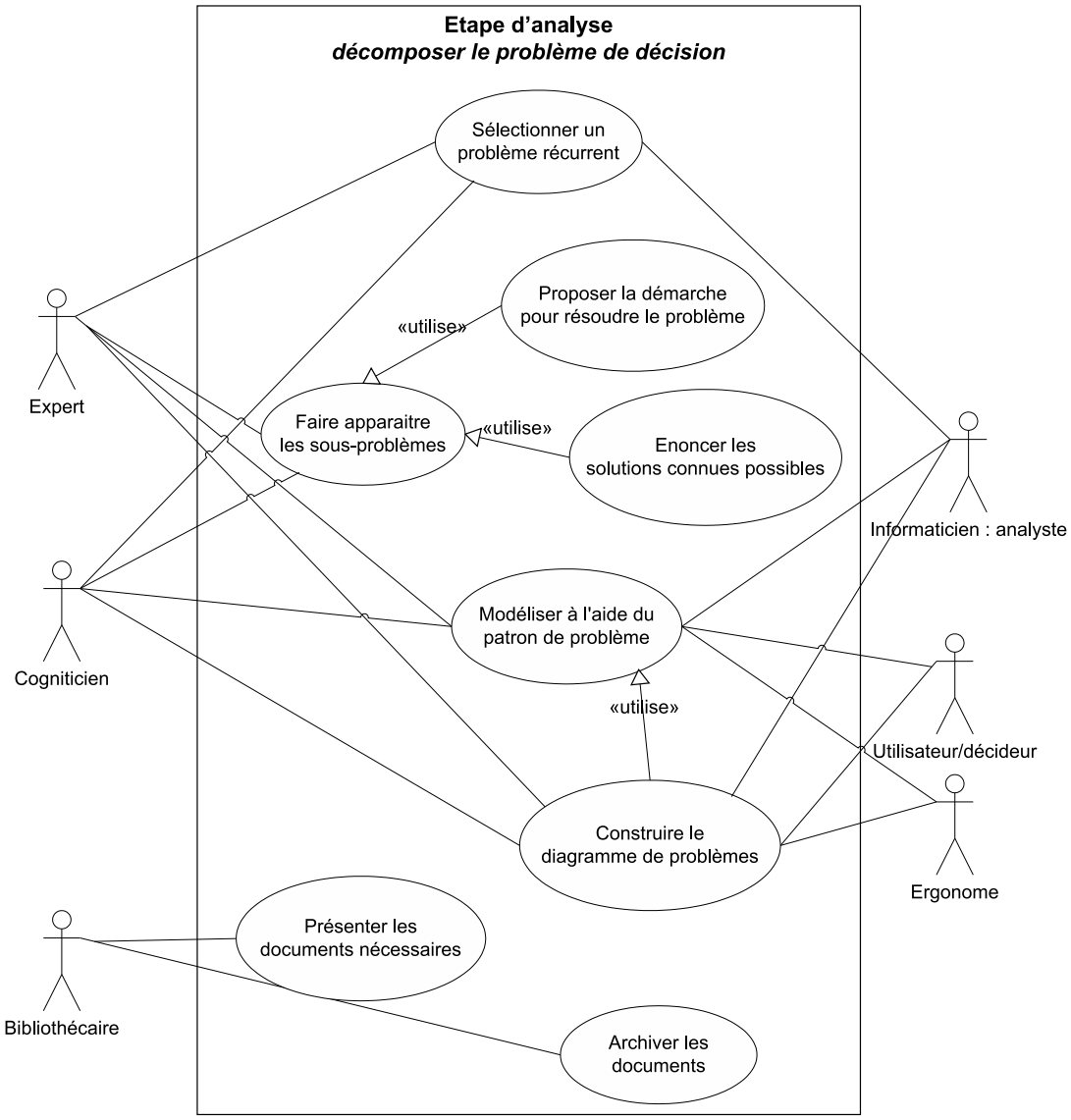


FIGURE 3.10 – Diagramme des cas d'utilisation de la décomposition d'un problème

mettent de relier les problèmes décrits qui sont en relation,

– Le champ "Analyse" permet à l'utilisateur de préciser les actions qui peuvent être menées pour analyser le problème.
– Enfin, le champ "Solutions" est rempli s'il existe une ou plusieurs solution ou un moyen d'évaluer une solution pour ce problème.

Une fois l'ensemble des patrons instanciés relié, on obtient un diagramme comme celui de la Figure 3.12. Par la suite, après validation de ces travaux, ce type de diagramme pourrait être ajouté aux logiciels de modélisation connus. Selon le domaine traité, il est possible d'ajouter des relations "ET" et "OU" entre des sous-problèmes d'un même problème. Des exemples existent dans les travaux concernant les décompositions de tâches [Buisine, 1999]. Nous n'avons volontairement pas souhaité intégrer ces notions à la vision générale. Etablir ces relations permet plus de précision dans le diagramme mais le rend plus

Nom du problème	
Définition du problème	
Attributs	
Problème(s) père	
Sous-problème(s)	
Problèmes connexes	
Analyse	
Solutions	

FIGURE 3.11 – Patron de problème

complexe et difficile à établir. De plus, selon le problème à traiter, il peut ne pas être possible d'utiliser les liaisons "ET" ; utiliser un seul type de relation (par exemple que des "OU") alourdirait inutilement le diagramme.

Cette formalisation des problèmes du domaine associée aux connaissances des experts a plusieurs avantages ; elle permet :

- de confronter la décomposition résultante au fur et à mesure avec les utilisateurs et les experts[16],
- de structurer les connaissances implicites de manière explicite en incitant les experts à remplir les champs et à justifier leur choix, ils expriment ainsi plus facilement leurs connaissances implicites,
- de partager les connaissances,
- de préparer la réutilisation et la maintenance,
- de formaliser l'ensemble du problème de décision sous la forme de diagramme de patron de problème du domaine, cf. Figure 3.12,
- d'identifier les composants métier.

Un exemple de décomposition de problème est donné en Figure 3.13. Il concerne le problème du déplacement. Ce problème est défini par la recherche d'un moyen de déplacement permettant de se rendre à un lieu donné. Ce problème est décomposé en trois sous-problèmes qui sont la recherche d'un moyen de transport privé, semi-privé et public. Le problème de recherche de transport privé a été lui aussi décomposé en deux sous-problèmes qui sont la recherche d'un moyen de transport personnel et non personnel. Tous ses problèmes ont un attribut identique correspondant à la destination, on pourrait ajouter dans certains cas l'origine, notamment dans la recherche de moyen de transport public. A chacun des sous-problèmes une ou plusieurs solutions sont citées telles que la marche à pied, les rollers, le covoiturage, le métro...La partie analyse donne les actions qui pourront être effectuées en fonction du problème. Les trois champs problème père, sous-problèmes et problèmes connexes permettent de naviguer dans l'arborescence.

Les phases 2, 3 et 4 du processus de décision, proposé par Balasubramaninian et ses collègues, consistent à identifier et générer les options, à spécifier les facteurs, suppositions, raisons et autres informations pertinentes, puis à évaluer les options par rapport aux facteurs pertinents, suppositions et autres variables. Nous avons vu que la décomposition du problème permettrait d'aider le décideur lors de la première phase de son processus en lui proposant une décomposition de son problème. Les "Attributs"

[16]L'importance de feedback dans les démarches participatives a été soulignée dans de nombreux travaux.

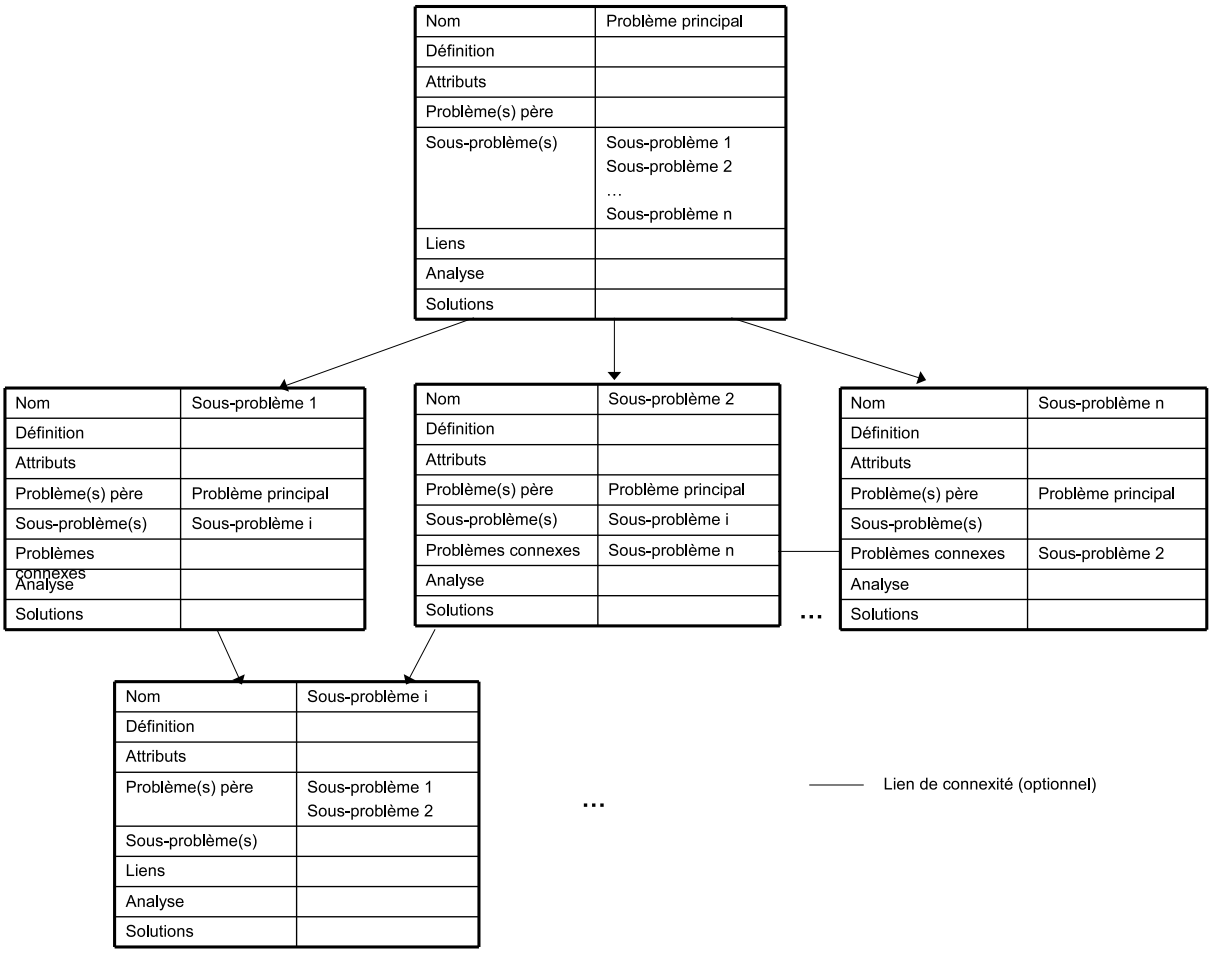


FIGURE 3.12 – Diagramme de patron de problème

et les "Solutions" connus associés aux sous-problèmes vont l'aider dans les phases 2 et 3 où il doit déterminer les options et les facteurs à considérer. La phase 4 consiste à évaluer l'ensemble de ces options. Pour aider l'utilisateur lors de cette partie, il faut lui fournir les moyens d'évaluer ses options. C'est à ce moment que les parties "solutions" et "analyses" des patrons de problèmes vont intervenir. Elles vont permettre d'identifier les outils qui seraient utiles pour réaliser la tâche d'évaluation ; ces outils prendront la forme, pour le SIAD, de composants métier. En conclusion, la décomposition du problème à l'aide des patrons présentés permet d'identifier les composants métier. Ces composants apparaîtront aux utilisateurs tels des outils pour mener des études visant à résoudre leur problème.

Les deux analyses (des besoins et du processus de décision) aboutissent à une modélisation du SIAD, par exemple sous la forme de diagrammes UML ; ce modèle correspond au modèle dit idéal. L'objectif est d'identifier et d'organiser l'ensemble de tâches devant être remplies par le couple Homme-Machine, en fonction des buts fixés par la logique de fonctionnement du futur système. Plusieurs modèles abordent la modélisation de la tâche en se basant sur le principe de décomposition hiérarchique. Ce principe permet d'introduire graduellement des niveaux de détails de plus en plus fin (décomposition de tâches en sous-tâches) en fonction de la structure du système à réaliser [Buisine, 1999], ou modèles "ConcurTask-

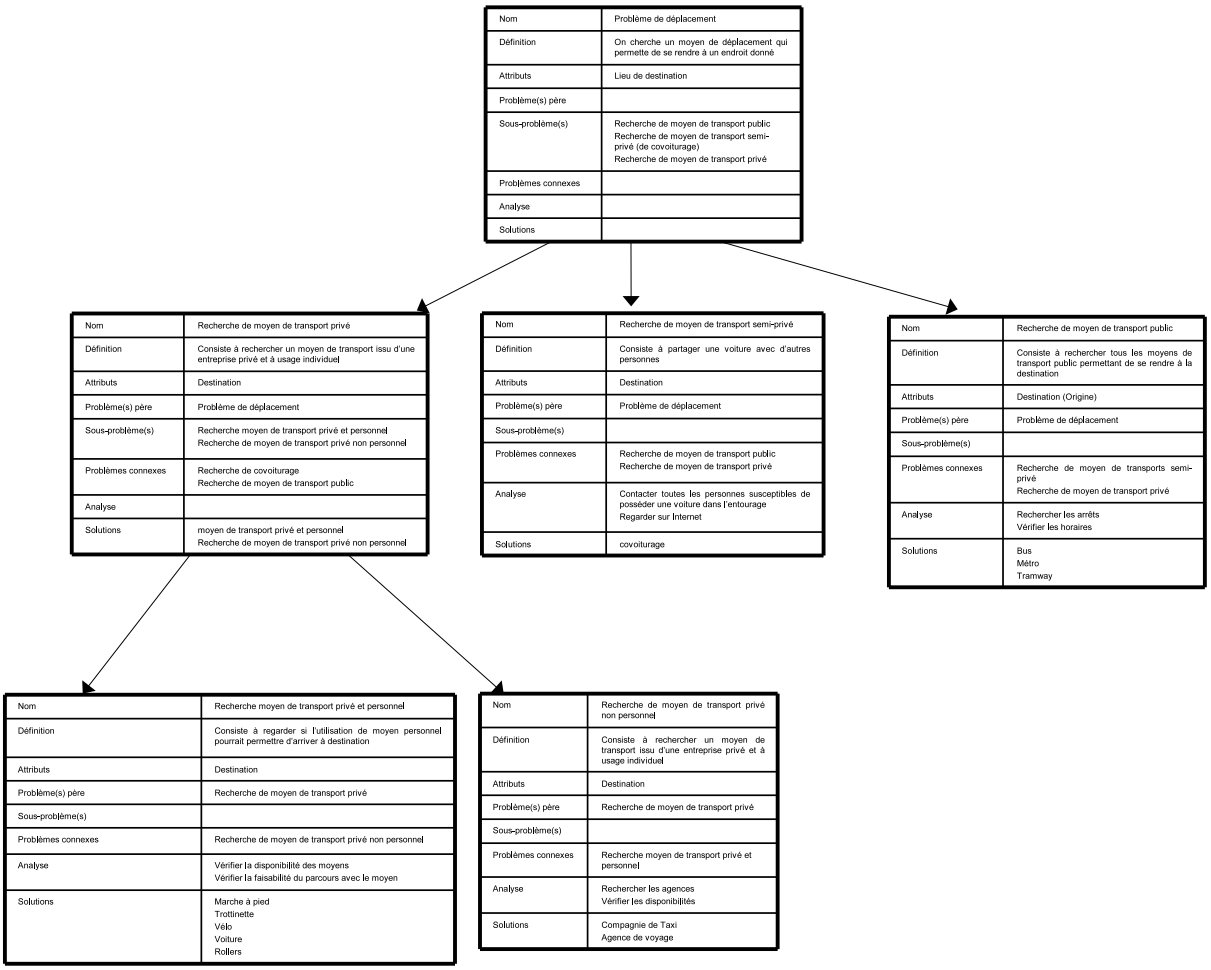


FIGURE 3.13 – Exemple de décomposition d'un problème de déplacement, mise en forme à l'aide du diagramme de patron de problème

Trees" [Paterno, 2001]. Les formalismes de représentation de ces modèles permettent de recueillir les propriétés (attributs) de chaque tâche et les relations qu'elles entretiennent exprimant ainsi la dynamique du modèle ; c'est-à-dire les contraintes logique et/ou temporelles. Il est important aussi de représenter les données en liaison avec les tâches afin de relier les données de l'application avec les traitements qu'elles autorisent.

Lors de cette modélisation, la tâche à réaliser lors de l'utilisation d'un SIAD est "résoudre un problème". Celle-ci est décomposée en sous-tâches, en respectant la notation de GLADIS++ [Buisine, 1999], et en se basant sur le processus de décision présenté précédemment, cf. Figure 3.14. La décomposition du problème, faite en parallèle à l'aide des patrons de problème, sera utile pour guider l'utilisateur lors des sous-tâches "analyser les sous-problèmes" et "choisir parmi les sous-problèmes".

La modélisation du SIAD détermine les tâches du système. Ces tâches vont être réparties de manière à identifier les tâches automatiques où l'utilisateur n'intervient pas et les tâches interactives (ou humaines) qui le font intervenir. Cette répartition peut être menée en parallèle à la décomposition des tâches ou après. Le but de cette répartition est de savoir "qui fait quoi ?". Elle se fait par rapport aux caractéris-

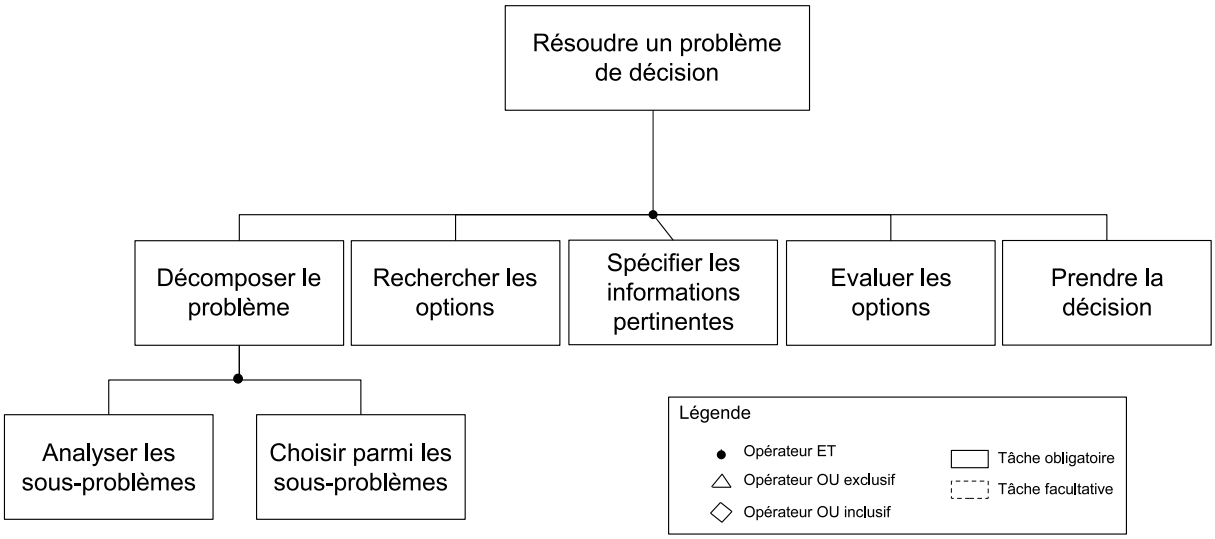


FIGURE 3.14 – Exemple d'arborescence de tâches de "Résoudre un problème" selon [Buisine, 1999]

tiques et aux capacités de traitement de chacun ; il n'y a pas de règles générales à respecter. Les facteurs d'influence seront par exemple les éléments répétitifs, la capacité de mémorisation, les prises de décision, les erreurs et les recouvrements d'erreurs, la rapidité des traitements, etc. Les tâches interactives seront regroupées et restructurées visant à définir le découpage des vues d'affichage en fonction des stratégies cognitives des utilisateurs. Celles-ci mettront en œuvre nécessairement des interactions homme-machine.

Une fois que l'ensemble de ces étapes a été validé par les utilisateurs et les experts lors du passage dans l'étape d'évaluation centrale, l'étape de spécification peut débuter.

3.3.2 Etape de spécification

L'étape de spécification se décompose également en plusieurs activités, cf. Figure 3.15. Les composants métier identifiés dans l'étape d'analyse sont spécifiés. Les tâches interactives sont modélisées à partir de l'identification des tâches humaines, faite dans l'étape d'analyse, et la modélisation de l'utilisateur. A partir de ces modélisations, les besoins en assistance sont analysés et permettent de choisir les modules d'aide. Les interfaces homme-machine peuvent ensuite être spécifiées.

A partir de l'identification des composants métier, lorsqu'une solution envisageable à un problème a été détectée dans la phase d'analyse, elle peut être formalisée à l'aide d'un second patron, cf. Figure 3.16. De la même façon, ce patron a été établi sur le modèle des patrons issus du catalogue de [Gamma et al., 1994]. Ce patron permet de récapituler :

- le "nom" qui correspond au composant métier,
- l'"intention" qui correspond à l'action que devra accomplir le composant,
- le "contexte" dans lequel il s'applique et les problèmes qu'il résout,
- les "données d'entrée obligatoires" qui devront être renseignées par l'utilisateur,
- les "données d'entrée facultatives" qui ne sont pas indispensables au fonctionnement de l'outil mais qui peuvent permettre de fournir plus de précision,

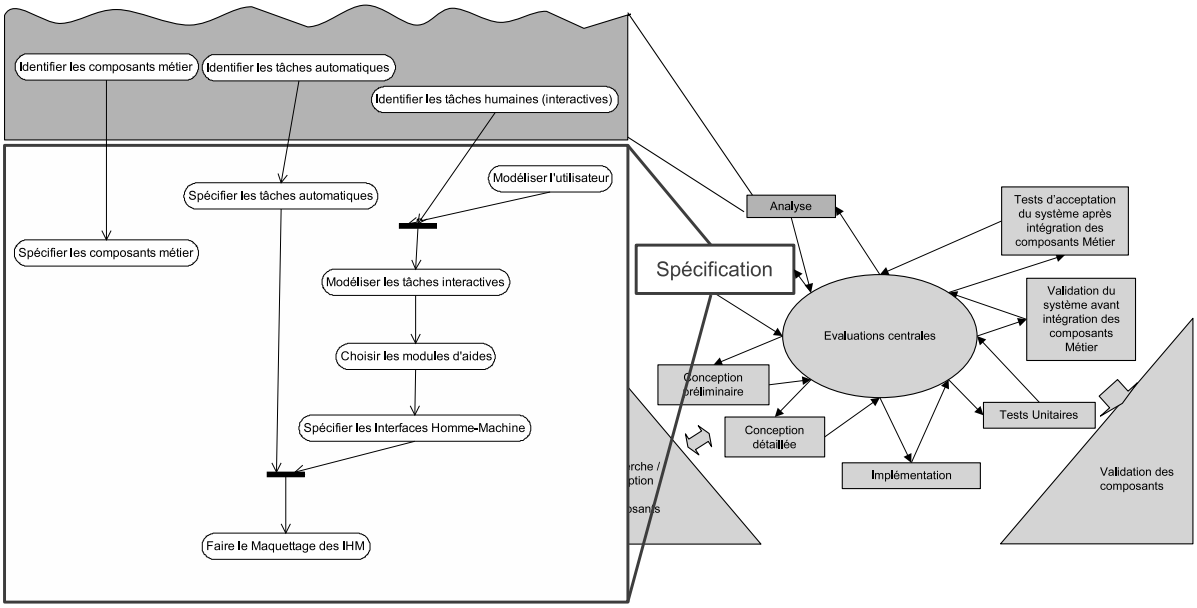


FIGURE 3.15 – Etape de spécification

– les "données de sortie" correspondent au résultat attendu suite à l'utilisation de ce composant ; ce peut être un résultat physique, telle qu'une impression dans notre exemple, ou une réponse plus numérique (pourcentage, chiffre) ou plus complexe (fichier excel, ensemble de données...) ;
– et la partie "solution" permet de fournir une description succincte ou un algorithme plus précis.

Composant = nom_du_composant	
Intention	Action que fait le composant
Contexte	Problème(s) qu'il résoud
Données d'entrée obligatoires	Données d'entrées impératives
Données d'entrée facultatives	Données d'entrées facultatives de précision
Données de sortie	Type de résultat
Solution	Algorithme ou modélisation

FIGURE 3.16 – Patron de composant

Les deux premiers champs sont indispensables à la compréhension et à la recherche du composant métier. Concernant le champ "contexte", les travaux de [Rolland et al., 2000, Ramadour, 2002] montrent l'importance d'associer un contexte à une solution de problème. Pour Rolland, le contexte est défini comme une paire <situation, intention> tandis que Ramadour présente un composant sous la forme <intention, contexte>. Ceci nous semble insuffisant, nous avons ajouté à ces champs les champs décrivant les données d'entrées et de sorties. Les données d'entrées ont été divisées en deux champs, "données obligatoires" et "données facultatives". Ceci permet de connaître le niveau d'exigence du composant et le niveau de précision possible. Par exemple, un logiciel d'impression a besoin de connaître le document à imprimer, les pages à imprimer et le nombre d'exemplaires, cf. Figure 3.17. Pour une impression particulière, on peut lui préciser le type de papier, un format particulier, le type d'encre à utiliser. Ces informations ne sont pas obligatoirement renseignées par l'utilisateur car des valeurs par défaut existent,

mais elles peuvent être modifiées pour avoir plus de précision, de personnalisation. C'est pourquoi nous avons nommé ces caractéristiques des *données facultatives de précision*.

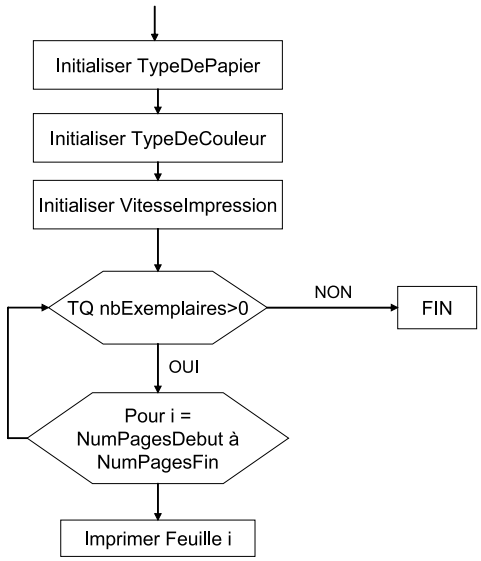
Composant = impression	
Intention	Lance l' impression d'un document sur une imprimante
Contexte	Doit être relié à une imprimante
Données d'entrée obligatoires	Document, pages à imprimer, nombre d'exemplaires
Données d'entrée facultatives	Type de papier, type de couleur, vitesse d'impression
Données de sortie	Feuilles imprimées
Solution	Algorithme d'impression 

FIGURE 3.17 – Exemple d'instanciation du patron de composant au logiciel d'impression

Les autres données importantes sont les *données de sortie*. En effet, l'utilisation d'un outil dépend des contraintes d'entrée comme nous l'avons vu auparavant mais aussi des résultats en sortie. Par exemple, un utilisateur choisira un outil de traitement de texte pour taper une lettre si son objectif est de posséder un document texte imprimable et modifiable. S'il souhaite une lettre manuscrite, il choisira d'utiliser un stylo et un bloc de correspondance. Les résultats fournis font partie des critères de choix de composant. Le dernier champ (*Solution*) permet de donner une modélisation ou un algorithme correspondant au composant.

Cette formalisation des composants métier sous la forme de patrons servira pour la recherche des composants, mais aussi lors de la validation pour faciliter la confrontation entre les besoins en composants du système et les composants sélectionnés. De manière à avoir une vue globale du problème et des outils le supportant, l'ensemble des patrons instanciés peut être visible sur un diagramme, cf. Figure 3.18.

Le modèle utilisateur a pour but de comprendre et d'expliciter des caractéristiques de l'utilisateur et en conséquence de contribuer à une conception de l'interface centrée utilisateur. Ce modèle est important dans une approche de développement de système interactif. Ce modèle peut être différent en fonction de l'application, nous suggérons l'utilisation du modèle proposé par [Robert, 2003], cf. Tableau 3.3. Il a donc trouvé sa place dans cette étape de spécification avant la modélisation des tâches interactives.

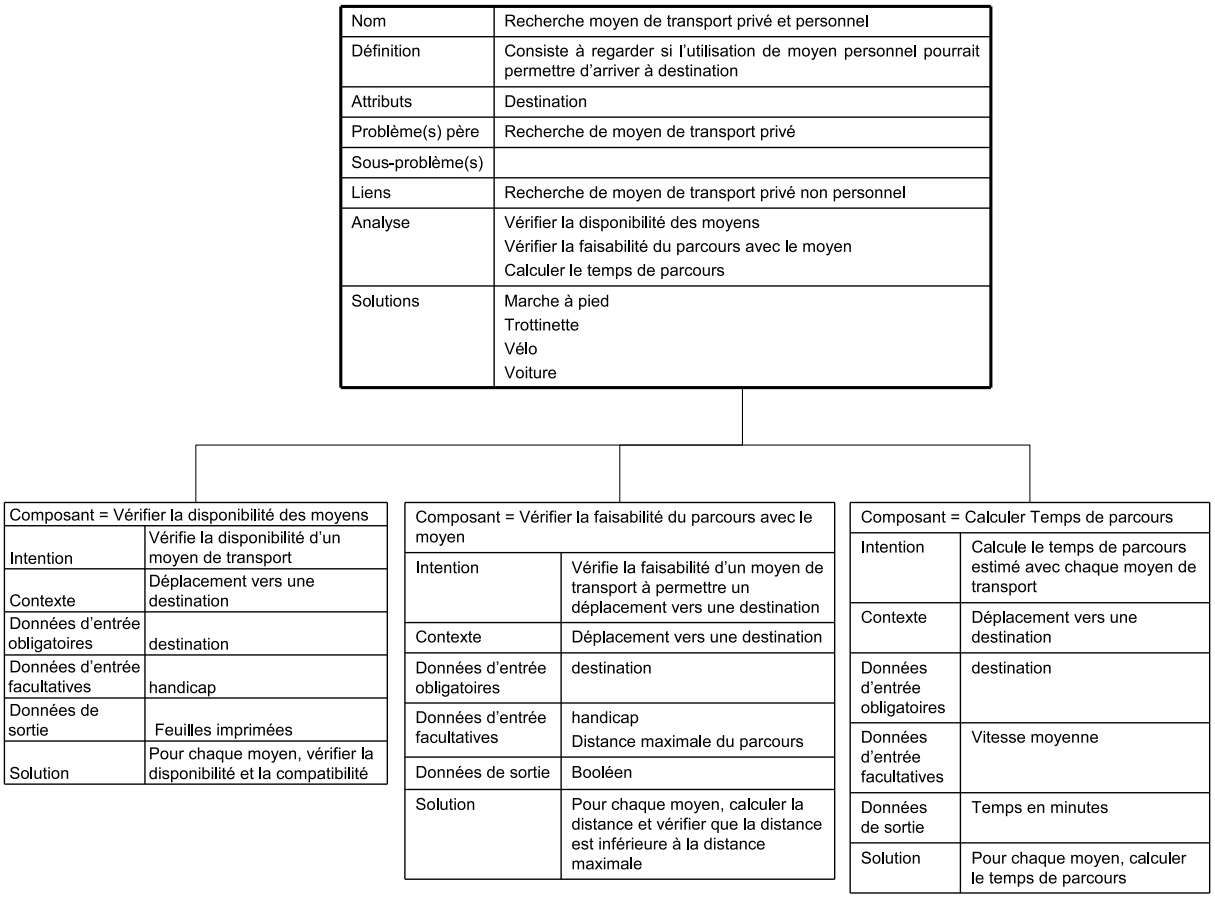


FIGURE 3.18 – Composants associés au sous-problème de recherche d'un moyen privé personnel

De manière générale, la définition du modèle utilisateur fait l'objet de recherches actives au niveau international . Nos travaux ne se sont pas focalisés sur ces recherches que nous ne détaillerons pas.

Le modèle utilisateur associé aux tâches interactives qui ont été identifiées dans l'étape d'analyse, fournit un éclairage pour modéliser les tâches interactives. Cette modélisation se réfère aux procédures composées d'opérations élémentaires que l'utilisateur est supposé effectuer pour réaliser la tâche. Ces procédures formalisent les séquences de dialogue définissant les stratégies et les requêtes de l'utilisateur nécessaires à l'accomplissement du but fixé. L'opération élémentaire physique se traduit par des actions visibles sur l'IHM, comme par exemple la saisie d'une chaîne de caractères, la sélection d'une valeur, etc. Par contre, l'opération élémentaire cognitive est inobservable et représente une activité mentale de type comparaison, choix, décision, ou bien une combinaison de ces trois unités d'activité.

La modélisation des tâches interactives et l'analyse du processus de décision permettent le choix des modules d'aides. Les besoins en outils (fonctions) d'assistance, peuvent prendre la forme d'outils d'aide à la décision (filtrage d'alarmes, diagnostic, planification, etc.). Pour cela, la modélisation des différents scénarios possibles permet de mettre en évidence ceux qui sont inintéressants ("stériles", qui n'aboutissent à rien) ou désastreux (par exemple : perte de données). Cette modélisation permet la mise en place d'aides appropriées de manière à ce que les scénarios ne permettant pas à l'utilisateur de réaliser

sa tâche soient écartés.

A partir du choix des modules d'aides, il devient possible de spécifier l'interface homme-machine. Cette spécification a pour objet l'analyse et la définition du comportement de l'interface. Elle se démarque de l'activité de spécification en génie logiciel classique du fait que les interactions décrites dans la spécification se focalisent sur les relations entre l'utilisateur et le système interactif. Il s'agit de recenser rigoureusement les besoins ergonomiques et techniques, puis de définir le nombre d'écrans à utiliser, l'enchaînement des vues, les modes de présentation des informations, les modes d'activation des différents outils d'aide, les modalités de dialogue homme-machine, etc. Ce passage doit être aussi conforme, en principe, aux relations temporelles et structurelles du modèle de la tâche (issu de la modélisation des tâches interactives). La dynamique du dialogue devient plus difficile à décrire lorsque l'on offre un maximum de liberté à l'utilisateur qui peut alors déclencher plusieurs fils de dialogue en parallèle (dialogue multi-fils). Ces contraintes exigent de spécifier de façon non ambiguë et cohérente le comportement de l'interaction homme-machine [Lepreux et al., 2003c, Lepreux et al., 2004b].

Une évaluation préliminaire du modèle de la tâche peut s'effectuer à ce niveau. Cette évaluation vise à vérifier s'il y a compatibilité entre le modèle du système et le modèle de la tâche. La vérification consiste à s'assurer que le modèle de la tâche est inclus dans le modèle du système, ce qui prouve que l'utilisateur peut réaliser sa tâche avec le système tel qu'il est défini dans le modèle. Tant que le résultat est jugé non satisfaisant, une modification est introduite au niveau du modèle du système de manière à produire des modèles de tâche en accord avec le modèle du nouveau système. Les évaluations successives doivent aboutir à une stabilisation de la décomposition du problème par les experts car elle sera utilisée dans les phases suivantes pour le premier maquettage des interfaces.

Une fois la spécification des interfaces validée, le maquettage des Interfaces Homme-Machine peut être réalisé avec la collaboration des experts et des utilisateurs. Dans un premier temps le maquettage peut être fait sur un support papier inspiré de croquis des utilisateurs. Une fois accepté, le maquettage est refait "au propre" dessiné avec, par exemple, un éditeur de dessin. Ensuite, il est prototypé pour valider les interactions.

L'étape de spécification est suivie de l'étape de conception préliminaire conduite en parallèle de la recherche des composants métier. Si les composants sont trouvés et après vérification de leur correspondance avec les besoins, ils pourront être intégrés à l'étape de conception. S'ils ne sont pas trouvés, ils devront être réalisés et seront intégrés à la prochaine itération. Cette pratique permet de concevoir des composants indépendamment de l'architecture du système et de paralléliser le développement.

3.3.3 Etape de conception préliminaire

L'étape de conception préliminaire consiste à traiter les spécifications qui ont été validées en regardant "comment" elles doivent être traitées pour être respectées, cf. Figure 3.19. Pour respecter un processus incrémental, les fonctionnalités sont classées par ordre d'importance de manière à être intégrées par ensembles successifs. Cette classification se fait avec la participation des utilisateurs. Par exemple, pour un SIAD nous avons vu qu'il y avait une tâche de présentation des outils, une d'utilisation des outils et une de synthèse des résultats. Nous pourrons considérer que cela fait trois ensembles. L'architecture du

Données socio- démographiques : • Age, sexe, langue, situation géographique…
Déficiences : • Physiques : membres supérieurs, membre inférieurs… • Sensorielles : visuelles, auditives…
Données sociologiques : • Historiques, sociales, culturelles, politiques, économiques…
Formation et habiletés : • Niveau de formation : secondaire, collégial, universitaire… • Vitesse de lecture, • Niveau d'habileté au clavier.
Travail et expérience : • Catégorie d'emploi (médecin, pilote, infirmière, technicien dentaire…), • Connaissance de la tâche : novice, intermédiaire, expert… • Lieu de travail : maison, bureau, chez le client, en déplacement…
Connaissance et utilisation du système : • Connaissance du système informatique : novice, intermédiaire, expert… • Connaissance d'autres systèmes semblables à celui que l'on développe, • Type d'utilisation du système : obligatoire, occasionel… • Fréquence d'utilisation du système : faible, moyenne, élevée.
Connaissances informatiques • Générales : faibles, moyennes, élevées, • Système d'exploitation : Unix, Windows, MacOs, Linux… • Outils pour Internet : fureteurs, courriels, engins de recherche… • Famille de logiciels : conception, bureautique, édition, graphisme…
Aspects psychologiques : • Attitudes : positives, négatives, neutres, • Motivation : faible, moyenne, élevée.

TABLEAU 3.3 – Points à documenter dans le profil type des utilisateurs d'un système interactif selon [Robert, 2003]

SIAD permettant une programmation par composant est définie ; ADESIAD propose une architecture qui sera présentée par la suite. Les composants métier qui ont été trouvés peuvent être pris en compte pour être intégrés dans les étapes suivantes. De manière à pouvoir intégrer des composants métier à tout moment du cycle de vie, ils devront tous respecter le même modèle de composants c'est-à-dire posséder les mêmes interfaces (d'autres solutions sont possibles, elles sont abordées dans les perspectives). Le choix et la recherche des composants de conception et des composants de conception spécifiques aux interfaces sont faits pour satisfaire chaque ensemble, sur la base des patrons de conception (design patterns) définis entre autres par Gamma et ses collègues (cf. § 1.4).

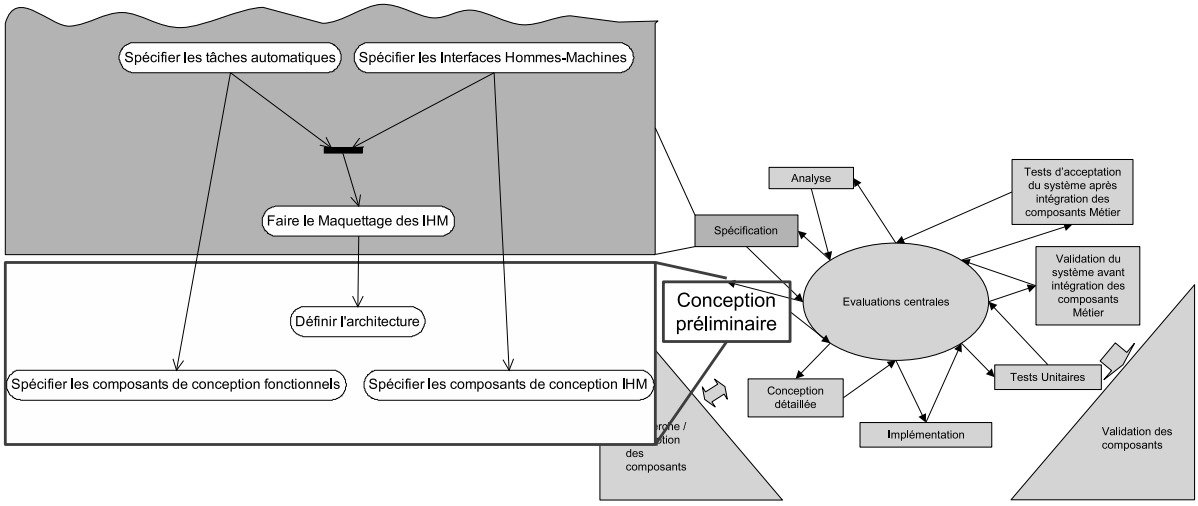


FIGURE 3.19 – Etape de conception préliminaire

3.3.4 Etape de conception détaillée

L'étape de conception détaillée permet de formaliser les spécifications des besoins pour arriver à la définition des algorithmes qui seront nécessaires, cf. Figure 3.20. Ces définitions permettent à leur tour de faire ressortir les besoins du système en terme de composants de code. Les composants de conception qui ont été trouvés sont intégrés. La modélisation du système s'affine avec les nouvelles données.

Cette étape débouche sur la recherche des composants de code fonctionnels mais aussi sur la recherche des composants de code spécifiques aux interfaces homme-machine. Les algorithmes définis à cette étape doivent être validés (tests structurels) avant d'être transmis à l'étape d'implémentation en vue d'être développés.

3.3.5 Etape d'implémentation

Cette étape d'implémentation consiste à intégrer les composants de code fonctionnels et IHM et à coder les parties non réutilisables qu'elles soient fonctionnelles ou IHM, cf. Figure 3.21. Ce codage doit respecter les algorithmes définis à l'étape précédente. L'ensemble des composants de code sont ensuite intégrés [Meinadier, 2002]. Des générateurs de code peuvent être utilisés pour assister l'informaticien

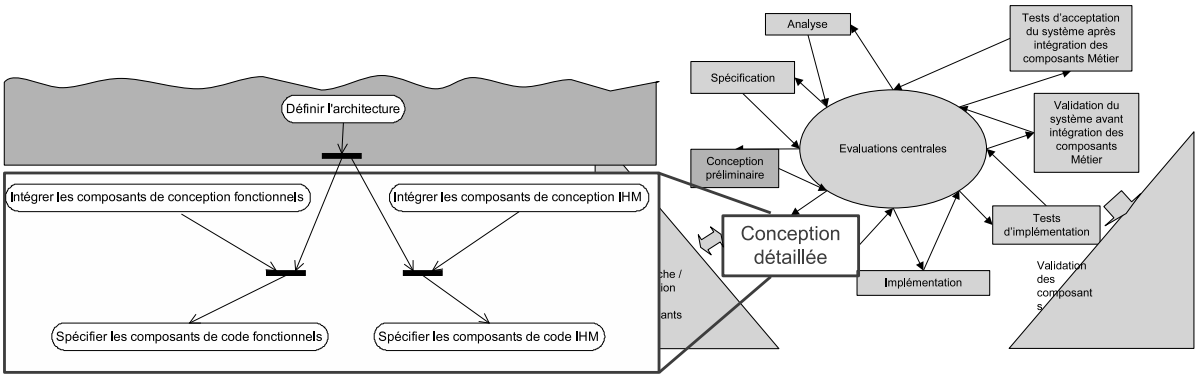


FIGURE 3.20 – Etape de conception détaillée

dans cette étape. Les générateurs de code les plus nombreux se basent sur la modélisation UML pour générer le squelette du code dans des langages objets (Java, c++...). Si la modélisation a été menée à l'aide d'un langage formel, la génération sera plus complète [Petit, 2003].

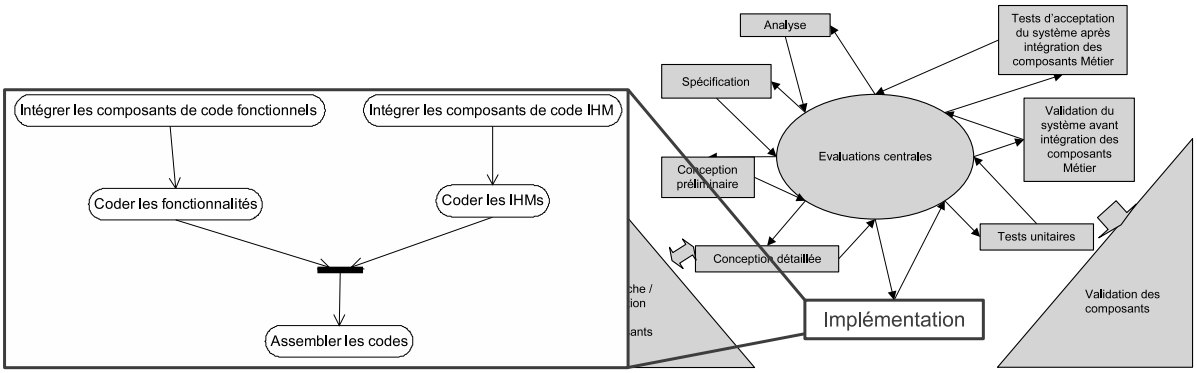


FIGURE 3.21 – Etape d'implémentation

3.3.6 Tests Unitaires

Les tests unitaires consistent à tester les fonctions, qui ont été développées durant l'itération, cf. Figure 3.22. Ils peuvent débuter dès que la fonction a été codée, vérifiée et s'il y a lieu, acceptée, et que le dossier des tests unitaires qui lui est associé est prêt [Xanthakis et al., 2000]. Les travaux à mener lors de la phase de tests unitaires consistent essentiellement à exécuter chacun des tests prévus dans le dossier de tests en rédigeant au fur et à mesure les comptes rendus de résultats. Lorsqu'une non-conformité est détectée, il faut alors engager la procédure de modification prévue qui commence par la rédaction d'une fiche, habituellement appelée fiche d'anomalie ou fiche de non-conformité.

Les tests peuvent être de type fonctionnel ("boîte noire") où la fonction est mise en situation, selon un jeu de test, les données en entrées sont saisies et les sorties résultantes sont comparées avec les sorties attendues. Ils peuvent être de type structurel ("boîte blanche"). Dans ce cas le code est analysé, un graphe de flot est établi et les chemins minimaux sont déterminés. Le jeu de test doit conte-

nir des tests correspondant à chacun de ces chemins, on vérifie alors que les chemins sont corrects. D'autres méthodes de tests existent. Il existe des outils permettant de faire des tests unitaires tel que JUnit [Saumont and Mirecourt, 2003].

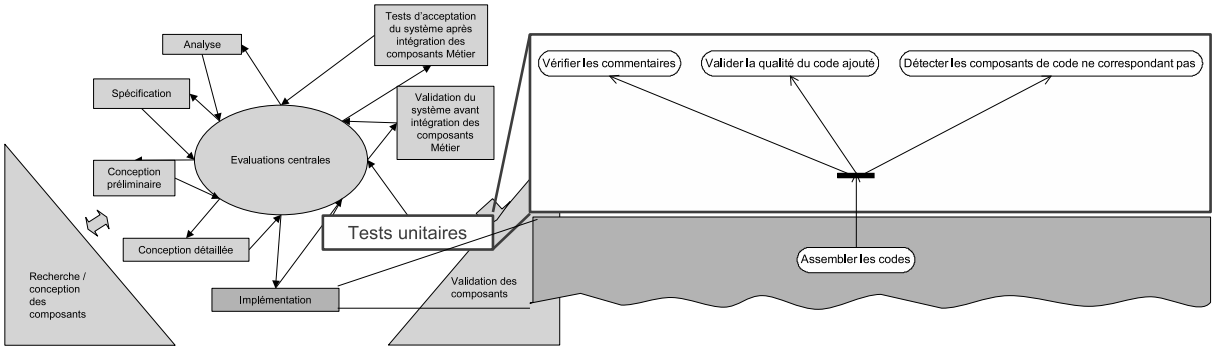


FIGURE 3.22 – Etape de tests unitaires

3.3.7 Validation du système avant intégration des composants métier

Cette étape comprend deux points : les tests d'intégration du système (hors composants métier) et la validation du système sans la prise en compte des composants métier, cf. figure 3.23. N'oublions pas qu'un des objectifs de cette méthode est de concevoir un SIAD indépendamment des outils de mesure correspondant aux composants métier pour évaluer les solutions possibles d'une étude.

Les composants de code vont être progressivement assemblés au code testé lors de l'étape précédente. La démarche d'intégration devra comporter des étapes intermédiaires où des groupes de composants seront testés. Il s'agit de tester les interfaces logicielles entre ces composants, la façon dont chacun d'eux communique et se comporte dans le nouvel environnement. S'il y a des problèmes d'intégration de composants de code, il faudra passer à l'étape de validation des composants logiciels pour distinguer la cause de l'échec et éventuellement pour rechercher un autre composant qui puisse le remplacer. Les interactions entre le matériel et les composants doivent également être vérifiées au cours de cette étape.

La validation du système sans composant métier consiste à vérifier qu'il permet à l'utilisateur de mener les tâches inclues dans son processus de décision, particulièrement en terme d'interaction et de fonctionnalités. Elle permet d'évaluer le respect du processus de décision sans les perturbations possibles induites par des composants métier mal adaptés. Le modèle du sous-système résultant, dit *réel*, est établi ; il est comparé au modèle, dit *idéal*, défini dans l'étape d'analyse. Cette comparaison permet de valider le sous-système par rapport au besoin défini. On évalue également l'utilité et l'utilisabilité du système sans l'intervention des composants métier.

L'évaluation d'un système homme-machine fait l'objet d'un vaste courant de recherche au niveau international. Elle consiste à s'assurer en particulier que l'utilisateur[17] est capable de réaliser sa tâche (selon les critères liés à l'explication) au moyen de l'interface qui est proposée. Mal pensée, elle peut aboutir à un rejet du système. Deux propriétés sont explorées habituellement dans l'évaluation des

[17] De manière générale, puisque l'évaluation peut concerner un groupe de travail, une organisation.

interfaces homme-machine : l'utilité [18] et l'utilisabilité[19] [Shackel, 1991, Grudin, 1992, Farenc, 1997, Bastien and Scapin, 2001] dont de nombreux auteurs ont donné leur propre définition ou ont caractérisé les attributs de ces propriétés de manière à pouvoir les mesurer [Senach, 1990, Nielsen, 1993, Grislin, 1995].

Dans cette évaluation, on s'intéresse généralement aux performances du système global, selon d'une part les comportements des utilisateurs lors de leur interaction avec le système (par exemple, le temps requis pour l'exécution d'une tâche, l'exactitude du résultat, le nombre et le type des erreurs, les difficultés rencontrées, le jugement de l'utilisateur concernant notamment l'interface de dialogue et les systèmes d'aide éventuels...).

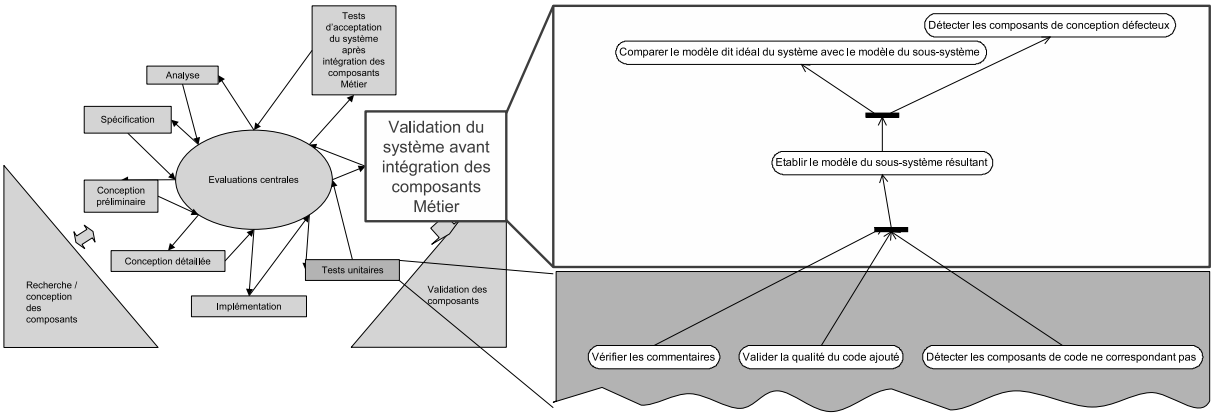


FIGURE 3.23 – Etape de validation du système avant intégration des composants métier

3.3.8 Tests d'acceptation du système après intégration des composants métier

L'évaluation du système complet, c'est-à-dire intégrant les composants métier, permet de vérifier les interactions entre le SIAD et les composants métier, que ceux-ci sont correctement utilisés et permet de détecter les composants défectueux, cf. Figure 3.24. L'utilisateur doit maintenant être capable de réaliser l'ensemble de son processus de décision. Il va évaluer le sous-système. Il valide les besoins en outils ; s'il découvre qu'il lui manque des composants métier pour réaliser la totalité de ses études afin d'aboutir à une décision, il peut le signaler. Il faudra également évaluer l'adéquation entre la présentation des outils et leur contenu réel. Cette évaluation doit assurer que le système permet une aide complète et permet de vérifier la qualité des interactions homme-machine. Certains tests seront rejoués lors de la recette du logiciel qui n'est pas intégrée dans le cycle mais située à la sortie avant livraison du logiciel.

[18] L'utilité concerne l'adéquation qui existe entre les fonctions fournies par le système et celles nécessaires à l'utilisateur pour mener à bien ses tâches qui lui sont assignées. Elle est davantage lié aux fonctionnalités du logiciel.

[19] L'utilisabilité rend compte de la qualité de l'interaction homme-machine, en termes de facilité d'apprentissage et d'utilisation, ainsi que de la documentation [Grislin and Kolski, 1996]. Scapin et Bastien [Scapin and Bastien, 1997] préfèrent utiliser le terme de Qualité Ergonomique des logiciels interactifs pour bien montrer que l'intérêt se porte autant aux fonctionnalités du logiciel qu'à son interface.

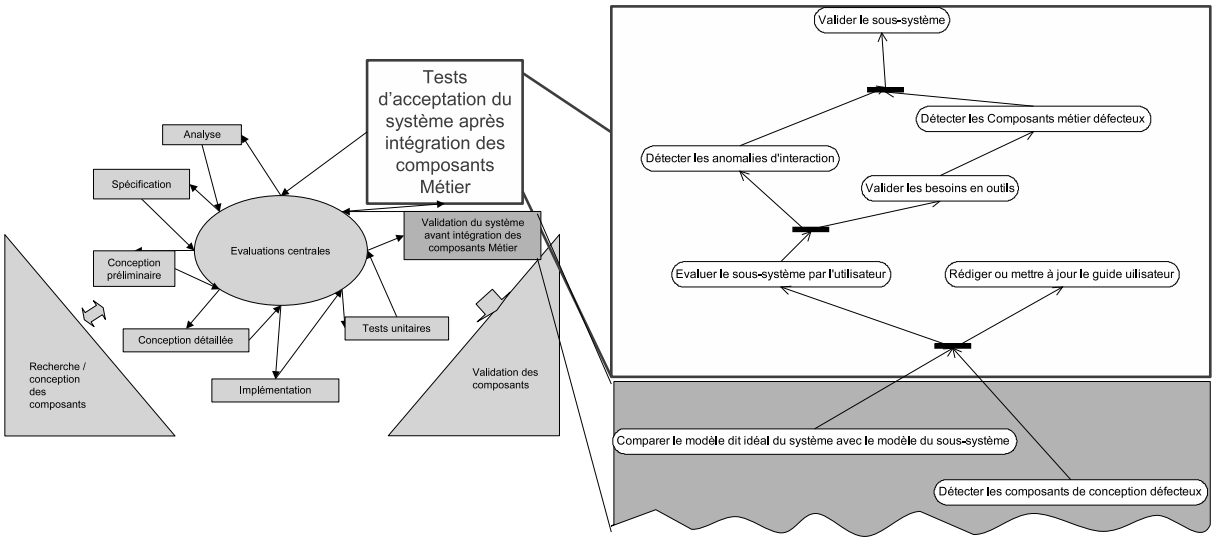


FIGURE 3.24 – Etape de tests du système après intégration des composants métier

3.3.9 Evaluations centrales

Cette étape se situe tout au long du cycle de vie. En effet, chaque étape doit être validée à la fois par les utilisateurs, les experts et techniquement, cf. Figure 3.25. Cette étape nous a été inspirée par le modèle en étoile [Hix and Hartson, 1993]. Elle permet d'impliquer les utilisateurs et les experts pour accroître leur participation dans le processus. Elle permet également de faire des retours en arrière plus importants que dans le cycle en V puisque cette étape permet de détecter les problèmes et permet de retourner à l'étape ayant posé problème. L'intérêt de ces évaluations est qu'elles permettent de revenir en arrière sans attendre la fin de l'itération. Le processus de développement n'est plus linéaire mais plutôt parallèle (pour les tâches qui peuvent être effectuées en parallèle ce qui est facilité par la programmation par composant et la séparation des fonctionnalités des IHM).

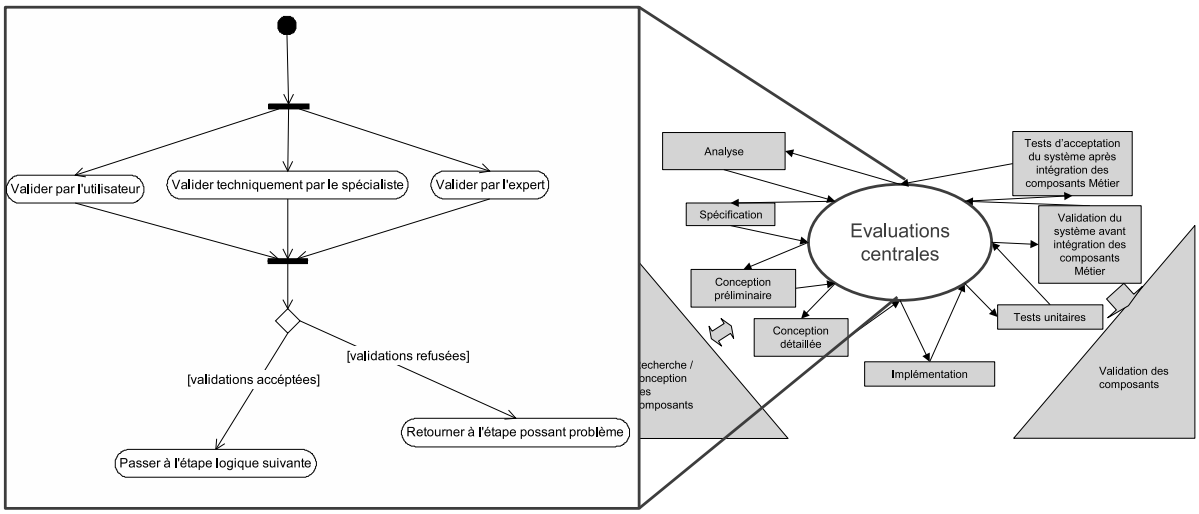


FIGURE 3.25 – Etape d'évaluations centrales

3.3.10 Recherche des composants métier

La recherche des composants métier peut se faire suite à l'étape de spécification, cf. Figure 3.26. En effet, à ce moment les composants métier nécessaires à la prise de décision sont spécifiés à l'aide des patrons de composants. C'est à l'aide de ces patrons que l'informaticien chargé de l'intégration peut chercher les composants dans des bases internes à l'entreprise ou externes. Une fois que les composants se rapprochant au mieux des spécifications sont trouvés, il faut les classer par préférence et vérifier que le composant sélectionné parmi l'ensemble correspond bien à ce qui est recherché (test fonctionnel, vérification des interfaces...). L'utilisateur et l'expert peuvent participer à cette tâche de validation pour s'assurer que le composant sera en concordance avec la définition du besoin.

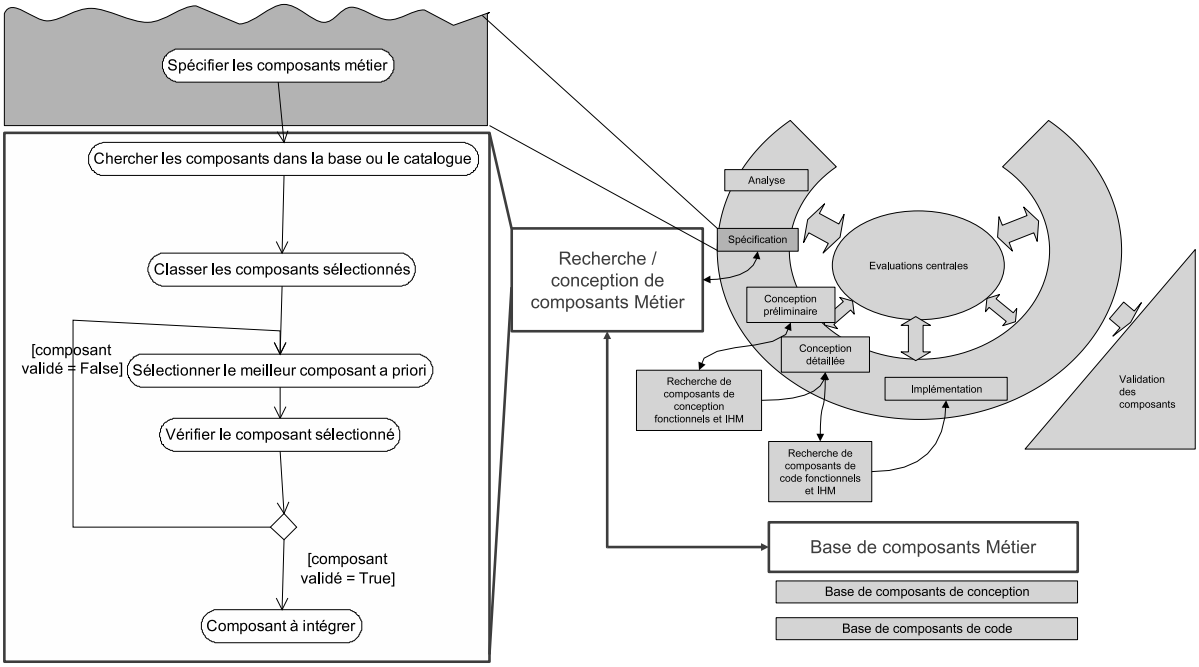


FIGURE 3.26 – Etape de recherche de composants métier

3.3.11 Recherche des composants de conception

L'étape de conception préliminaire met en évidence des problèmes de conception, cf. Figure 3.27. Ceux-ci sont recherchés dans un catalogue de patrons tel que celui de Gamma et ses collègues (cf. § 1.4.2.1) pour trouver des solutions appropriées au problème. A chaque patron de conception est associé un composant de conception. De la même façon que pour les composants métier, plusieurs solutions peuvent apparaître, il faut alors choisir la solution la mieux adaptée *a priori*. Ceci est le travail des informaticiens (architectes, programmeurs...). Si aucune solution n'est trouvée, c'est-à-dire que le catalogue ne contient pas de patron pour ce problème de conception, il n'y a pas de composant de conception pour ce problème. Il faut décider s'il nécessite de créer un nouveau patron/composant. Pour cela, le problème doit être récurrent. Il faut alors établir des solutions, ajouter le patron dans le catalogue et développer le composant de conception.

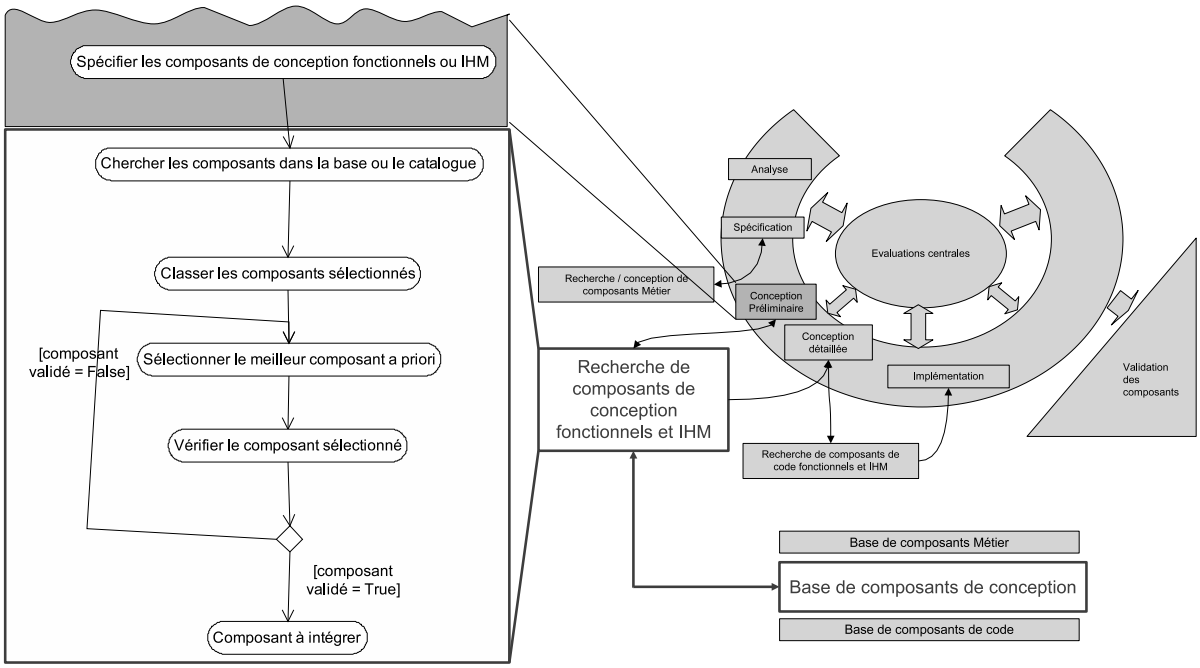


FIGURE 3.27 – Etape de recherche de composants de conception

3.3.12 Recherche des composants de code

La recherche des composants de code se fait entre l'étape de conception détaillée et avant l'étape d'implémentation. Elle permet de sélectionner les composants logiciels qui permettent de réaliser une tâche, cf. Figure 3.28. De la même façon que pour les autres composants, tous les composants de code pouvant correspondre sont recherchés, classés par ordre d'intérêt et un composant est sélectionné dans l'ensemble. Ces composants seront souvent spécifiques à la plateforme de programmation (CORBA, J2EE, .NET). Ils seront intégrés au système via leur interface logicielle. Ils doivent être testés pour vérifier qu'ils réalisent bien la tâche prescrite.

3.3.13 Validation des composants de code

La validation des composants de code se fait suite à l'étape de tests unitaires. Puisque les composants de code ont été testés lors de leur sélection, les défauts qui peuvent être détectés concernent leur intégration, cf. Figure 3.29. Ils peuvent également être détectés dans l'étape de validation du système avant intégration des composants métier. Si le défaut est important, il faut sélectionner un autre composant de code (retour à l'étape de recherche de composant de code). Si le défaut est peu important, il faut regarder si le problème vient du système ou du composant. Si le problème vient du composant, il faut regarder s'il est modifiable (boîte blanche) et dans ce cas, le modifier pour qu'il soit réintégrable, sinon il faut rechercher un autre composant.

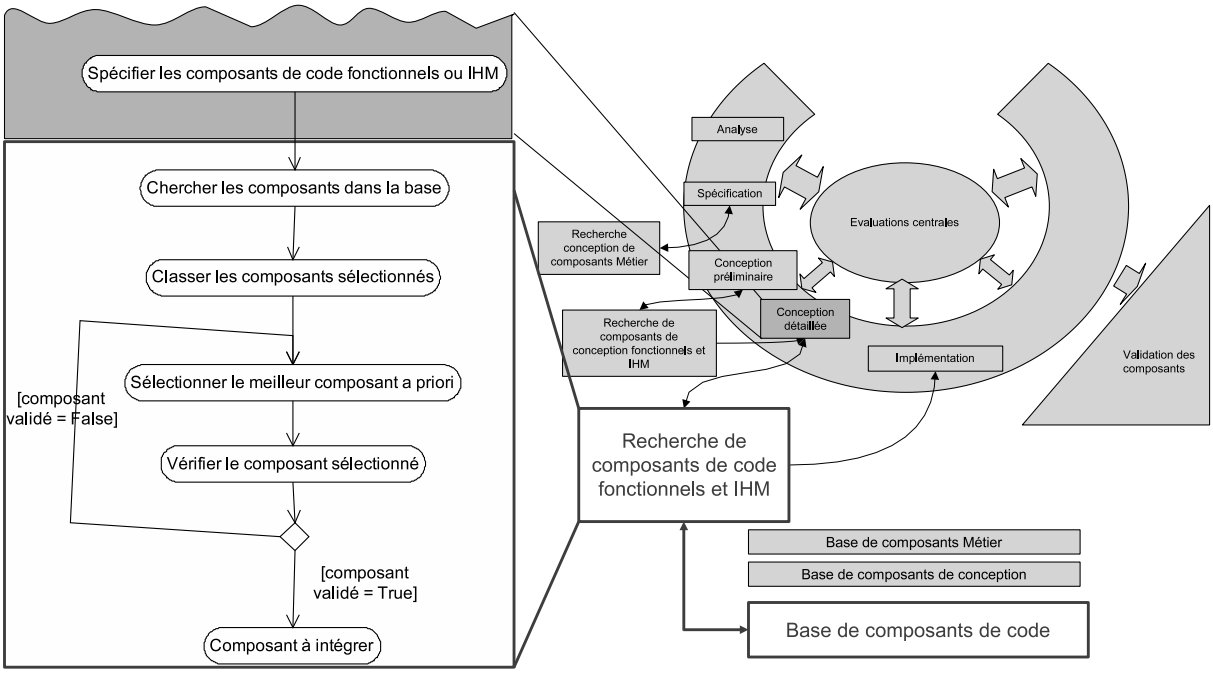


FIGURE 3.28 – Etape de recherche de composants de code

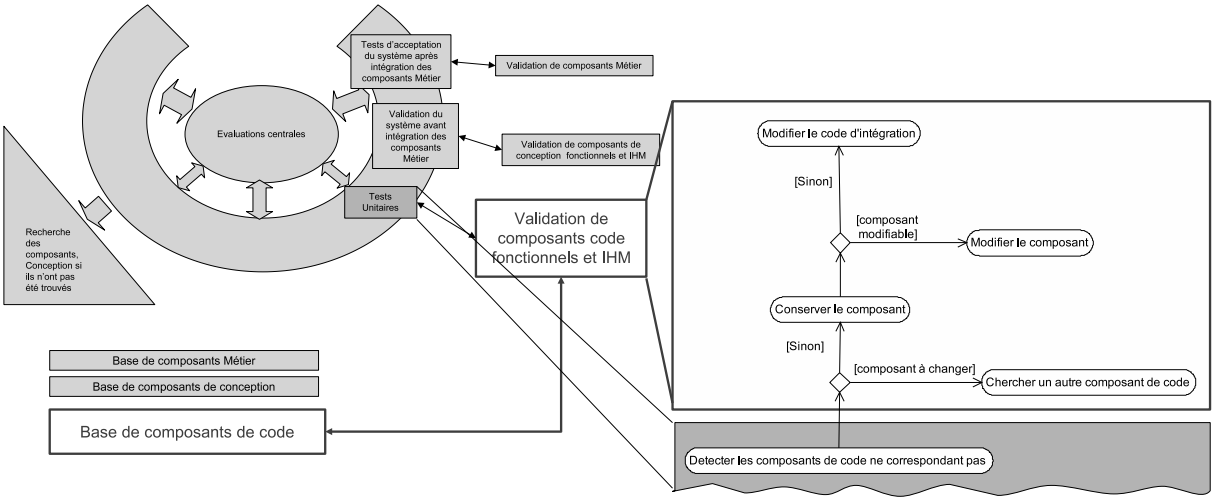


FIGURE 3.29 – Etape de validation de composants de code

3.3.14 Validation des composants de conception

Lors de l'intégration du système (sans les composants métier), il peut apparaître des problèmes liés aux solutions de conception choisies. Il faut alors regarder si ces problèmes peuvent être résolus (conservation et modification du composant choisi) ou si ils nécessitent une modification plus profonde, c'est-à-dire une modification de la solution de conception choisie auparavant (chercher un autre composant de conception), cf. Figure 3.30. Cela peut avoir des répercussions sur la programmation et le choix des com-

posants de code effectués auparavant. Si les problèmes rencontrés nécessitent un changement de solution de conception, il faut alors retourner à l'étape de recherche de composant de conception. Si lors de ce retour, aucune autre solution ne paraît satisfaisante, il faut retourner à l'étape de conception préliminaire.

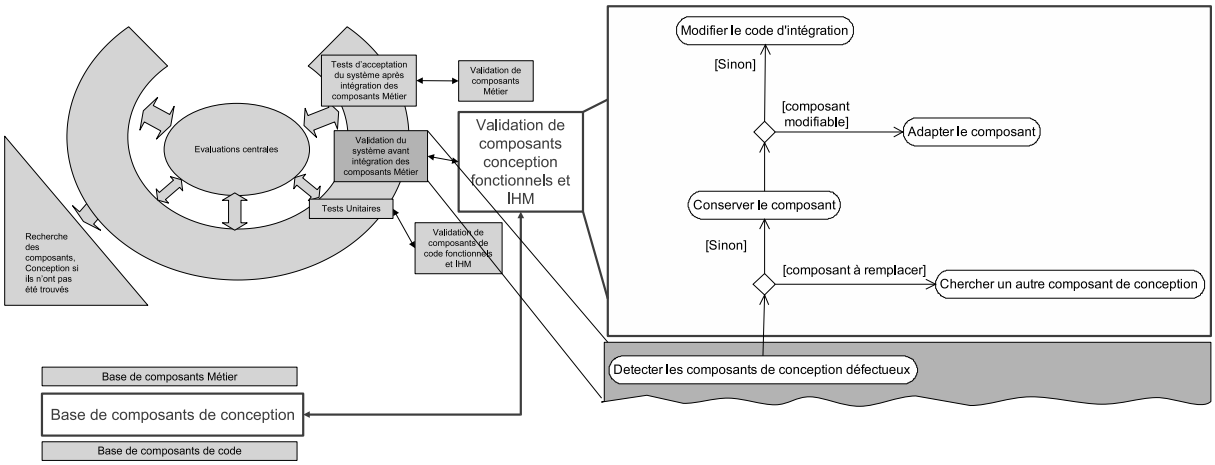


FIGURE 3.30 – Etape de validation de composants de conception

3.3.15 Validation des composants métier

La validation des composants métier se fait en fin de cycle car ils doivent être indépendants du système. Ces composants sont testés et validés lors de leur conception. Lors de l'étape d'évaluation du système avec les composants métier on cherche à voir si ils s'intègrent bien dans le système, si les interactions homme-SIAD-Composants métier sont correctes et si les composants métier apportent une réelle aide à l'utilisateur dans sa prise de décision. Les anomalies, détectées lors de l'étape de validation du système avec les composants métier, peuvent montrer :

- Une mauvaise utilisation d'un composant métier,
- Qu'un composant métier est mal adapté (il manque des fonctionnalités, il demande trop de détails et devient difficile à utiliser pour les utilisateur ; il ne répond pas au besoin des utilisateurs),
- Un manque au niveau des composants métier.

Dans le premier cas, il suffit de modifier le système pour prendre en compte correctement le composant métier. Dans le second cas, il faut rechercher un composant métier mieux adapté ou si ce n'est pas possible modifier le composant métier. Dans le troisième cas, il faut retourner à l'étape d'analyse pour ajouter les composants métier manquant (demandés par les utilisateurs) à la prochaine itération, cf. Figure 3.31.

3.3.16 Conclusion sur la méthode proposée

La méthode présente une description fine de l'ensemble des étapes du modèle d'ADESIAD. Cette méthode permet le développement de SIAD ne dépendant pas des composants métier. En effet, la méthode préconise un développement séparé du système et des composants. Ceci permet d'améliorer l'évolutivité

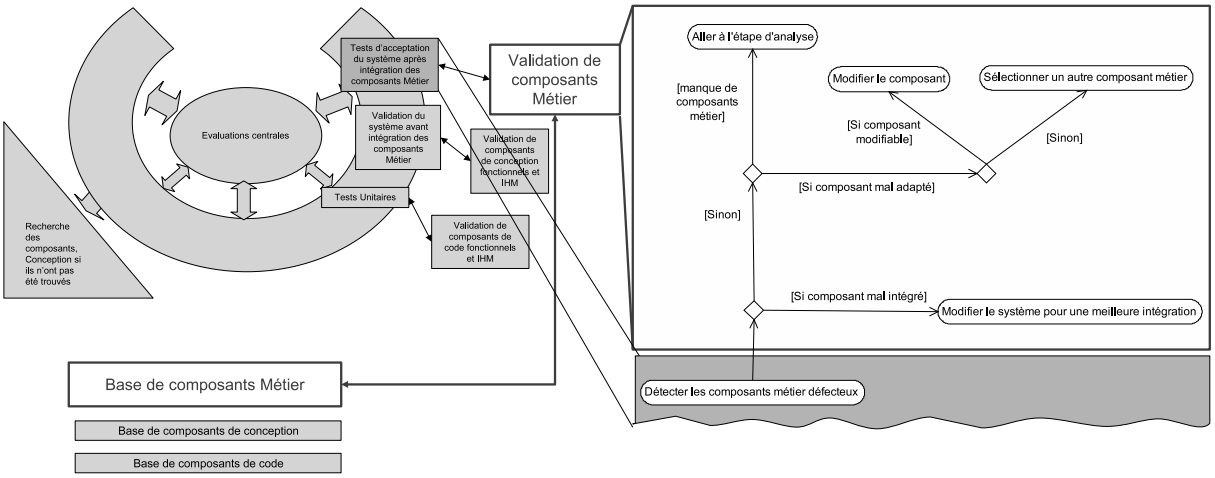


FIGURE 3.31 – Etape de validation de composants métier

du SIAD car d'autres composants métier peuvent être ajoutés sans repenser l'architecture du système. La combinaison originale des quatre principes associés aux modèles en U, en étoile, en X et MODESTI permet une prise en compte de l'utilisateur (décideur), de l'expert, et permet un développement à base de composant et une validation permanente par l'utilisateur. L'évaluation centrale et finale permet une meilleure maîtrise des risques et une prise en compte permanente de l'utilisateur. De plus la partie originale spécifique au SIAD, concernant la décomposition du problème permettant la modélisation des composants métier et la formalisation à l'aide de patron, fait partie de la valeur ajoutée de ce travail. Ces patrons donnent un moyen aux experts d'exprimer leurs connaissances ; ils peuvent en outre être exploités pour le partage et la réutilisation des connaissances dans l'entreprise. Ils offrent aussi un support pour la spécification et la recherche de composants métier.

3.4 ADESIAD : l'architecture

Nous allons maintenant rappeler les intérêts des différentes architectures présentées dans le chapitre 2. Nous pourrons alors proposer une architecture qui intègre encore une fois l'ensemble ou le maximum de ces intérêts.

3.4.1 Apport des différentes architectures

Les architectures des systèmes informatiques sont importantes car elles conditionnent l'évolutivité, les performances et le développement de ces systèmes. Ce sont elles qui donnent les contraintes de conception et d'implémentation. Les architectures présentées dans le chapitre 2 ont le point commun de séparer les composants d'interface des composants fonctionnels. Parfois d'autres composants intermédiaires s'insèrent entre les deux. Cette particularité est très importante pour ADESIAD car, dans la méthode, la conception des IHM a été séparée de la conception des fonctionnalités.

Nous allons présenter l'architecture choisie pour le développement de SIAD.

3.4.2 Architecture proposée

L'architecture proposée pour répondre aux critères rappelés précédemment est montrée sur la Figure 3.32. Elle comprend l'architecture du SIAD comme composant principal qui est relié à des composants métier via des interfaces. Les composants métier sont développés de manière indépendante et correspondent à des "boites noires" ; leurs codes et leurs architectures ne sont pas connus. Cependant, les composants métier étant des composants de grosse granularité, nous pouvons supposer qu'ils sont composites (ils sont eux-mêmes composés d'autres composants) et qu'ils possèdent, pour certains d'entre eux, leur base de données et une IHM. Sur la Figure 3.32, les composants métier sont volontairement représentés en dehors du SIAD pour montrer leurs indépendances mais surtout l'indépendance du SIAD par rapport à eux.

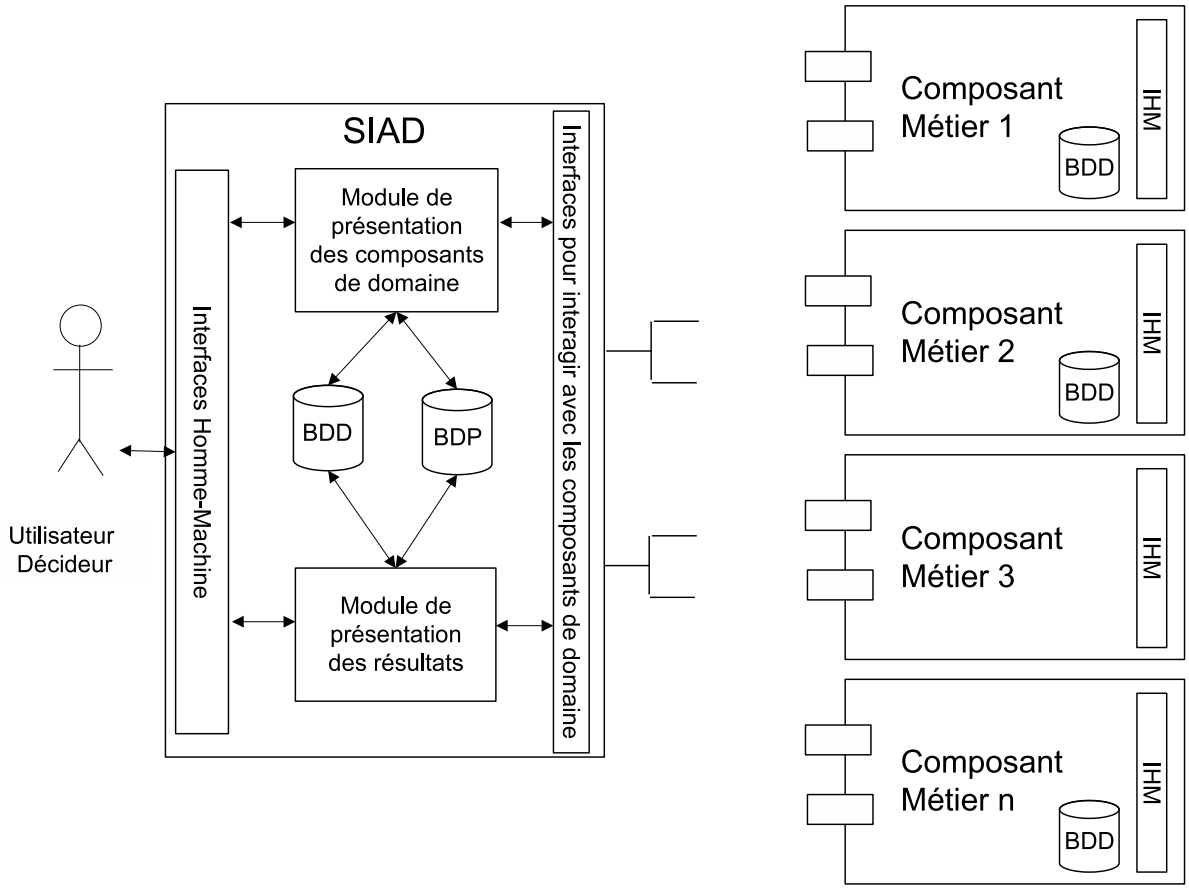


FIGURE 3.32 – Architecture de SIAD basé sur les composants

Le SIAD est lui-même composé d'une IHM permettant des interactions avec l'utilisateur. Il peut posséder une base de données (BDD) contenant des informations sur les utilisateurs, sur les utilisations précédentes et sur les dossiers déjà traités. De même, il peut contenir une base de problèmes (BDP) qui permet de faire le lien entre les problèmes, les solutions et les outils qui pourraient être utilisés pour aider à la résolution d'un problème. La base de problème pourrait être conjointe à la base de données mais nous avons voulu la distinguer pour montrer la spécificité de cette base aux SIAD.

Le SIAD est relié aux composants métier via les interfaces qui permettent des échanges de données et de contrôles. Le SIAD est composé de deux modules : un de présentation des composants métier qui permet à l'utilisateur de choisir les composants les mieux adaptés à son problème en fonction de son niveau d'expertise et un module de présentation des résultats qui à la suite d'utilisation des composants métier permet à l'utilisateur de voir une synthèse des résultats en relation avec son problème. Ces deux modules utilisent les bases de données et les bases de problèmes. Les autres types de composants ne sont pas représentés sur la Figure 3.32 car ils sont utilisés à l'intérieur des divers modules. L'architecture présentée correspond à une architecture fonctionnelle. Elle se veut indépendante des solutions techniques et standards actuels qui évolueront au cours des années à venir [Lapassat, 2003].

3.4.3 Conclusion sur l'architecture proposée

L'architecture proposée tente de supporter le processus et la méthode d'ADESIAD. Cette architecture est basée sur une programmation par composants. Elle vise à faciliter l'intégration de composants métier à n'importe quel moment du cycle de vie. L'architecture proposée suppose que les interfaces des différents composants métier sont similaires ; or la multiplicité des modèles de composants (cf. § 1.4.2.1) rend difficile l'intégration de composants basés sur des modèles différents. L'autre solution envisageable serait de doter le SIAD d'interfaces valables pour la majorité des modèles de composants. Cette solution ne serait de toute façon qu'une solution temporaire au sens où elle ne serait valable qu'un certain temps ; elle ne permettrait pas l'évolutivité du système sans intégration d'autres interfaces. Chaque composant métier intègre la connaissance experte pour le problème qu'il traite tandis que le SIAD intègre la connaissance experte sur la manière de traiter un problème.

3.5 Conclusion

Ce chapitre a rappelé les particularités de la conception des systèmes intégrant des notions de prise de décisions, de réutilisation, de gestion des connaissances, et d'interaction homme-machine. Ces particularités ont ensuite été intégrées à une seule et même approche ADESIAD (modèle, méthode et architecture) permettant de guider une équipe de développement en prenant en compte un certain nombre de spécificités des domaines concernés.

L'approche proposée se base sur une conception par composants. Trois types de composants (métier, de conception, de code) sont intégrés dans le cycle dès le niveau d'analyse et jouent un rôle extrêmement important dans ADESIAD. En effet, ces composants permettent un conception itérative, parallèle et incrémentale. Ils ont pour but de faciliter la réutilisation et la capitalisation des connaissances. Le cycle est raccourci et permet une évaluation complète plus rapide à chaque itération. Une démarche participative, plaçant l'évaluation au centre du processus permet des retours en arrière plus rapide et donc moins coûteux. Les évaluations sont facilitées par le choix d'une formalisation sous la forme de patron qui permet également la capitalisation et le partage des connaissances. Toutefois, l'approche proposée ne nous semble pas spécifique au développement de SIAD mais pourrait être utilisée pour un grand nombre de projets.

Dans la méthode, chaque activité du modèle de développement a été située. Certaines, issues de ces travaux, telles que l'analyse du processus de décision ou la décomposition du problème à l'aide de patrons, ont été plus détaillées. Les patrons utilisés pour formaliser la connaissance concernant un problème et pour détailler les composants métier à rechercher, ont également fait preuve de nombreuses descriptions et exemples.

Enfin, l'architecture proposée a permis de montrer la structure que devrait avoir un SIAD selon l'approche globale. Elle est restée conceptuelle car les choix plus précis dépendent de l'environnement dans lequel sera situé le SIAD.

Dans le chapitre suivant, ADESIAD va être mise en application dans le domaine ferroviaire en vue du développement de SIADIF (Systèmes Interactif d'Aide à la Décision en Investissement dans une infrastructure Ferroviaire).

Chapitre 4

Développement de SIADIF : un Système Interactif d'Aide à la Décision en Investissement dans une infrastructure Ferroviaire - basé sur ADESIAD

Sommaire

4.1 Introduction

Les chapitres précédents ont abouti à la présentation d'une approche de développement de SIAD intitulée ADESIAD. Cette approche va maintenant être appliquée à un cas concret dans le domaine ferroviaire pour le développement de SIADIF (Système Interactif d'Aide à la Décision en Investissement dans une infrastructure Ferroviaire). Ce système est développé en collaboration avec RFF (Réseau Ferré de France), propriétaire du réseau ferroviaire français. Il a pour but d'aider les employés de RFF à mener des études permettant des prises de décision relatives à l'investissement dans les infrastructures ferroviaires. Une introduction au problème de RFF a été présentée dans le premier chapitre pour montrer les manques en systèmes d'aide. Après avoir présenté les travaux antérieurs ayant abouti à ces travaux en première partie, la seconde partie de ce chapitre vise à appliquer l'approche ADESIAD au développement de SIADIF.

4.2 Travaux antérieurs : projet INFRAFER

Nos précédents travaux, [Lepreux et al., 2001a, Lepreux et al., 2001b, Lepreux et al., 2003a], ont porté sur l'adaptation du modèle de développement en U afin de mieux prendre en considération les experts dans un contexte d'aide à la décision. A travers ces travaux, le modèle en U (déjà décrit dans sa version initiale dans le chapitre 2 § 2.3.1.5) a été complété par une phase d'intégration des connaissances expertes (cf. Figure 4.1), visant à faciliter l'extraction des connaissances expertes. Pour cela, l'immersion du concepteur dans le domaine est nécessaire. Une fois cette base de communication établie, les échanges entre experts et concepteurs deviennent en principe plus riches. Les énoncés et les règles de travail sont alors plus facilement formalisables sous la forme d'algorithmes. Soulignons qu'un travail important consiste pour les experts à valider ces formalismes.

Ce modèle étendu fût utilisé pour une étude de cas ferroviaire pour le développement d'un système de calcul de capacité et de simulation de trafic. Le système, nommé INFRAFER (INFRAstructure FERroviaire) fût développé dans le cadre d'un projet PREDIT en collaboration avec RFF et CORYS-TESS [Paulhac et al., 2002]. Ce type d'outil devait permettre aux experts d'évaluer la capacité ferroviaire (et de la valider par la simulation) pour les aider à prendre des décisions dans les investissements. Le développement de ce système en collaboration avec les experts de RFF a montré l'importance de la prise en compte des experts tout au long du processus. La Figure ?4.2 montre comment la connaissance (en haut à gauche) extraite lors de réunion et d'interviews, accompagnée des documents (en haut à droite) utilisés par les experts liés à leur connaissance, ont permis de mettre au point le système et son interface homme-machine (en bas). La connaissance des experts a été utilisée tout au long du processus aussi bien concernant l'interface homme-machine, comme le montre la Figure 4.2, mais aussi pour la prise en compte des habitudes de travail et de l'importance de certains traitements.

Mené dans le cadre de mon DEA [Lepreux, 2001], ces travaux ont montré :

– une déficience en outil d'aide à la décision global dans ce domaine,
– un manque de méthodologies de développement de SIAD.

Le besoin d'un SIAD plus complet est apparu. Sa méthodologie de développement devrait prendre en

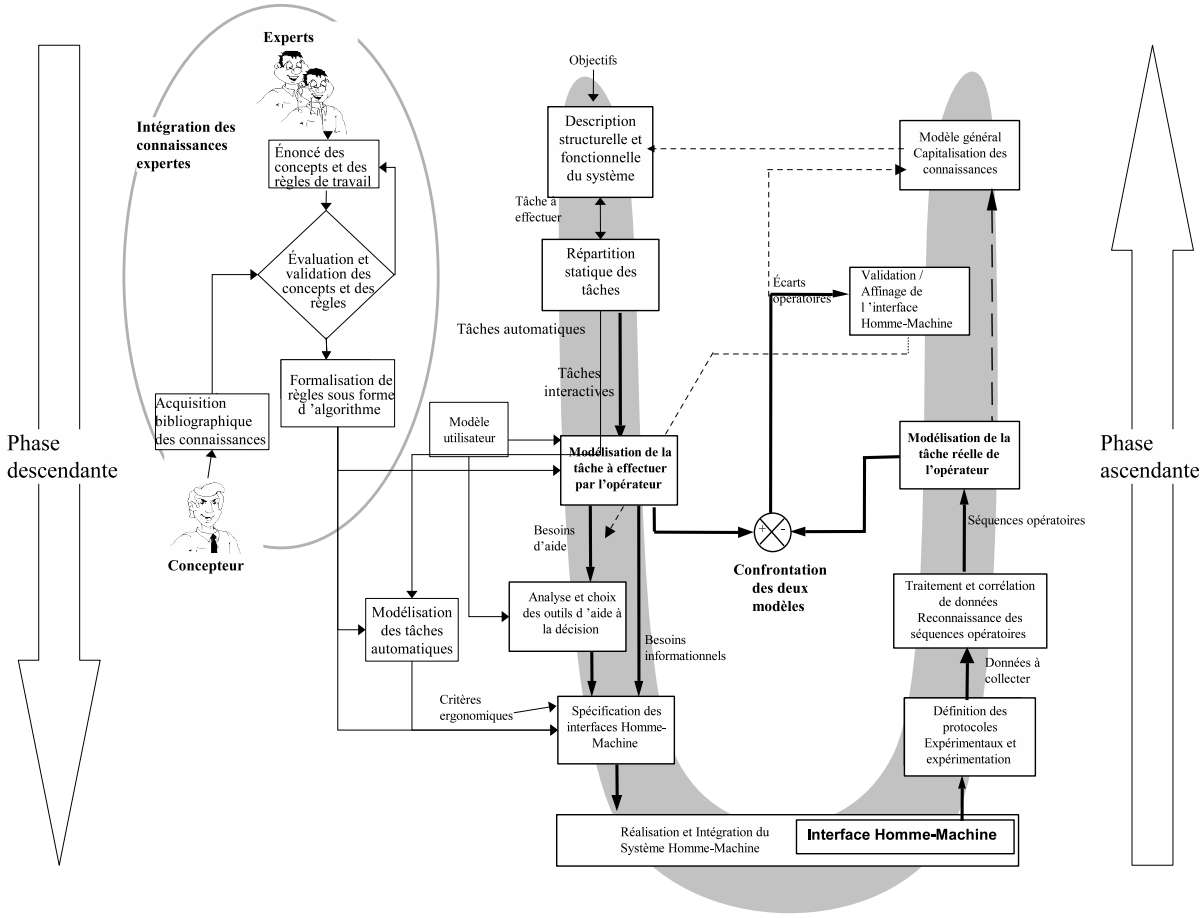


FIGURE 4.1 – Modèle en U adapté pour la prise en compte d'experts [Lepreux et al., 2001a]

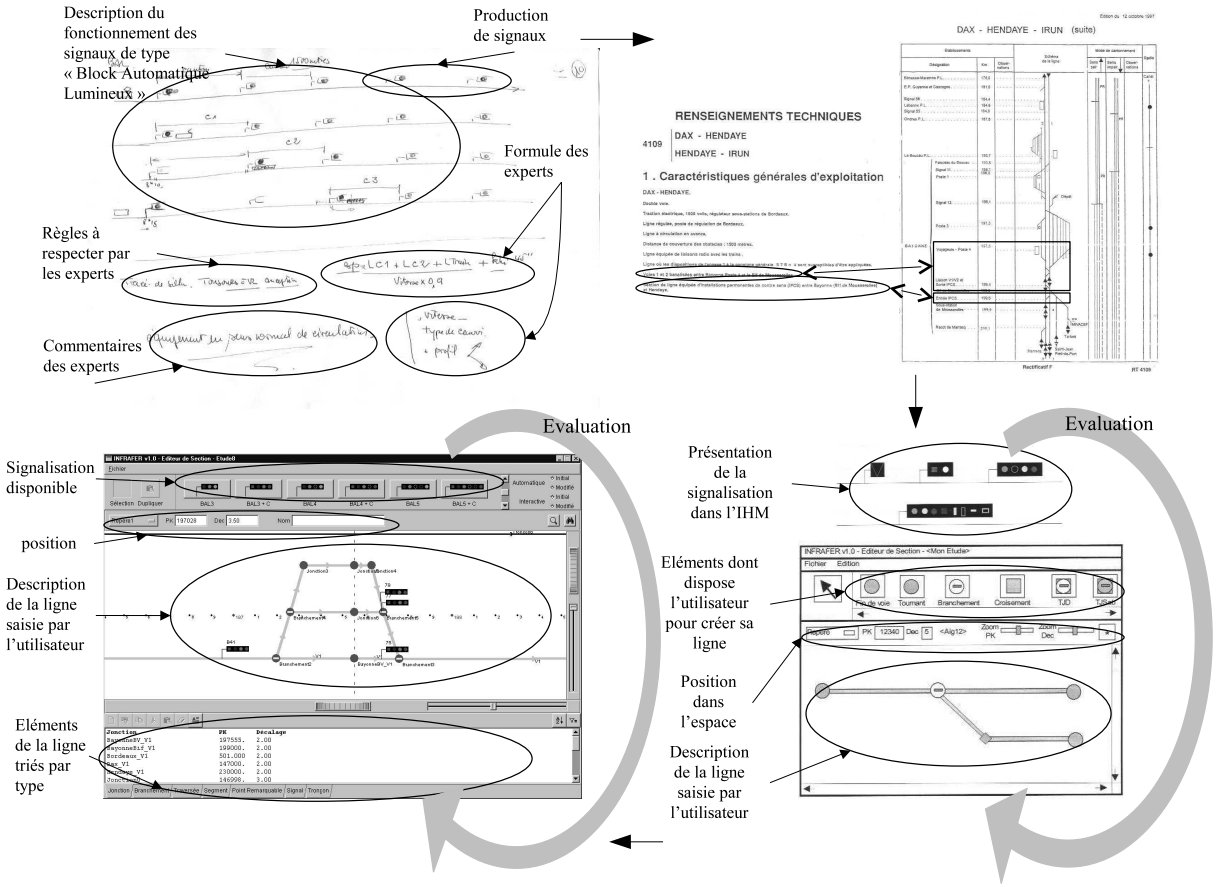


FIGURE 4.2 – Prise en compte des connaissances pour la conception du système et de son interface

compte les utilisateurs et les experts par intégration de composants extérieurs. Les travaux menés durant cette thèse ont alors tenté de répondre à ces attentes. Le chapitre précédent a proposé une approche de développement plus complète tandis que ce chapitre va montrer son application au SIAD pour les investissements dans les infrastructures ferroviaires (SIADIF).

4.3 SIADIF et le modèle d'ADESIAD

En accord avec RFF, suite au bilan du projet INFRAFER, le SIAD, développé dans le cadre de l'aide à l'investissement dans les infrastructures ferroviaires, doit :

- être centré sur le décideur, l'utilisateur et l'expert.
- donner accès à des outils d'analyse correspondant au problème (par exemple, le problème d'investissement).
- accompagner le décideur durant tout son processus de décision, c'est-à-dire en le guidant dans le choix de ces outils.

Ces caractéristiques nous ont amené à considérer les outils d'analyse comme des composants métier et à viser un développement à base de composants.

Le système à développer (SIADIF) est en concordance avec les systèmes ciblés par l'approche ADESIAD. Il est à considérer comme étant novateur ; le principe itératif du modèle d'ADESIAD accompagné des évaluations centrales devient un atout indispensable pour un développement plus efficace. La conception parallèle du système et des composants métier doit permettre des gains de temps en développement. La réutilisation des systèmes existants doit devenir aussi un atout pour le SIAD.

4.4 SIADIF et la méthode d'ADESIAD

Nous allons dans cette partie mettre en pratique la méthode présentée dans le chapitre précédent. Nous expliquerons à chaque étape la démarche suivie et une partie des résultats associés.

4.4.1 Etape d'analyse

4.4.1.1 Recueillir les connaissances expertes

L'extraction de la connaissance des experts s'est faite tout au long du projet. Elle a débuté par une phase d'initiation qui a permis aux intervenants dans le projet de communiquer pour avoir des bases communes. Pour faciliter la communication, les premières définitions concernant le fonctionnement de RFF et le fonctionnement du système ferroviaire sont fournies. Une bonne analyse, et par conséquent l'ensemble du projet, dépend de cette communication. Par la suite, de nombreuses réunions ont permis d'extraire les connaissances au fur et à mesure du projet.

Dans le cadre de ces travaux, cette phase a été facilitée par l'historique du projet. En effet, les connaissances principales liées au ferroviaire avaient été transmises lors de nos travaux précédents concernant le logiciel de calcul de capacité et de simulation de trafic (lors du projet PREDIT INFRAFER). Il nous restait à intégrer les connaissances liées au fonctionnement de RFF pour les investissements et aux rôles des différents acteurs intervenant dans le processus de prise de décision. Ces rôles ont été représentés à l'aide du diagramme de cas d'utilisation d'UML, cf. Figure 4.3. Puisque nous nous situons dans le cadre d'un développement itératif, nous avons centré le système sur les cas d'utilisation concernant la proposition d'aménagement et l'analyse de la proposition. Les cas d'utilisation correspondant à l'évaluation du coût et de la participation de RFF et le vote du projet ne concerneront pas nos travaux actuels. Dans ses premières itérations, le développement du système a principalement concerné les études ferroviaires.

4.4.1.2 Analyse de l'existant et du besoin

Dans les cas d'utilisation qui nous intéressent lors de cette itération, les utilisateurs sont les experts ferroviaires. Puisque ceux-ci utiliseront le système, ils doivent participer à son développement. L'analyse de l'existant a permis de constater qu'il n'existait pas de logiciel d'aide à la décision pour ce problème d'investissement. Des logiciels d'évaluation d'infrastructure existent et ont été présentés dans le premier chapitre. Les utilisateurs nous ont donné leur avis concernant ces logiciels existants qu'ils connaissent. Les résultats de cette analyse ont été présentés dans le premier chapitre. Rappelons qu'ils mettent en évidence le manque d'outils complets pour l'aide à la décision dans les investissements. Seuls des outils

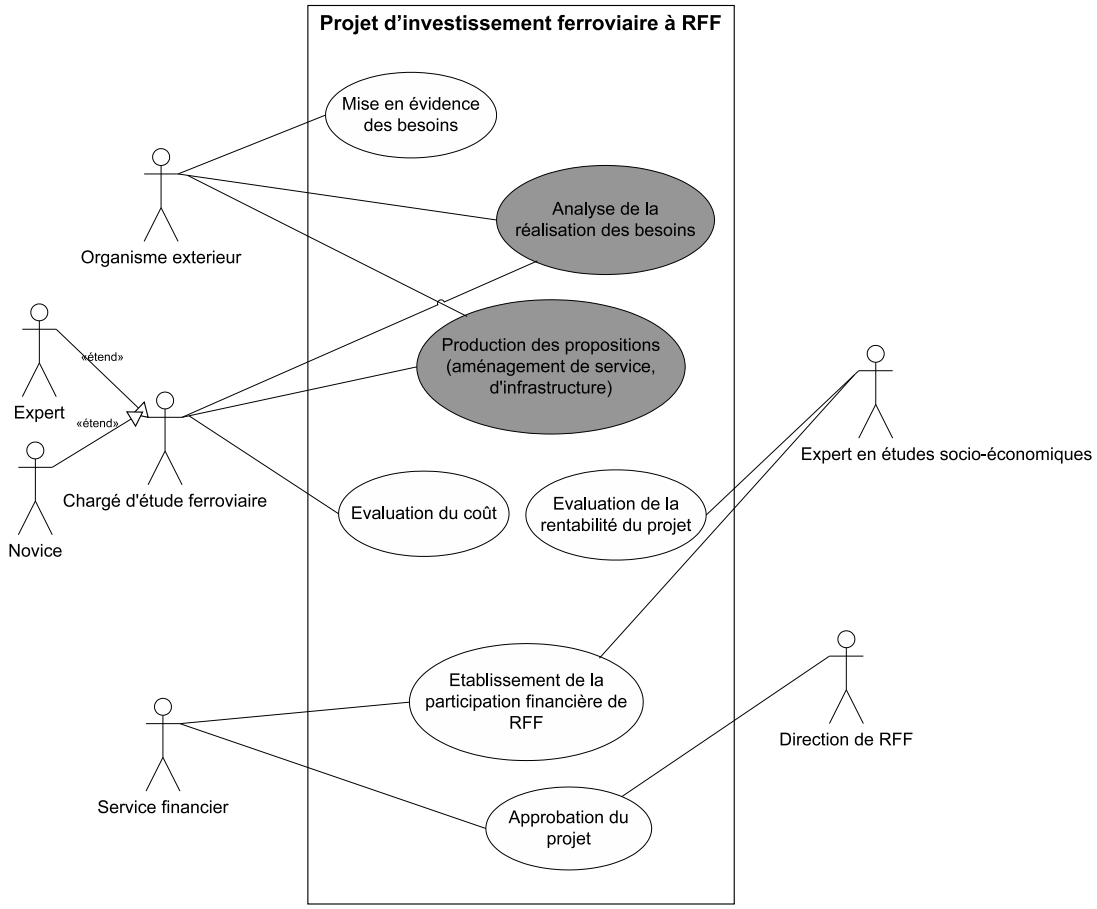


FIGURE 4.3 – Diagramme des cas d'utilisation pour la réalisation d'un projet d'investissement (les cas grisés sont ceux pris en compte dans la conception de SIADIF)

plus spécifiques à des sous problèmes existent. Le besoin a été exprimé progressivement par des experts et utilisateurs lors de plusieurs réunions consacrées à cet effet. Dans l'extrait du document de travail visible en Figure 4.4, un chargé d'études ferroviaires expert a exprimé le besoin de mesurer les solutions des problèmes (idéales techniquement ou plus économes) par rapport à des critères respectant des unités pour permettre des comparatifs entre ces solutions.

4.4.1.3 Décomposition du problème

De nombreuses réunions avec les experts ont permis de décomposer le problème principal. Plusieurs itérations ont été nécessaires avant que la première décomposition ne soit validée par les experts et les futurs utilisateurs. La première décomposition du problème principal d'investissement est donnée sur la Figure 4.6. Elle a ensuite évolué vers la décomposition visualisée sous la forme d'un arbre (cf. Figure 4.7), les feuilles correspondant à des problèmes non décomposables auxquels on peut associer des outils d'analyse. Chaque décomposition est accompagnée d'explications des experts pour que le problème soit bien compris, cf. Figure 4.5. Les patrons n'étaient pas encore définis tels que présentés dans le chapitre trois. Ils ont évolué au cours du développement pour s'adapter au besoin du développement et

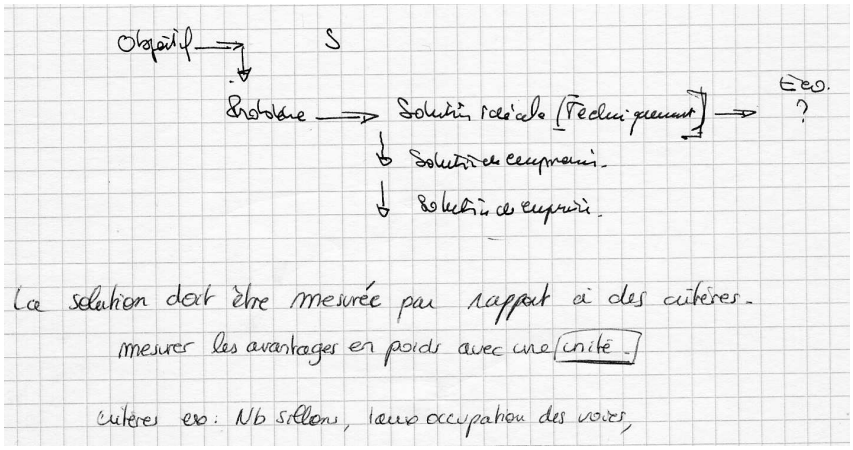


FIGURE 4.4 – Extrait manuscrit d'une réunion sur les besoins des utilisateurs

du problème. Ils ont donc ensuite été utilisés pour aboutir à une troisième décomposition (cf. Figure 4.8).

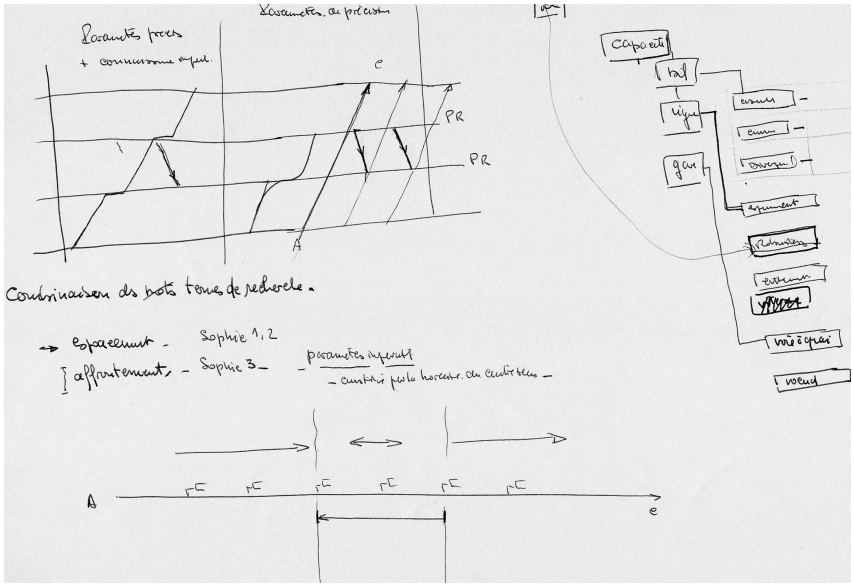


FIGURE 4.5 – Extrait d'un manuscrit venant d'experts concernant la décomposition du problème de capacité ferroviaire

A titre d'exemple nous en avons représenté quatre :

- Le problème père d'investissement (cf. Figure 4.9),
- Le problème fils correspondant à l'analyse de l'infrastructure par rapport aux sillons (cf. Figure 4.10),
- Un des problèmes fils du problème père correspondant au problème de capacité (Figure 4.11), et
- Un problème fils du problème de capacité, non décomposable, en l'occurrence le problème de capacité en ligne (Figure 4.12).

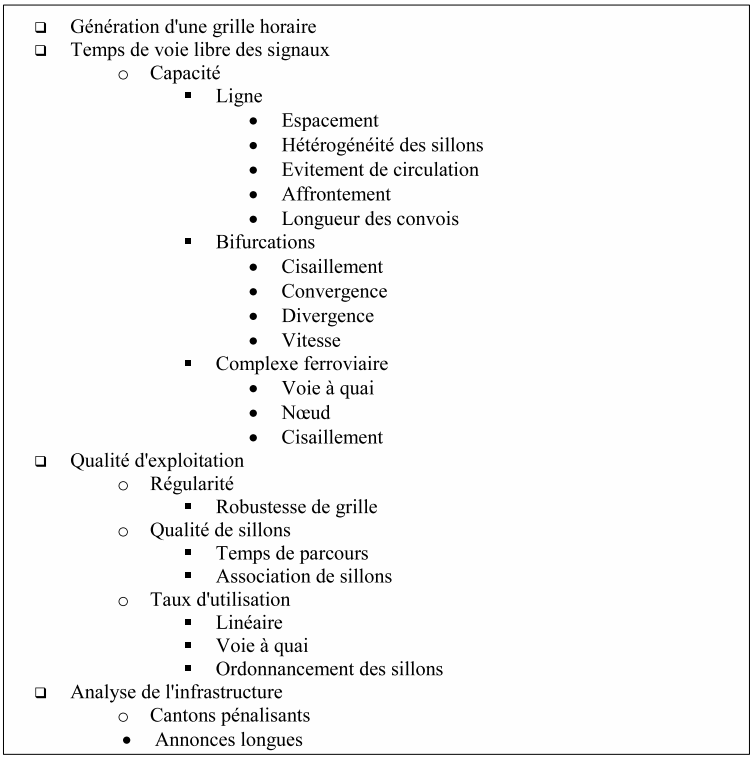


FIGURE 4.6 – Première proposition de décomposition du problème

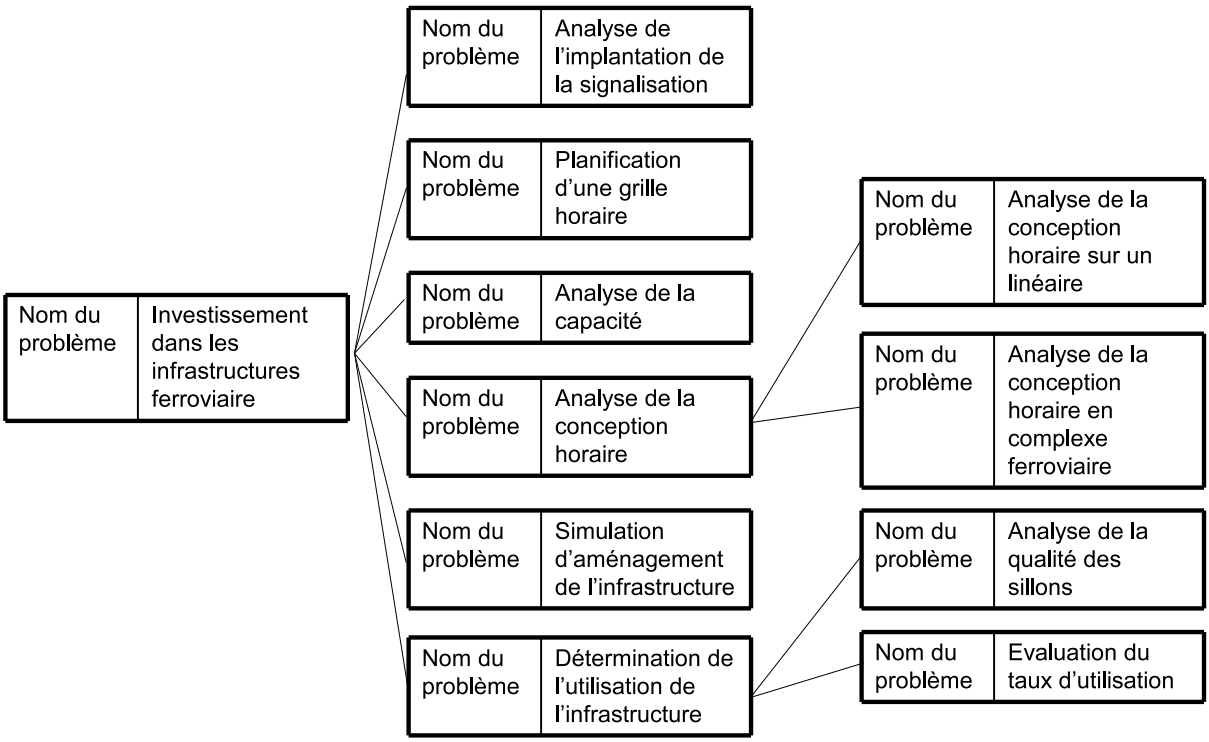


FIGURE 4.7 – Seconde décomposition du problème d'investissement

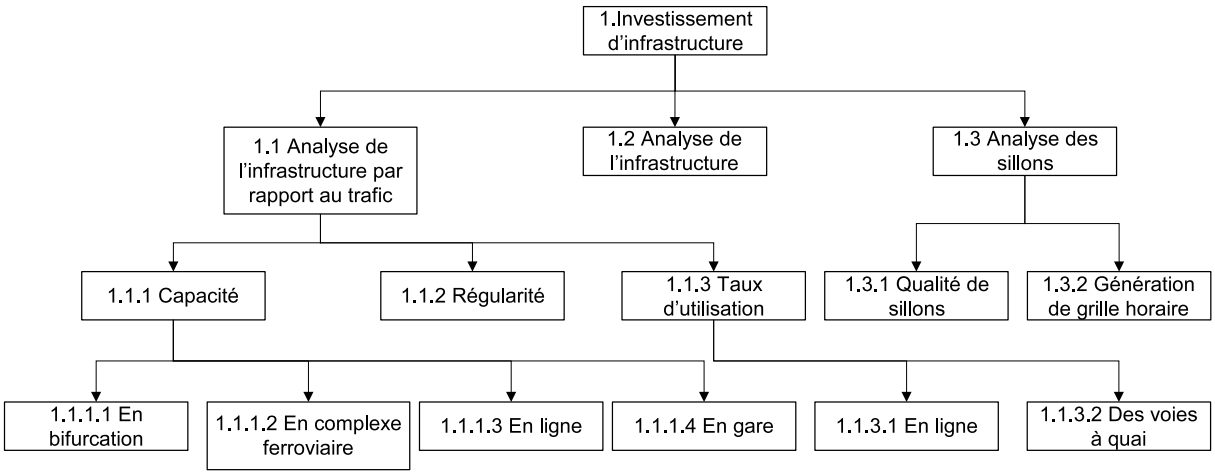


FIGURE 4.8 – Troisième décomposition du problème d'investissement faite à l'aide des patrons

Nom du problème	Investissement d'infrastructures
Définition du problème	Adaptation de l'infrastructure à la demande d'utilisation
Attributs	Infrastructure de base, besoin (demande)
Problème(s) père	Non
Sous problème(s)	Analyse de l'infrastructure Analyse de l'infrastructure par rapport aux sillons Analyse des sillons
Problèmes connexes	Non
Analyse	Non définie
Solutions	Non définies

FIGURE 4.9 – Instanciation du patron au problème père (investissement)

Nom du problème	Analyse de l'infrastructure par rapport aux sillons
Définition du problème	Un investissement dans une infrastructure ferroviaire est nécessaire quand l'infrastructure courante n'est pas suffisante pour assurer un besoin
Attributs	Infrastructure de base, besoin
Problème(s) père	Non
Sous problème(s)	Capacité Régularité Taux d'utilisation de l'infrastructure
Problèmes connexes	Analyse des sillons Analyse de l'infrastructure
Analyse	Simulateur de trafic avec infrastructure fine
Solutions	Non définies

FIGURE 4.10 – Instanciation du patron au problème d'analyse de l'infrastructure par rapport aux sillons

Nom du problème	Capacité
Définition du problème	La capacité ferroviaire correspond au nombre maximum de sillons qui peuvent s'intégrer sur une portion d'infrastructure durant un intervalle de temps donné.
Attributs	Infrastructure, grille horaire, besoin : quantité et type de sillons
Problème(s) père	Investissement d'infrastructures
Sous problème(s)	Ligne (système d'espacement) Bifurcation Gare (nb de voies de stationnement) Complexe ferroviaire (croisement)
Problèmes connexes	Régularité Taux d'utilisation
Analyse	Non définie
Solutions	Non définies

FIGURE 4.11 – Instanciation du patron au problème de capacité

Nom du problème	Capacité en ligne
Définition du problème	La capacité ferroviaire correspond au nombre maximum de sillons qui peuvent s'intégrer sur une portion d'infrastructure durant un intervalle de temps donné.
Attributs	Infrastructure, grille horaire, besoin : quantité et type de sillons
Problème(s) père	Capacité
Sous problème(s)	Non
Problèmes connexes	Capacité en bifurcation, en gare, en complexe ferroviaire, Taux d'utilisation en ligne
Analyse	Insertion d'un type de sillon Insertion d'un type de sillon avec cadencement Insertion d'un ensemble de sillons hétérogènes Analyse de l'implantation de la signalisation
Solutions	Modification du système d'espacement (la modification peut s'effectuer entre un block manuel, un block automatique lumineux ou la planification des signaux). Installation d'une voie de dépassement. Aménagement de la vitesse

FIGURE 4.12 – Instanciation du patron au problème de capacité en ligne

4.4.1.4 Modéliser SIADIF

Le diagramme des cas d'utilisation, cf. Figure 4.13, montre que pour traiter un problème le chargé d'études (que nous nommerons utilisateur dans la suite) aura besoin de chercher des outils parmi les outils d'analyse existants. Il montre également que l'utilisateur et les décideurs (le service financier et la direction de RFF) n'ont pas tout à fait les mêmes rôles. Le décideur final intervient seulement à la fin des études. Les tâches sont décomposées en sous-tâches comme le montre la Figure 4.14 pour la tâche "traiter le problème". La notation utilisée correspond à celle de la décomposition de tâches de Buisine [Buisine, 1999], cf. § 3.3.1. De même la tâche "choisir les outils" se traite en trois étapes : le choix du mode de présentation, la présentation des outils et la sélection. Cette décomposition rend possible la modélisation du système de manière statique à l'aide d'un diagramme de classes (Figure 4.15).

4.4.1.5 Identification des tâches humaines versus automatiques

Les tâches qui ont été déterminées dans l'étape précédente de modélisation vont être réparties en tâches automatiques où l'utilisateur n'apparaît pas et en tâches interactives qui le font intervenir. Dans l'exemple précédent illustrant la décomposition de la tâche "Traiter le problème", les tâches interactives seront "Choisir le mode de présentation", "Sélectionner les outils", "Utiliser les outils", "Analyser les résultats". Les tâches automatiques seront "Présenter les outils" et "Présenter les résultats". La tâche "Présenter les résultats" pourra devenir interactive si on décide de faire intervenir l'utilisateur dans le choix de la présentation ou dans la présentation elle-même. Cette répartition a été issue de discussions avec les experts et utilisateurs, un extrait d'un document manuscrit rédigé lors d'une réunion illustre la vue des experts et utilisateurs et montre les attentes qu'ils ont du système, cf. Figure 4.16.

L'étape d'analyse est ensuite validée par les utilisateurs, décideurs et l'ensemble des acteurs (informaticien et cogniticien) intervenant dans cette étape.

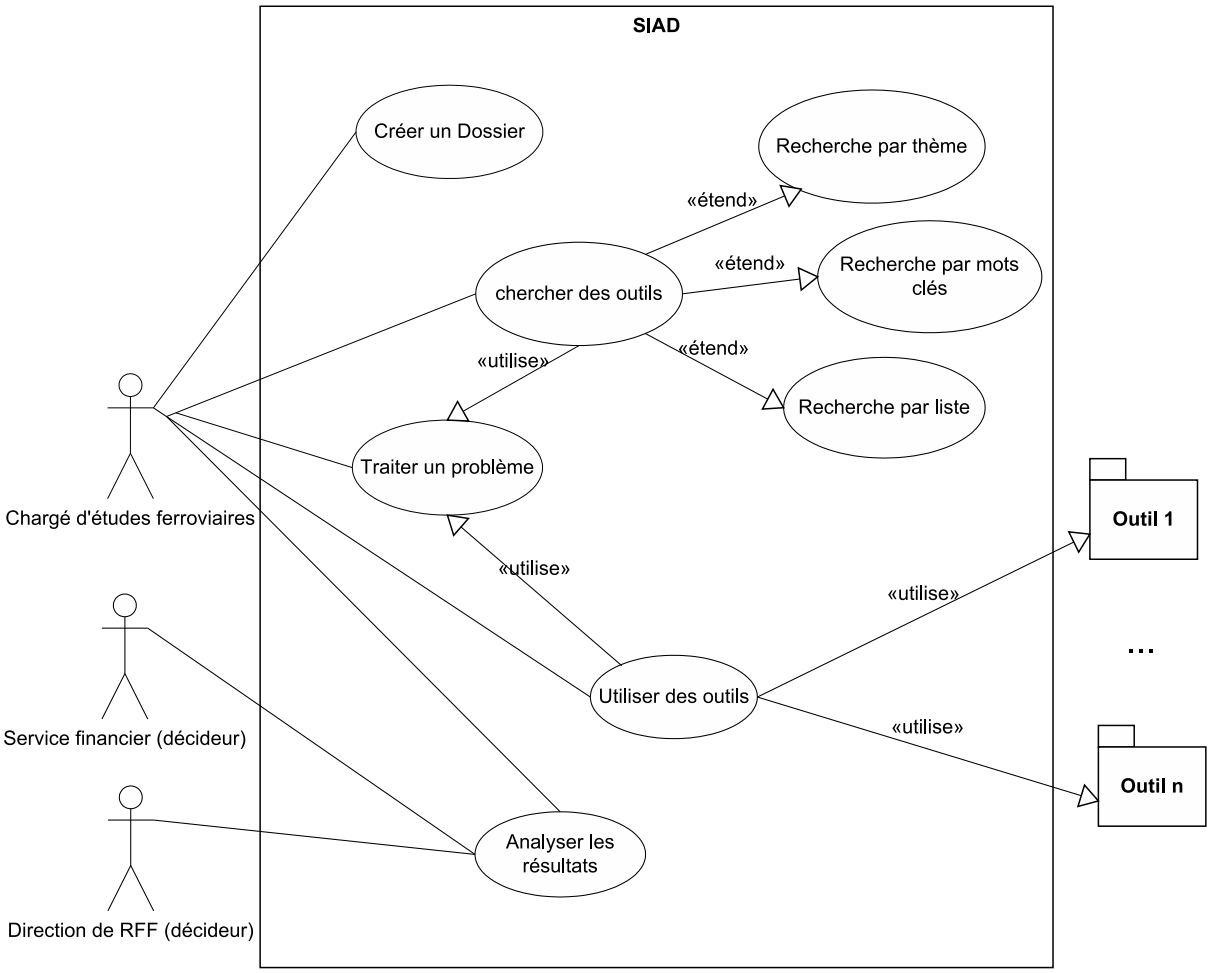


FIGURE 4.13 – Diagramme des cas d'utilisation du SIAD

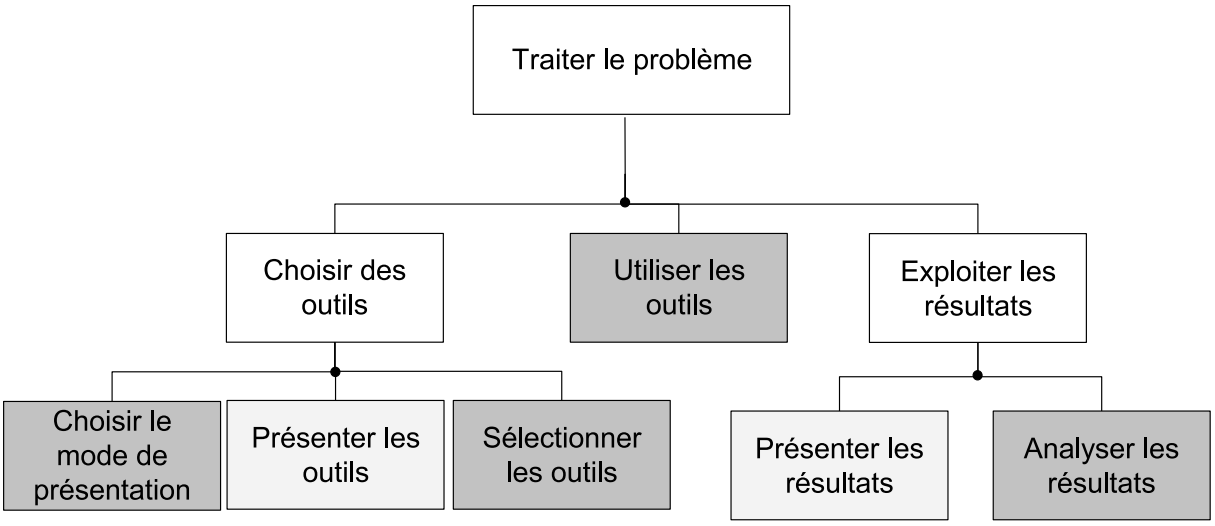


FIGURE 4.14 – Décomposition de la tâche "traiter le problème" utilisant la notation de [Buisine, 1999]

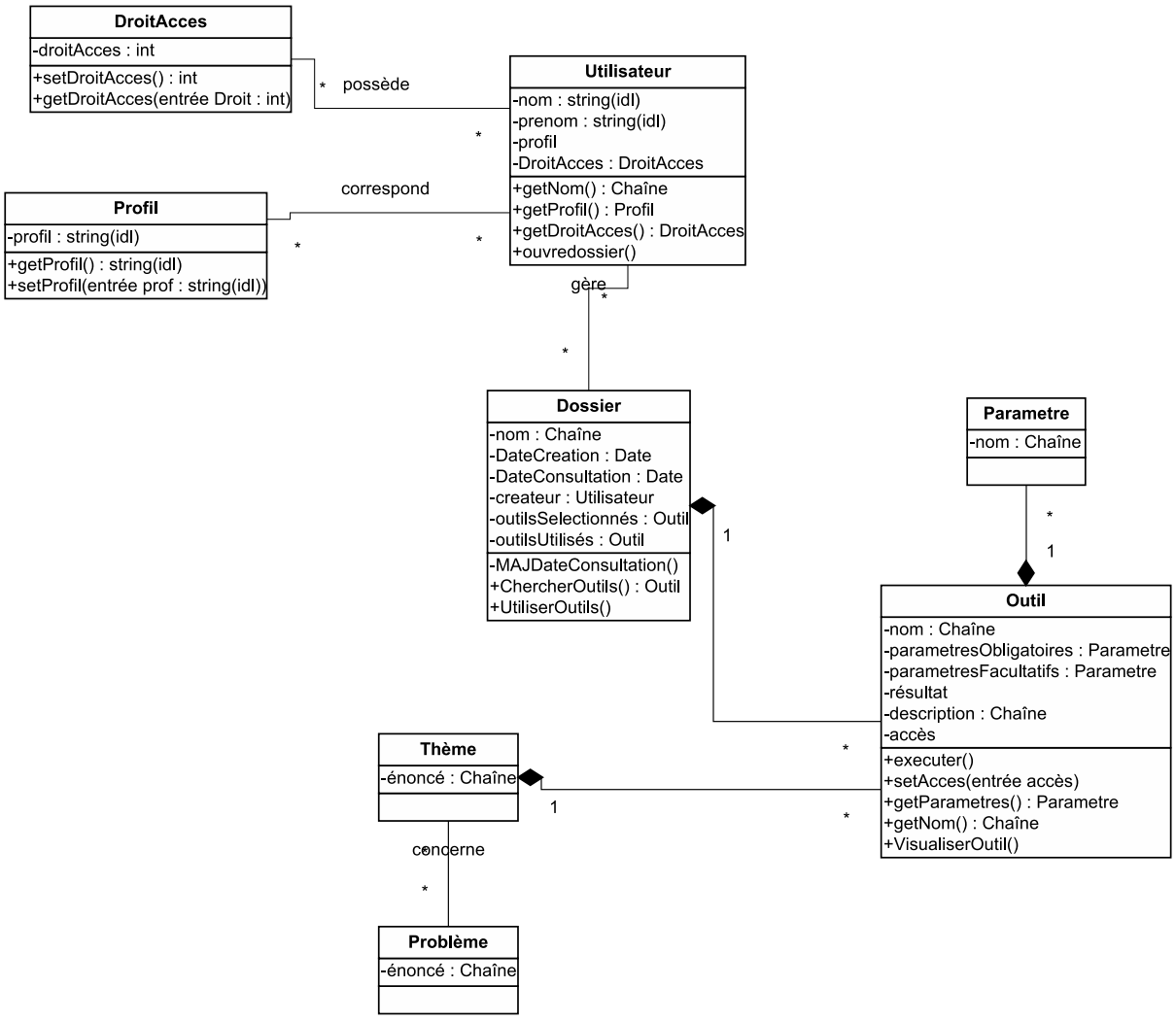


FIGURE 4.15 – Diagramme de Classes conceptuel de SIADIF

4.4.2 Etape de spécifi cation

En cohérence avec le chapitre précédent, l'étape de spécification a pour principal but la spécification des composants métier, la spécification des tâches automatiques, la spécification des tâches interactives, et le maquettage des IHM.

4.4.2.1 Spécifier les composants métier

La spécification des composants métier se base sur l'utilisation du patron de composant proposé dans la méthode d'ADESIAD. Chacune des feuilles de l'arbre de décomposition du problème (cf. Figure 4.7) doit pouvoir être traitée par un ou plusieurs outils (cf. Figure 4.17). Ces outils sont formalisés à cette étape à l'aide du patron de composant. Par exemple, l'estimation du problème de capacité en ligne peut être menée à l'aide des composants "insertion d'un type de sillon" (cf. Figure 4.18), "insertion avec cadencement", "insertion de plusieurs types de sillons". Les champs du patron ont été évalués et validés par

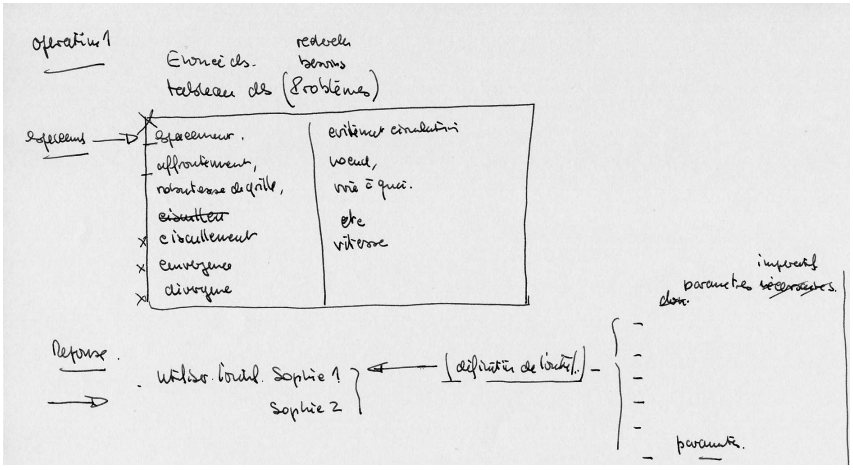


FIGURE 4.16 – Extrait manuscrit d'un expert concernant le déroulement de la tâche "Sélectionner les outils"

les experts et les utilisateurs, un extrait montre les réflexions concernant les champs "Paramètres obligatoires" et "Paramètres experts (facultatifs)", cf. Figure 4.19 [Lepreux et al., 2003d]. L'intitulé du champ mots-clés a été ajouté ultérieurement aux patrons de composants de manière à compléter la description et en vue de fournir les informations nécessaires au mode de recherche d'outils par mots-clés. En effet, les premiers tests de ce mode ont mis en évidence un problème de vocabulaire dans le domaine. La recherche se faisait initialement sur le contenu des champs. Cependant les utilisateurs utilisaient d'autres termes équivalents à ceux utilisés dans les patrons. Le champ "mots-clés" a donc été ajouté pour tenter de résoudre ce problème.

4.4.2.2 Spécifier les tâches automatiques

Les tâches automatiques sont spécifiées à l'aide des diagrammes d'activités du langage UML. La Figure 4.20 présente le diagramme d'activités de la tâche "Présentation des outils". Elle permet de structurer la présentation des outils en fonction du mode sélectionné auparavant. Les modes qui ont été définis en collaboration avec les experts sont au nombre de trois :

- Le mode "par Liste" consiste à présenter l'ensemble des outils de manière non triée sous la forme d'un tableau présentant les caractéristiques de l'outil telles qu'elles sont définies dans le patron.
- Le mode "par Thème" consiste à utiliser la décomposition du problème en sous-problèmes pour guider l'utilisateur dans le choix de ses outils. Il présente une arborescence des thèmes. L'utilisateur navigue dans cette arborescence pour sélectionner les outils qui l'intéressent.
- Le troisième mode "Par mots clés" n'a pas réellement été demandé par l'utilisateur. L'idée leur a été soumise ; elle leur a paru intéressante mais reste à être évaluée. Il a été intégré à titre expérimental. Il consiste à présenter un ensemble fini de mots clés et à permettre à l'utilisateur de mettre du texte correspondant à son problème. Il présente alors une liste d'outils triés par ordre d'importance par rapport aux mots clés sélectionnés et à l'intitulé du problème. L'utilisateur peut sélectionner les outils qui l'intéressent dans cette liste.

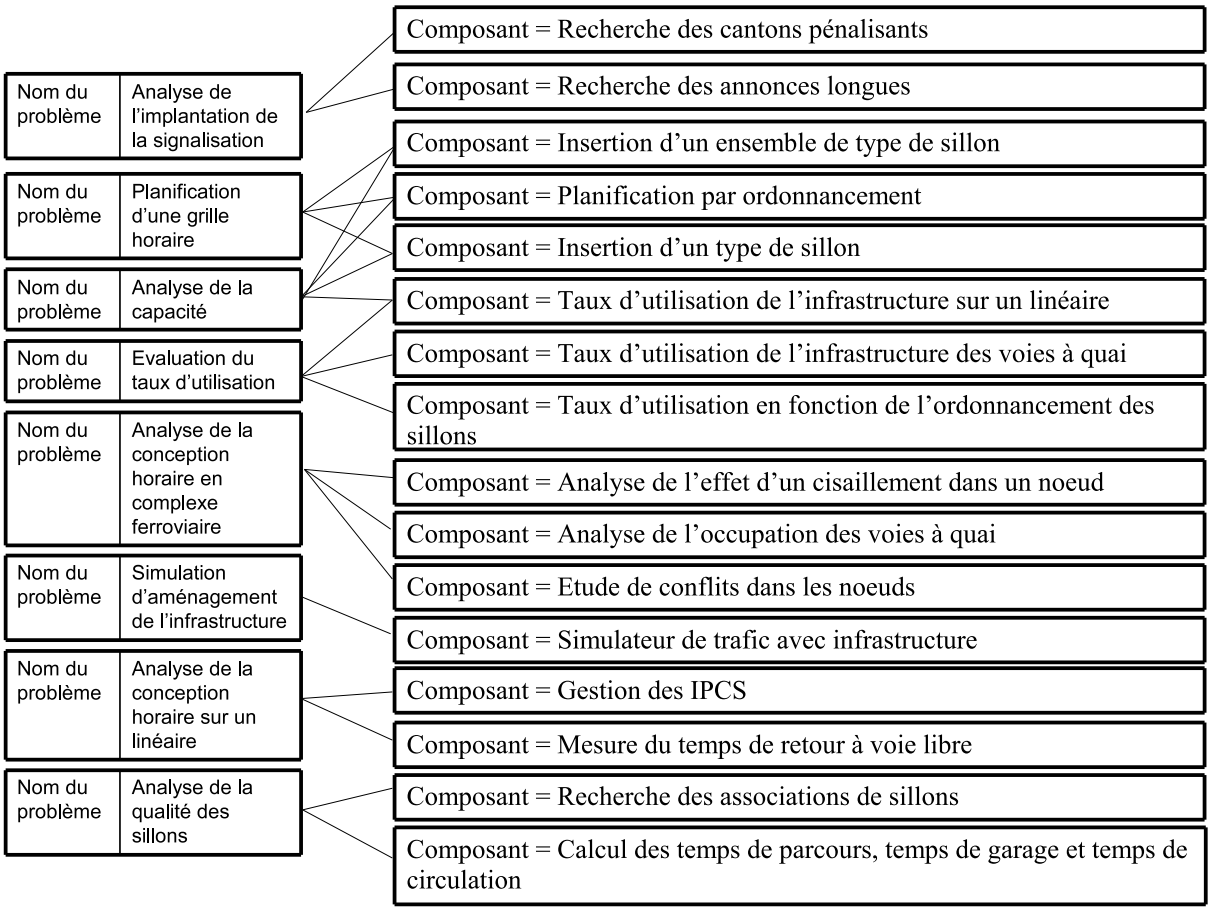


FIGURE 4.17 – Association des composants métier aux problèmes non décomposables correspondant à la seconde décomposition vue en Figure 4.7

Lors de la spécification, il a été suggéré que la présentation des caractéristiques des outils dans les trois modes soit la même : un tableau de six colonnes indique à l'utilisateur le nom, les données obligatoires et facultatives en entrées, les résultats sortants et une description de l'objectif de l'outil ; la sixième colonne intègre une case à cocher pour permettre la sélection.

4.4.2.3 Modéliser l'utilisateur

Nous avons distingué trois types d'utilisateur : le décideur, le chargé d'études ferroviaires intermédiaires et le chargé d'études ferroviaires expert. Les profils types correspondant sont présentés en Tableau 4.1, Tableau 4.2, Tableau 4.3. Tous ces utilisateurs devront être initiés au monde ferroviaire, ils devront connaître le vocabulaire utilisé (dans le profil, cette information se situe dans "travail et expérience" : "connaissance de la tâche"). Le niveau de compétence dans le domaine des études ferroviaire, qui varie selon l'expérience de l'utilisateur, influe sur l'utilisation du système. Dans les itérations futures des aides en ligne pourront être ajoutées pour guider chaque type d'utilisateur.

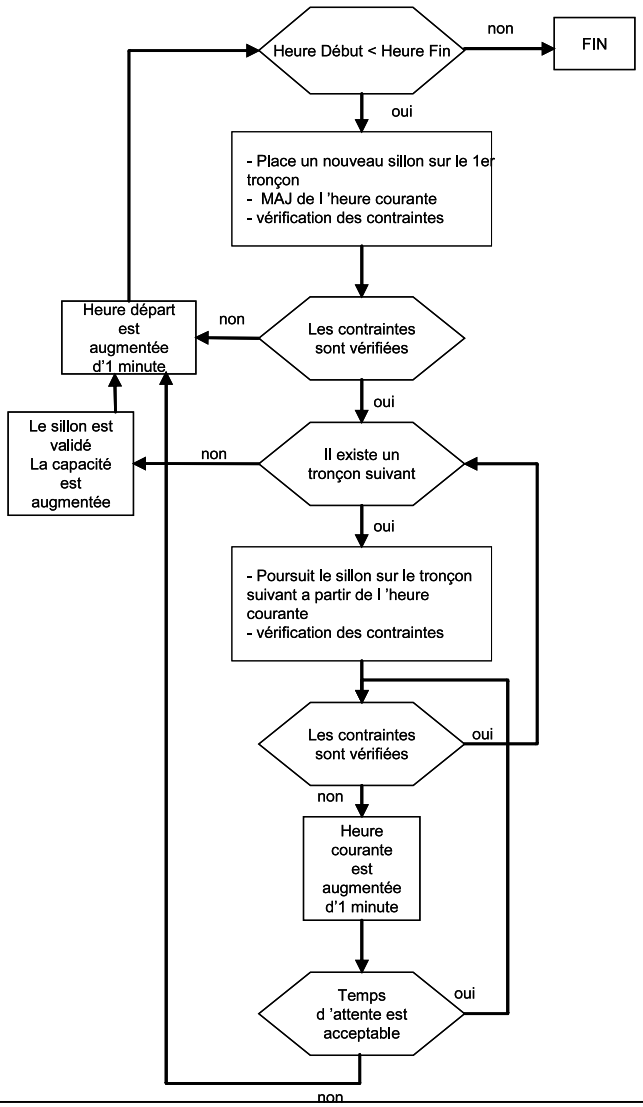
Composant = Insertion_d'un_type_de_sillon	
Intention	Insèrer un sillon de type donné dans une grille horaire
Contexte	Analyse de la capacité résiduelle sur un linéaire
Données entrée obligatoires	Infrastructure, grille horaire, type de sillon à insérer, temps d'espacement
Données entrée facultatives	Cantons pénalisants
Données sortie	Nombre de sillons pouvant être insérés Horaire des sillons insérés
Mots clés	Planifications, sillons, timetable, horaires, trafic, graphique, capacité, réserve, BAL, espacement
Solution	

FIGURE 4.18 – Instanciation du patron au composant métier d'insertion d'un sillon dans une grille horaire

Déficiences : NON
Formation et habiletés : • Niveau de formation : universitaire ou école d'ingénieurs • Vitesse de lecture : rapide • Niveau d'habileté au clavier : bonne
Travail et expérience : • Connaissance de la tâche : expert • Lieu de travail : bureau
Connaissance et utilisation du système : • Connaissance du système informatique : expert • Connaissance d'autres systèmes semblables à celui que l'on développe : bonne • Type d'utilisation du système : discrétionnaire • Fréquence d'utilisation du système : non définie
Connaissances informatiques • Générales : moyennes • Système d'exploitation : Windows • Outils pour Internet : courriels, engins de recherche • Famille de logiciels : bureautique
Aspects psychologiques : • Attitudes : positives • Motivation : élevée

TABLEAU 4.1 – Profil type des utilisateurs *"décideur"*

Déficiences : NON
Formation et habiletés : • Niveau de formation : universitaire • Vitesse de lecture : bonne, • Niveau d'habileté au clavier : bon
Travail et expérience : • Catégorie d'emploi : chargé d'études, • Connaissance de la tâche : intermédiaire, • Lieu de travail : bureau.
Connaissance et utilisation du système : • Connaissance du système informatique : intermédiaire, • Connaissance d'autres systèmes semblables à celui que l'on développe : moyenne • Type d'utilisation du système : discrétionnaire • Fréquence d'utilisation du système : non définie.
Connaissances informatiques • Générales : moyennes, • Système d'exploitation : Windows, • Outils pour Internet : courriels, engins de recherche, • Famille de logiciels : bureautique.
Aspects psychologiques : • Attitudes : positives, • Motivation : élevée.

TABLEAU 4.2 – Profil type des utilisateurs "chargés d'études ferroviaires intermédiaires"

Déficiences : NON
Formation et habiletés : • Niveau de formation : variable • Vitesse de lecture : moyenne • Niveau d'habileté au clavier : moyen.
Travail et expérience : • Catégorie d'emploi : chargé d'études, • Connaissance de la tâche : expert • Lieu de travail : bureau.
Connaissance et utilisation du système : • Connaissance du système informatique : intermédiaire • Connaissance d'autres systèmes semblables à celui que l'on développe : bonne, • Type d'utilisation du système : discrétionnaire, • Fréquence d'utilisation du système : non définie.
Connaissances informatiques • Générales : faibles, • Système d'exploitation : Windows, • Outils pour Internet : courriels, engins de recherche, • Famille de logiciels : bureautique.
Aspects psychologiques : • Attitudes : neutres, • Motivation : élevée.

TABLEAU 4.3 – Profil type des utilisateurs "chargé d'études ferroviaires experts"

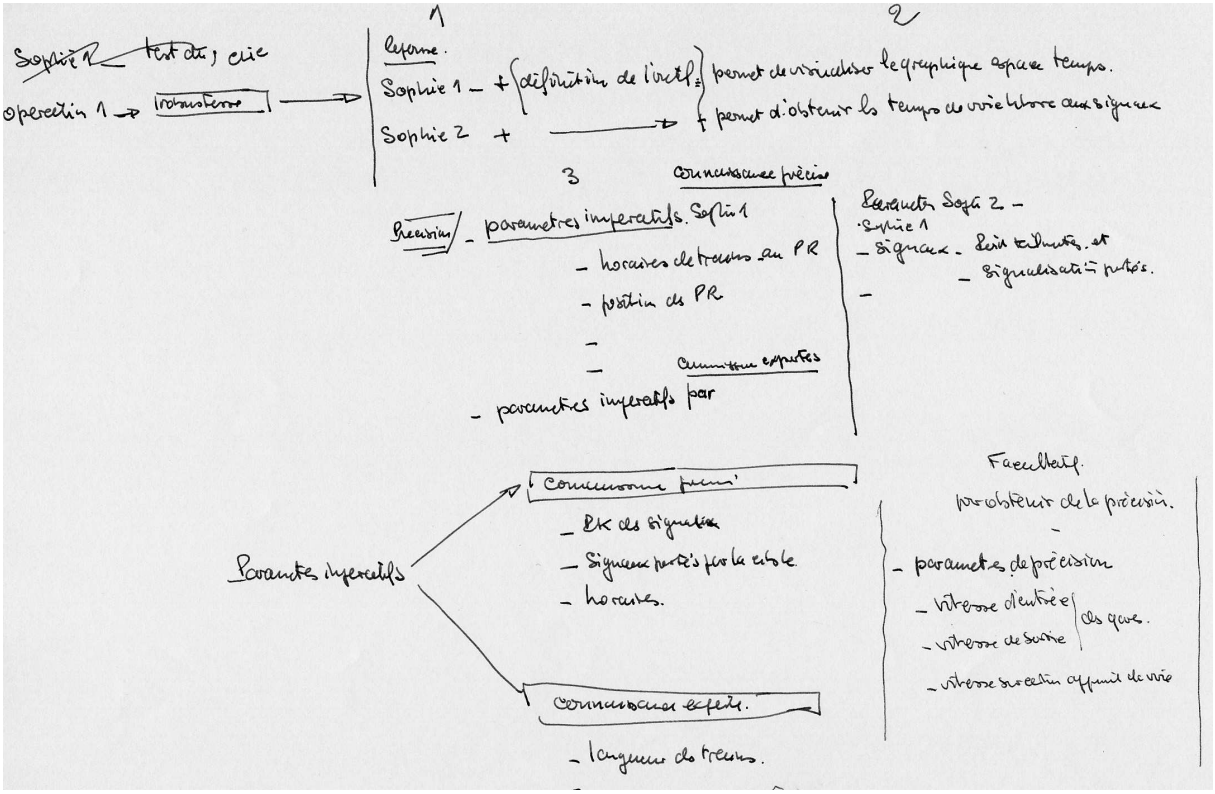


FIGURE 4.19 – Extrait d'un support de travail visant à expliciter les notions de "paramètres obligatoires" et "paramètres facultatifs"

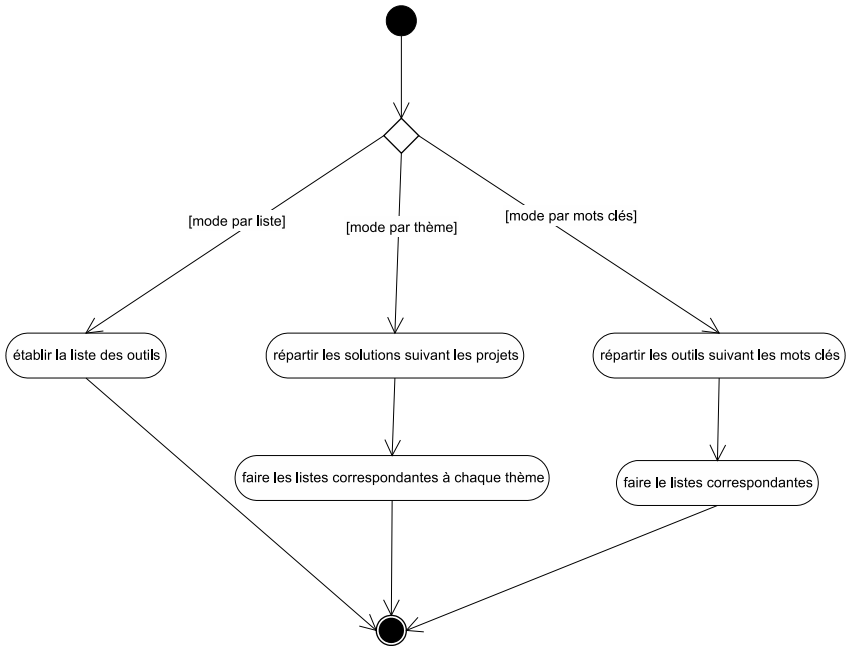


FIGURE 4.20 – Diagramme d'activités de la tâche automatique "Présenter les outils"

4.4.2.4 Modéliser les tâches interactives

Les tâches interactives sont à leur tour spécifiées. Elles sont modélisées à l'aide des diagrammes de séquence et de collaboration du langage UML. Ils dérivent les interactions entre l'utilisateur et le système pour réaliser la tâche, via l'interface homme-machine. Les tâches, qui nous intéressent dans la première itération du cycle, sont les tâches "Choisir le mode de présentation des outils", (Figure 4.21) et "Sélectionner les outils", (cf. figure 4.22).

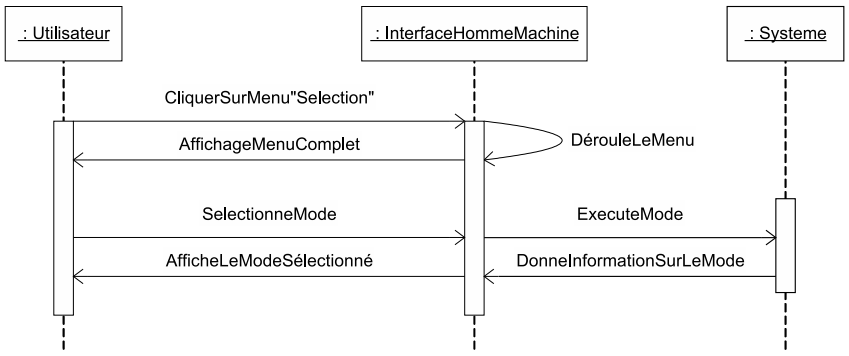


FIGURE 4.21 – Diagramme de séquence modélisant les interactions de la tâche "Choix du mode de présentation"

4.4.2.5 Choisir les modules d'aides

Cette partie a pour but de mettre en évidence les erreurs de fonctionnement possibles et de proposer des aides appropriées de manière à ce que les scénarios ne permettant pas à l'utilisateur de réaliser sa tâche soient écartés. La modélisation de SIADIF, notamment le diagramme des cas d'utilisation (Figure 4.13), montre que le chargé d'études doit utiliser un dossier correspondant à son étude. Le scénario de la Figure 4.23 montre l'utilisation correcte du système où l'utilisateur ouvre un dossier avant de demander un traitement. Dans l'autre scénario, Figure 4.24, l'utilisateur demande un traitement alors qu'aucun dossier n'est ouvert. Le système lui enverra donc un message d'erreur lui indiquant qu'il est nécessaire d'ouvrir ou de créer un dossier pour faire un traitement. Une fois le dossier ouvert, il aura la possibilité de demander le traitement.

4.4.2.6 Spécifier les Interfaces Homme-Machine

Nous avons utilisé le diagramme état-transition de UML, cf. Figure 4.25, pour définir le nombre d'écrans à utiliser, l'enchaînement des vues, les modalités de dialogue homme-machine. Ce passage doit être conforme aux relations temporelles et structurelles du modèle de la tâche (issu de la modélisation des tâches interactives). De cette manière, le comportement de l'interaction homme-machine est spécifié de façon non ambiguë et cohérente. Pour chacune des vues, les modes de présentation des informations sont définies. Les modes d'activation des différents outils d'aide sont déterminés.

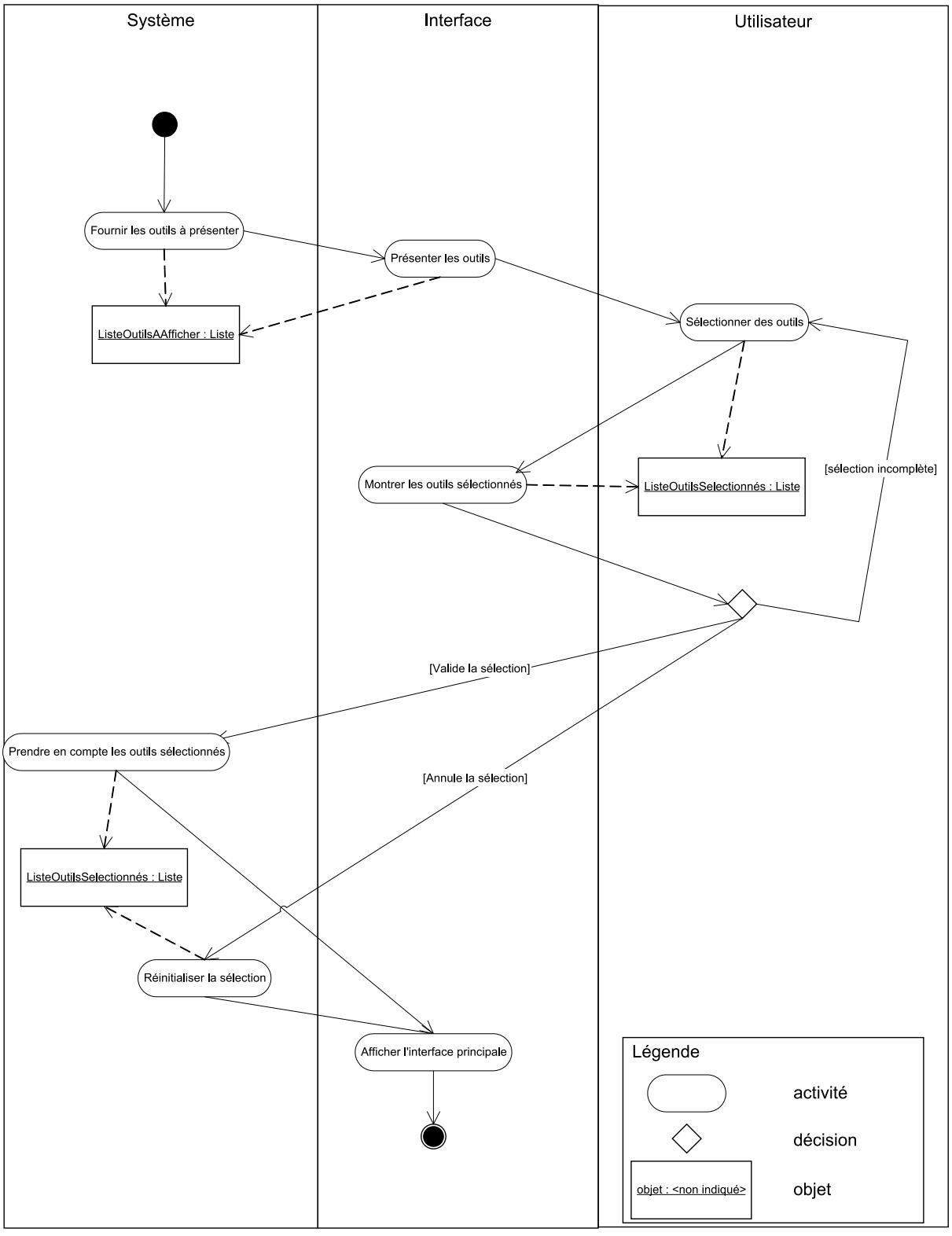


FIGURE 4.22 – Diagramme d'activités modélisant la dynamique de la tâche interactive "Sélectionner les outils"

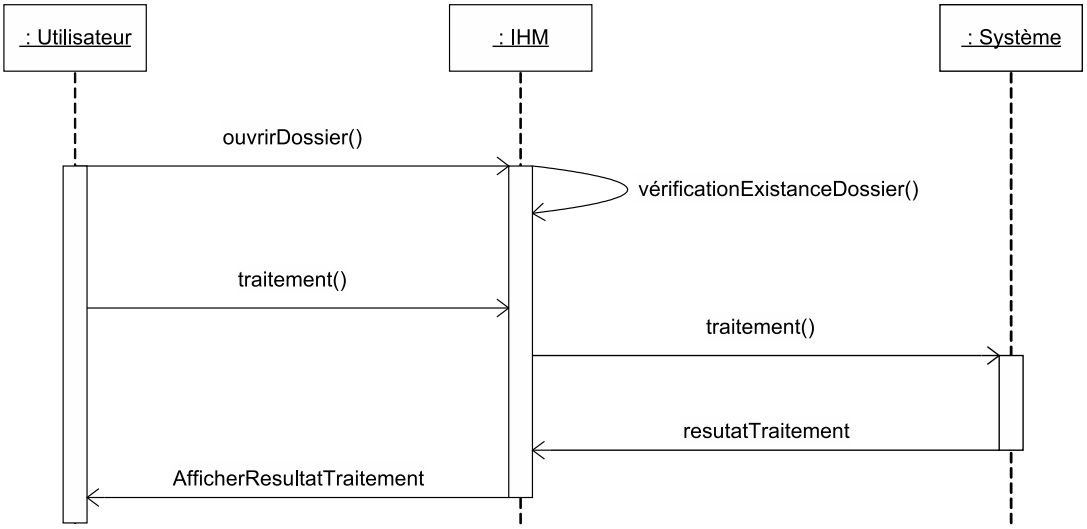


FIGURE 4.23 – Scénario d'utilisation correcte du système

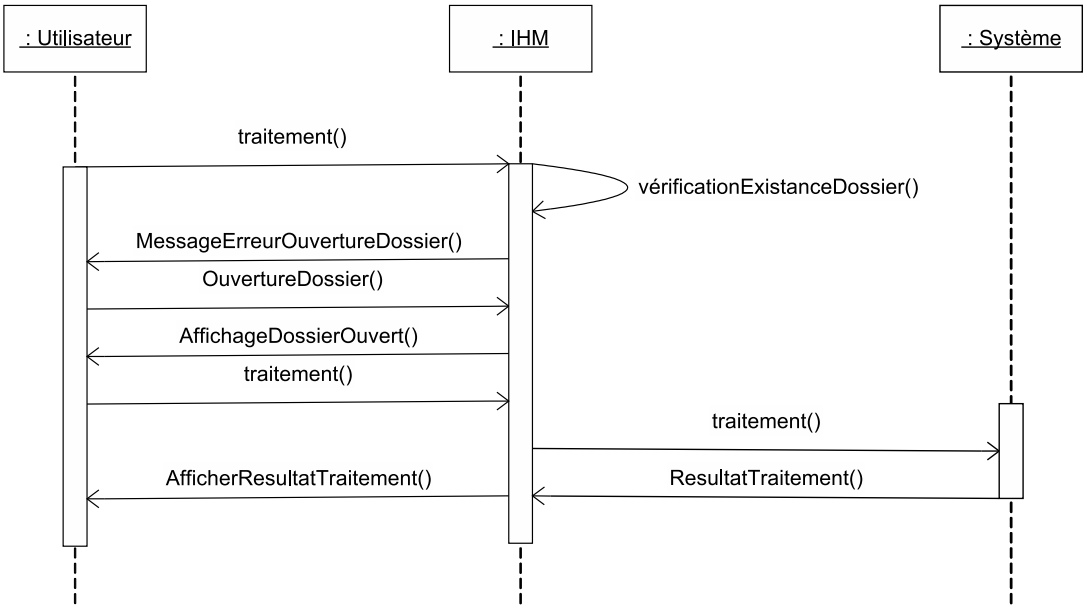


FIGURE 4.24 – Scénario de mauvaise utilisation du système

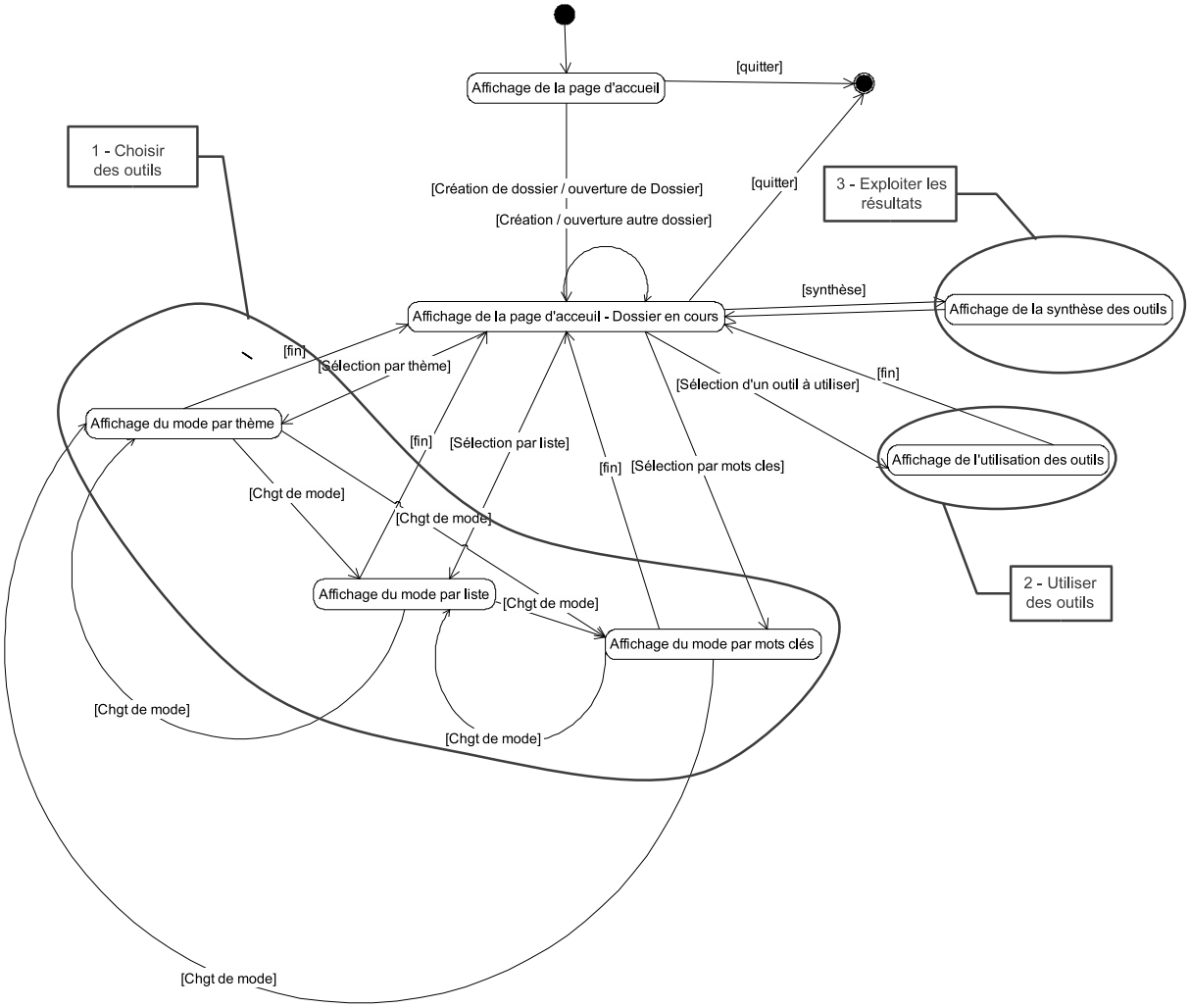


FIGURE 4.25 – Diagramme états-transitions pour la spécification des IHM

4.4.2.7 Faire le maquettage des IHM

Le maquettage a été fait en plusieurs temps. Le premier maquettage correspond à la présentation des outils dans le mode "Par thèmes" (cf. Figure 4.26) et la synthèse des résultats (cf. Figure 4.27) ; il a été réalisé à l'aide d'un éditeur graphique, nous l'avons nommé "maquettage statique". Après quelques réunions et une validation globale, le second maquettage a été réalisé à l'aide d'un langage de programmation (java) (Figure 4.28). Le maquettage des interactions de l'ensemble du système a été mené pour faciliter l'évaluation avec les utilisateurs des spécifications effectuées précédemment, nous l'avons appelé "maquettage dynamique". Le mélange des deux techniques (pages écran modifiées à l'aide d'un éditeur graphique) a été utilisé notamment pour illustrer la présentation de la synthèse des résultats (cf. Figure 4.29), ce que nous appelons "maquettage mixte".

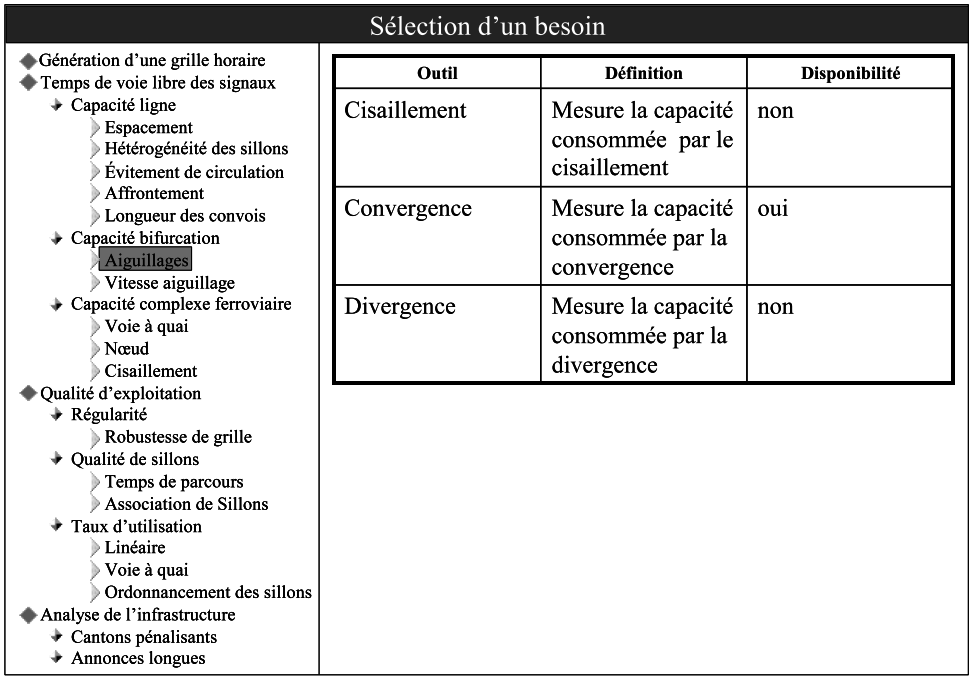


FIGURE 4.26 – Maquettage "statique" de l'affichage concernant le choix d'un outil

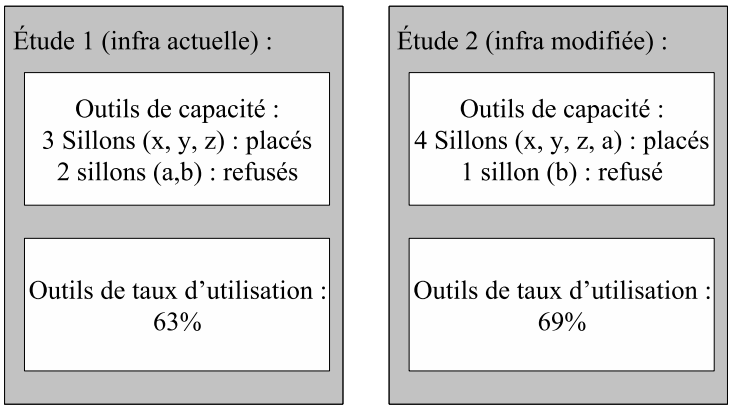


FIGURE 4.27 – Maquettage "statique" de l'affichage concernant la synthèse des outils

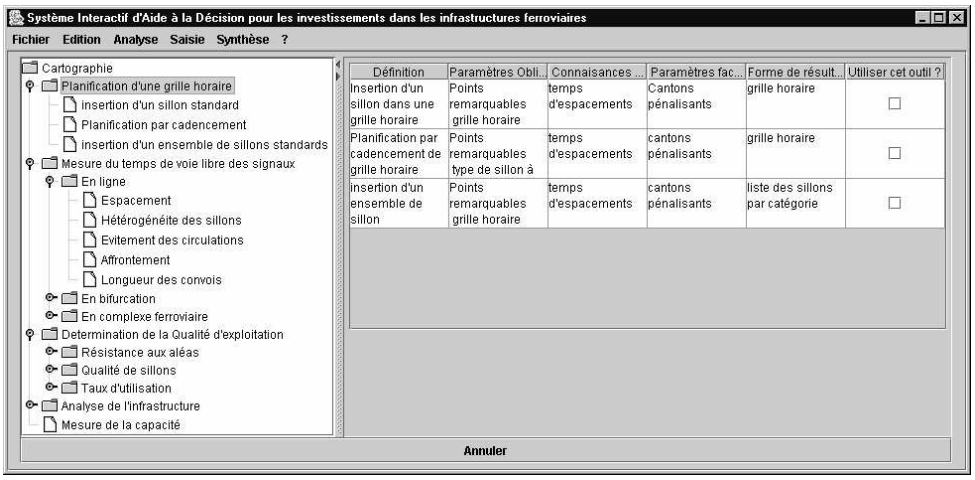


FIGURE 4.28 – Maquettage "dynamique" de la sélection d'outil selon le mode par thèmes

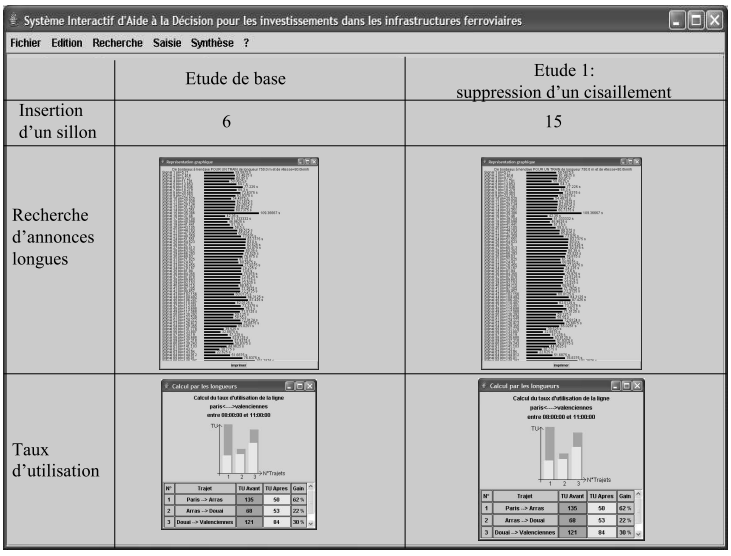


FIGURE 4.29 – Maquettage "mixte" de l'affichage correspondant à la synthèse

4.4.3 Etape de conception préliminaire

Pour respecter un processus incrémental, les fonctionnalités sont classées de manière à être intégrées par ensembles successifs. Cette classification se fait avec la participation des utilisateurs. Dans le cas applicatif de SIADIF, ces ensembles sont formés par les tâches de présentation des outils, d'utilisation des outils et de synthèse des résultats.

SIADIF a été bâti autour d'une architecture favorisant une programmation par composant conformément au développement dans ADESIAD (cf. chapitre 3 § 3.4.2). L'architecture adoptée, représentée par la Figure 4.30, montre que le SIAD est indépendant des composants métier. Les composants métier (ou de domaine) doivent respecter les interfaces requises afin d'être intégrés à n'importe quel moment du cycle de vie du système SIADIF. Ces composants métier correspondent aux descriptions données précédemment.

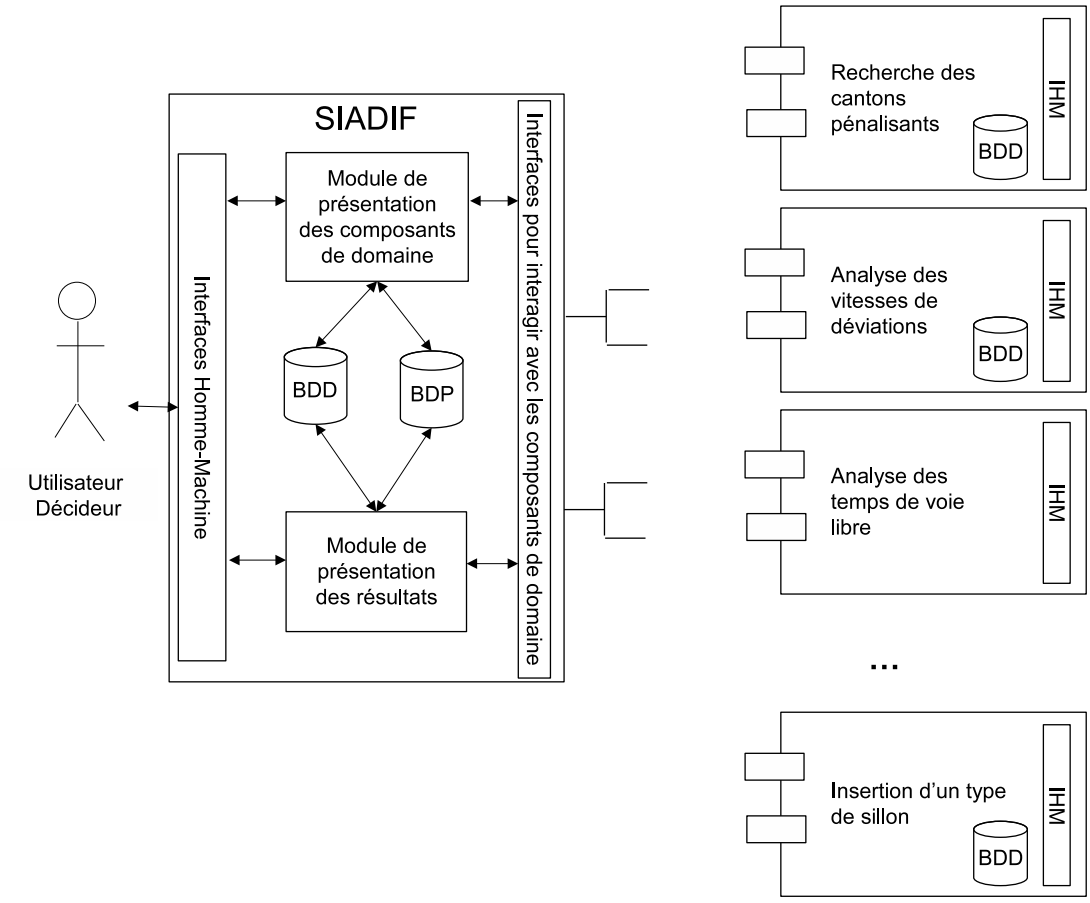


FIGURE 4.30 – Architecture de SIADIF

Nous avons utilisé divers composants de conception (design patterns), par exemple, le pattern *Adaptateur* (cf. Tableau 4.4), le pattern *Pont* (cf. Tableau 4.5) et le pattern *Itérateur* (cf. Tableau 4.6). Le pattern *Adaptateur* permet à un client d'utiliser un ensemble d'objets de la même façon. Ce patron sera utilisé notamment pour intégrer les outils sous la forme de composants. Ils seront tous vus de la même façon alors qu'ils sont différents. Le patron Pont permet de faire varier l'abstraction de l'implémentation. Ce patron est utilisé également pour permettre d'utiliser divers outils (implémentation) pour répondre à un besoin (abstraction). Les besoins associés au problème ont été reliés à l'implémentation en partie analyse mais pour faciliter l'évolution, ils doivent être indépendants en programmation. Le patron *Itérateur* a été utilisé pour les parcours de liste.

Le patron de conception que nous avons le plus utilisé pour les Interfaces homme-machine est le pattern *Décorateur*. Il permet d'associer dynamiquement des responsabilités supplémentaires à un objet. Il permet l'extension de la fonctionnalité d'un objet sans avoir recours à des sous-classes. Son utilisation est illustrée par le diagramme de classe représenté en Figure 4.31. La composition de plusieurs décorateurs (par exemple *DécorateurVueTexte*, *DécorateurDefile*, et *DécorateurBord*) permet de former un composant VueTexte ayant une bordure et un ascenseur.

Objectif	Faire correspondre à une interface donnée un objet existant que vous ne contrôlez pas.
Problème	Un système a les bonnes données et les bonnes méthodes, mais la mauvaise interface. Généralement utilisé lorsque vous devez créer des dérivées d'une classe abstraite en cours de définition ou déjà définie.
Solution	L'adaptateur fournit un encapsuleur avec l'interface voulue.
Participants et collaborateurs	La classe *Adaptateur* adapte l'interface à la classe *Adaptée* pour qu'elle corresponde à celle de la Cible de l'*Adaptateur* (c'est-à-dire la classe à partir de laquelle elle est dérivée). Cela permet au *Client* d'utiliser la classe *Adaptée* comme s'il s'agissait d'un type de *Cible*.
Conséquences	Grâce au pattern adaptateur, des objets existants peuvent être intégrés à de nouvelles structures de classes sans être limitées par leur interface.
Implémentation	Intégrer la classe existante dans une autre classe. La classe qui encapsule doit être compatible avec l'interface voulue et appeler les méthodes de la classe encapsulée.

TABLEAU 4.4 – Le pattern Adaptateur : caractéristiques (tirées de [Shalloway and Trott, 2002])

Objectif	Découpler un jeu d'implémentations à partir du jeu d'objets qui l'utilise.
Problème	Les dérivations d'une classe abstraite doivent utiliser plusieurs implémentations sans provoquer une explosion du nombre de classes.
Solution	Définir une interface pour toutes les implémentations voulues et la faire utiliser par les dérivations de la classe abstraite.
Participants et collaborateurs	*Abstraction* définit l'interface des objets en cours d'implémentation. *Implémenteur* définit l'interface des classes d'implémentation sans savoir quel *ImplémenteurConcret* est employé.
Conséquences	Le découplage des implémentations à partir des objets les utilisant accroît les possibilités d'extension. Les objets clients ignorent les problèmes d'implémentation.
Implémentation	Encapsuler les implémentations dans une classe abstraite. Contenir le descripteur de la classe dans la classe de base de l'abstraction implémentée. Remarque : en java : vous pouvez utiliser des interfaces au lieu d'une classe abstraite pour l'implémentation.

TABLEAU 4.5 – Le pattern Pont : caractéristiques (tirées de [Shalloway and Trott, 2002])

Objectif	Fournit un moyen d'accès séquentiel, aux éléments d'un agrégat d'objets, sans mettre à découvert la représentation interne de celui-ci.
Problème	Offrir un moyen d'accès à des agrégats sans risques pour sa structure interne. Fournir des possibilités de parcours variés sans alourdir la classe de l'agrégat.
Solution	Définir une classe *Itérate*ur qui soit indépendante de l'agrégat à parcourir. L'agrégat devra créer son *Itérateur*.
Participants et collaborateurs	*Itérateur*, *ItérateurConcret*, *Agregat*, *AgregatConcret*. Utilise le pattern *Fabrication*.
Conséquences	Il permet des modifications dans le parcours des agrégats. Les *Itérateurs* simplifient l'interface de l'agrégat. Il peut y avoir plus d'un parcours en cours simultanément sur un agrégat.
Implémentation	Extraire les fonctions en charge d'accès et parcours de l'agrégat pour les placer dans un objet *Itérateur*.

TABLEAU 4.6 – Le pattern Itérateur : caractéristiques (tirées de [Gamma et al., 1999])

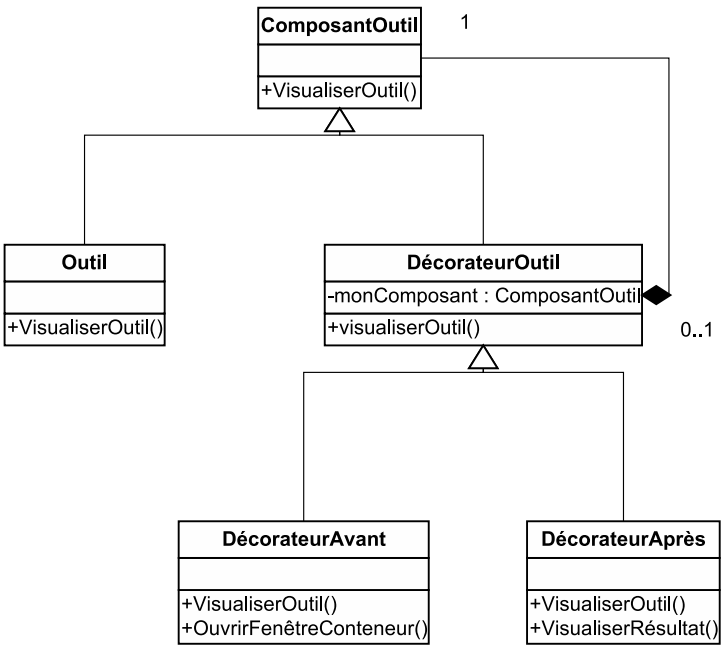


FIGURE 4.31 – Utilisation du pattern *Décorateur* pour l'affichage des outils

4.4.4 Etape de conception détaillée

L'étape de conception détaillée permet de formaliser les spécifications des besoins pour arriver aux spécifications des algorithmes qui seront nécessaires. Par exemple, la gestion des différents modes d'affichages, la mise à jour des listes présentant les outils sélectionnés ou utilisés, les sauvegardes selon un projet. Par exemple, chacune des méthodes permettant de réaliser l'activité "affichage du mode de recherche par thème" défini en Figure 4.20, est visible sur le diagramme de séquence de la Figure 4.32 et est définie pour être développée dans l'étape suivante. Pour cela, les composants de conception définis dans l'étape précédente sont utilisés. Cette étape débouche sur la recherche des composants de code fonctionnels mais aussi sur la recherche des composants de code spécifiques aux interfaces homme-machine. On retiendra par exemple les composants de code permettant de gérer les sauvegardes, la création de fichier, les listes... Les composants IHM ont été tirés de l'API Swing de java[20] pour permettre de gérer les tableaux, les arbres permettant de visualiser une arborescence, les boutons, etc. Les spécifications de cette étape doivent être validées avant d'être transmises à l'étape d'implémentation en vue d'être développées. Cette validation ne fait pas obligatoirement intervenir les experts et utilisateurs car ils n'ont pas les connaissances nécessaires en informatique pour cela. Cependant, ils peuvent participer à la validation.

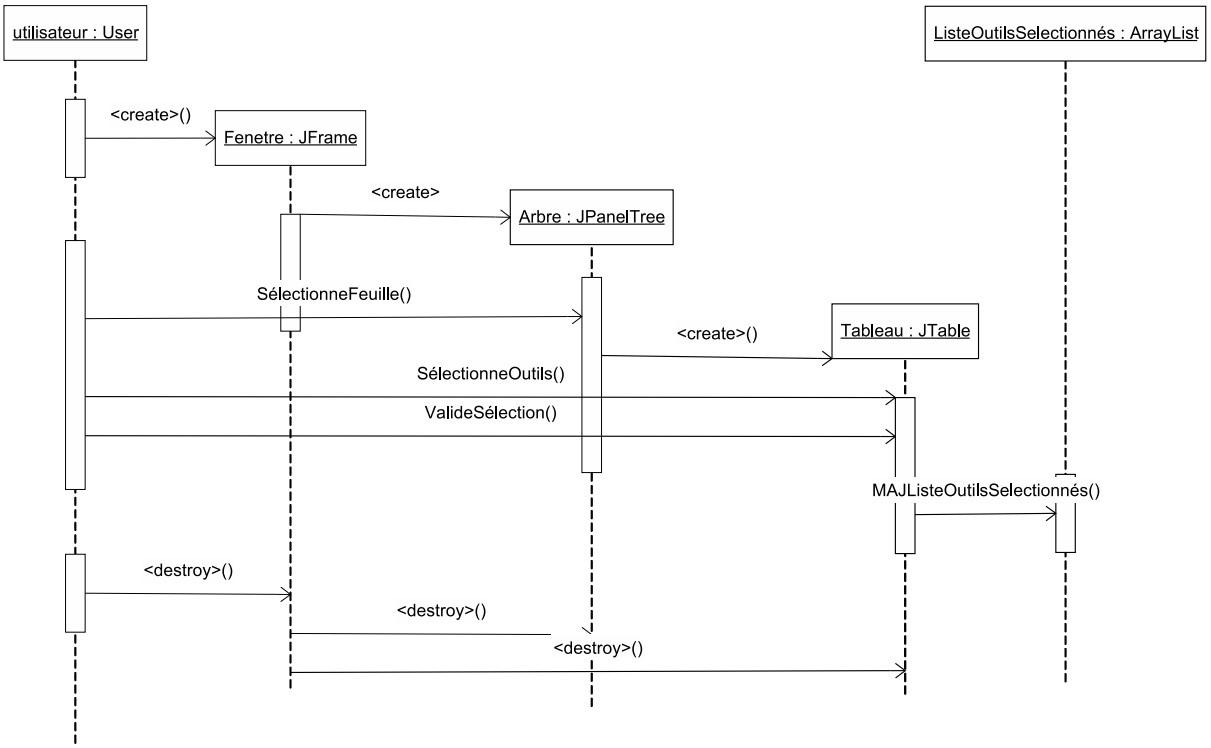


FIGURE 4.32 – Diagramme de séquence montrant les méthodes à développer pour réaliser l'activité "Afficher le mode par thème"

[20] Adresse Internet : http ://java.sun.com/docs/books/tutorial/uiswing/components/components.html

4.4.5 Etape d'implémentation et tests unitaires

A cette étape la majorité des algorithmes sont développés dans le langage Java. Ils seront par la suite testés avant d'être assemblés. Nous avons choisi Java car c'est un langage courant, bien documenté et qui apportait les solutions techniques nécessaires à nos besoins.

Nous avons longtemps hésité entre l'utilisation des EJB et de Jini pour l'intégration des composants métier. L'intérêt de Jini, classé comme gestionnaire de services, est qu'il permet l'intégration des outils (services) durant l'exécution, contrairement aux EJB qui doivent être intégrés en phase de développement. A ce jour, nous avons choisi les EJB pour l'intégration de composants métier. Le SIAD est donc développé en java et est client des composants métier qui sont eux considérés comme serveurs. Les interfaces nécessaires à la composition des composants métier avec le SIAD est encore à l'étude et fait partie de nos perspectives de recherche.

4.4.6 Etapes de validation des composants

Les composants sont assemblés pour fournir un prototype. La première étape de validation consiste à vérifier que les composants de code choisis correspondent bien au besoin. La page écran, montrée en Figure 4.28, présente les outils de mesure selon le mode par thème ; l'interface a nécessité l'utilisation des composants Swing : *JFrame*, *JButton*, *JPanel*, *JSplitPane*,... Dans la seconde étape, ce sont les composants de conception qui doivent être validés. Les problèmes, auxquels l'équipe de développement a été confrontés en phase de conception, doivent être résolus par les composants ; il faut vérifier qu'ils le sont effectivement et qu'il n'y avait pas de "meilleur moyen". La troisième étape de validation des composants concerne les composants métier. Dans cette première itération, les composants métier associés au SIAD n'ont pas été intégrés. Cependant certains ont été validés par les utilisateurs comme étant en correspondance par rapport au but recherché.

4.4.7 Etape de validation de SIADIF avant intégration des composants métier

Comme nous l'avons expliqué dans la chapitre précédent (cf. § 3.3.7), cette étape a deux objectifs : tester l'intégration du système (hors composants métier) et valider le système sans la prise en compte des composants métier.

Le premier objectif consiste à vérifier que les composants "jouent" le rôle qui leur a été confié sans "effet de bord". D'une part, le testeur doit vérifier que chaque composant se comporte comme convenu (vérifier que le composant est bon). D'autre part, il faut vérifier qu'il correspond bien à la tâche qu'il doit remplir (vérifier que c'est le bon composant).

Le second objectif consiste à vérifier que le système permet à l'utilisateur de mener à bien les tâches inclues dans son processus de décision. Elle permet d'évaluer le respect du processus de décision sans les perturbations possibles induites par des composants métier mal adaptés. On évalue l'utilité et l'utilisabilité du système sans l'intervention des composants métier.

4.4.7.1 Protocole d'évaluation

D'une part un questionnaire basé sur les critères de Bastien et Scapin[21] (cités en § 3.3.7) a été utilisé pour évaluer l'utilisabilité du système et, d'autre part, l'utilité du système a été évaluée par une mise en situation de futurs utilisateurs. Les documents relatifs à l'évaluation sont visibles en annexe.

Le protocole d'évaluation était centré sur l'évaluation de l'interaction et l'influence des modes de présentation des outils sur les utilisateurs. Pour mener à bien cette évaluation, les futurs utilisateurs ont été mis en situation la plus réelle possible. L'évaluation s'est déroulée en quatre étapes :

- Présentation de SIADIF : un diaporama d'une durée d'environ 5 minutes montre à chaque utilisateur le but du système et les différents modes qui lui sont proposés.
- Familiarisation de l'utilisateur avec SIADIF : chaque utilisateur analyse les réactions du système lors de son utilisation. Le système est utilisé par l'utilisateur avec de l'aide éventuelle provenant de l'évaluateur ; il donne ses impressions au fur et à mesure.
- Première mise en situation : un problème d'investissement, choisi parmi les problèmes de CPER (Contrat de Plan Etat Région) du Nord - Pas de Calais lui a été proposé. Il devait d'abord indiquer sa situation par rapport à ce problème et son niveau d'expertise. Ensuite, il devait utiliser le SIAD pour choisir les outils adaptés à son problème. Il a dû répéter cette opération trois fois (ce qui correspond au nombre de modes de présentation existant : par liste, par thème et par mots-clés) en choisissant un mode de présentation et n'utilisant que celui-là. A la fin de chaque sélection, chaque utilisateur devait remplir un questionnaire sur son impression par rapport au mode utilisé et les résultats obtenus. A la fin de cette partie, il devait aussi remplir un questionnaire concernant un ensemble de points ergonomiques relatifs à l'IHM, énoncés selon les critères de Bastien et Scapin [Shackel, 1991, Grudin, 1992, Farenc, 1997, Bastien and Scapin, 2001].
- Autres mises en situation : après une pause, deux autres problèmes d'investissement touchant d'autres points ferroviaires, issus également du CPER, ont été proposés à chaque utilisateur. Ils ont alors utilisé les trois modes selon un ordre établi.

Les documents recueillis à l'issu de l'évaluation sont les questionnaires sur les problèmes d'investissement ferroviaire, le questionnaire d'évaluation ergonomique de l'IHM et l'enregistrement sur vidéo (image et son) contenant l'ensemble des actions exécutées par l'utilisateur. Celui-ci a été enregistré à partir de la première mise en situation jusqu'à la fin de l'expérimentation.

Trois utilisateurs de profil différents et représentatifs ont participé à ces évaluations. Nous n'avons pas souhaité faire participer plus de monde car ces évaluations ne sont pas terminales ; l'évaluation concerne le SIAD sans composant métier à l'issue de la première itération dans le cycle. Nous avons sélectionné trois utilisateurs : un chargé d'études ferroviaire (expert, 30 ans d'expérience), un chargé d'étude ferroviaire (débutant, 2 ans d'expérience), un utilisateur ayant des notions issues du domaine ferroviaire mais ne travaillant pas directement dans le domaine de l'expertise ferroviaire (nous l'appellerons utilisateur novice).

La section suivante présente les résultats suivants deux visions : l'avis des utilisateurs et les résultats objectifs.

[21] Ils ont été choisis dans la mesure où ils sont largement reconnus dans la communauté francophone.

4.4.7.2 Résultats

Le système propose dans sa version actuelle l'accès à dix-sept outils susceptibles de contribuer à des études d'investissement ferroviaire, proposés par des experts du domaine lors des précédentes étapes et développés avec eux en parallèle du SIAD. Les premiers résultats présentés concernent le choix du mode de recherche en fonction des utilisateurs.

Sur le Tableau 4.7, on constate que l'utilisateur considéré comme expert a préféré commencer l'étude avec le mode de recherche par liste car il ne lui semblait pas utile d'être aidé par les autres mécanismes. L'utilisateur débutant a choisi le mode thème car il a apprécié la structure arborescente le guidant dans le choix des outils en fonction du problème ; d'après cet utilisateur, ce mode a l'avantage de lui éviter d'oublier des outils. Le troisième utilisateur, dit novice, a utilisé le mode mots-clés car il lui semblait lui permettre de trouver une sélection à son problème. A la fin des études, l'utilisateur expert gardera sa préférence pour le mode Liste, le débutant celle pour le mode Thème et l'utilisateur novice celle pour les modes par liste et par mots-clés. On notera que l'utilisateur novice a été gêné lors de l'utilisation de SIADIF avec le mode thème car des outils ont été proposés plusieurs fois dans des thèmes différents. Les autres utilisateurs n'ont pas été gênés et ont été globalement en accord avec les propositions de ce mode.

	Mode *Liste*	Mode *Thème*	Mode *Mots-clés*	Préférence finale concernant les trois modes
Utilisateur expert	1	3	2	Mode *Liste*
Utilisateur débutant	2	1	3	Mode *Thème*
Utilisateur novice	2	3	1	Mode *Liste* ou *Mots-clés*

TABLEAU 4.7 – Tableau illustrant l'ordonnancement du choix de mode par chaque utilisateur

A chaque fin d'utilisation selon un mode, il a été demandé aux utilisateurs de quantifier leur satisfaction par rapport à la sélection d'outils faite. Il en ressort que les trois utilisateurs ont jugé les résultats sortants du mode "recherche par *liste*" comme sensiblement meilleurs par rapport aux deux autres modes, cf. Figure 4.33. On remarque par ailleurs que les utilisateurs expert et débutant n'ont pas apprécié les résultats sortis de l'usage du mode par *Mots-clés*. Or pour l'utilisateur expert les outils sélectionnés dans le mode *Mots-clés* sont plus nombreux que dans les autres modes (cf. Figure 4.34 qui montre les outils sélectionnés parmi les 17 proposés par utilisateur en fonction du mode). Le mode a donc correctement proposé les outils (sans manque). Concernant l'utilisateur débutant la sélection est rigoureusement la même entre le mode *Liste* et le mode *mots-clés*. Pour cet utilisateur, la différence de sélection entre l'outil 1 et 3 est dûe à un changement d'avis concernant la stratégie à suivre pour résoudre le problème, indépendant de l'utilisation du système. La seule différence dûe au mode de recherche se situe au niveau d'un outil non sélectionné lors de l'utilisation du mode *Thème* car celui-ci n'était pas proposé. A contrario, l'outil 11 n'a pas été sélectionné par l'utilisateur expert lors de l'utilisation du mode *Thème* alors que ce mode le proposait ; c'est donc un problème de reconnaissance. L'outil ne devait pas être présent

là où l'utilisateur s'attendait à le trouver. C'est principalement pour cette raison que l'utilisateur expert est moins satisfait des résultats obtenus à l'aide du mode *Thème*.

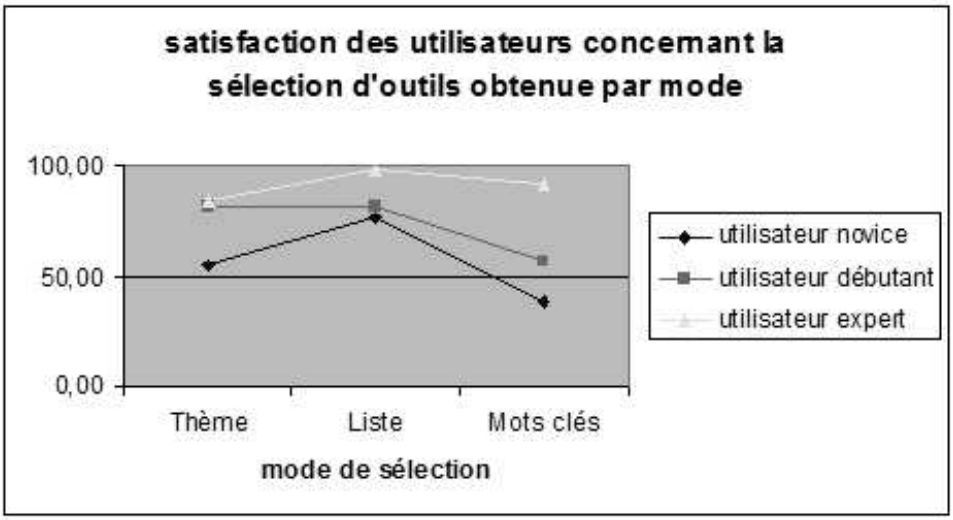


FIGURE 4.33 – Degré de satisfaction des résultats par rapport aux modes utilisés

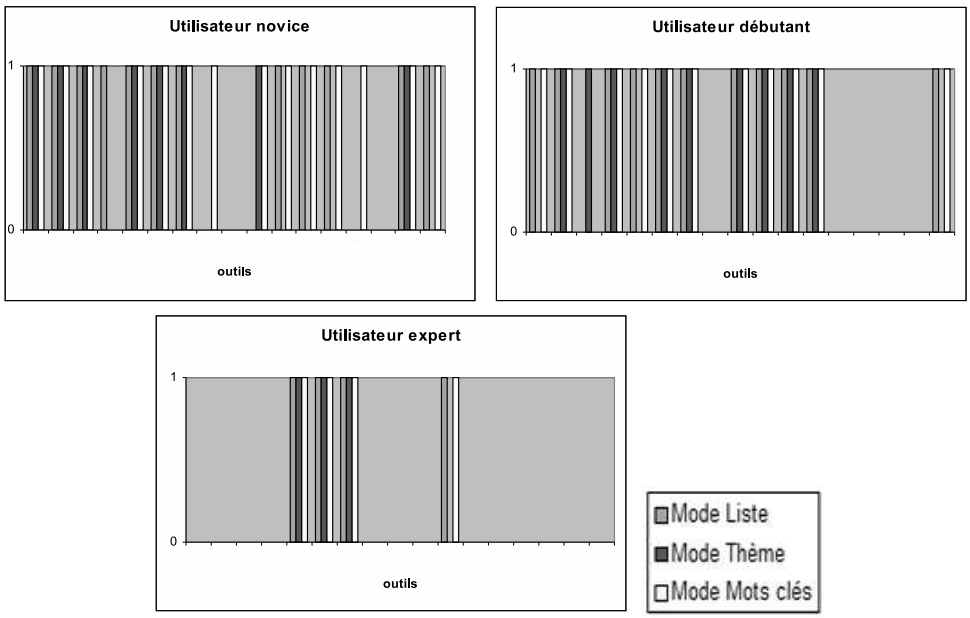


FIGURE 4.34 – Sélection des outils par chaque utilisateur

Si on prend en compte la sélection d'outils (parmi les dix sept proposés) suivant le mode qu'ils ont préféré, la Figure 4.35 montre qu'au total, sur le premier problème posé, 14 outils ont été sélectionnés. En détaillant, l'utilisateur expert en a sélectionné 4, les autres (débutant et novice) en ont sélectionnés 11 mais différents. La diversité des sélections faites par les utilisateurs montre qu'il est important de dissocier les outils et non de les rassembler sous une forme unique car cela leur permet de sélectionner réellement les outils en fonction de leur besoin.

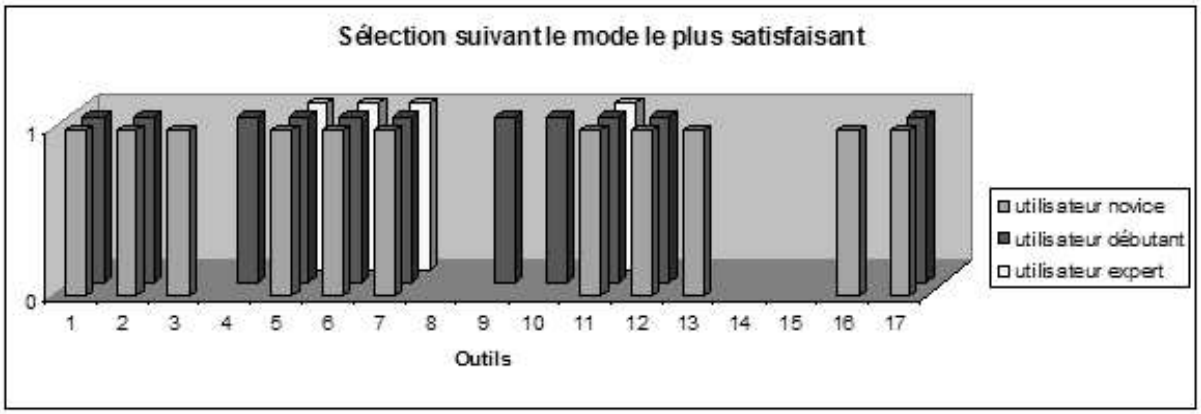


FIGURE 4.35 – Diversité des sélections selon les utilisateurs

D'un point de vue utilisabilité, le principal reproche concernait le manque de retour d'information particulièrement dans le mode thème. Le mode thème proposait les outils dans plusieurs tableaux. Des outils pouvaient se retrouver dans plusieurs thèmes. Le manque de retour d'information obligeait les utilisateurs à se souvenir ou à prendre en note les outils au fur et à mesure de la sélection. Cela s'est répercuté sur leur notation dans la grille d'utilisabilité, cf. Figure 4.36 concernant le guidage.

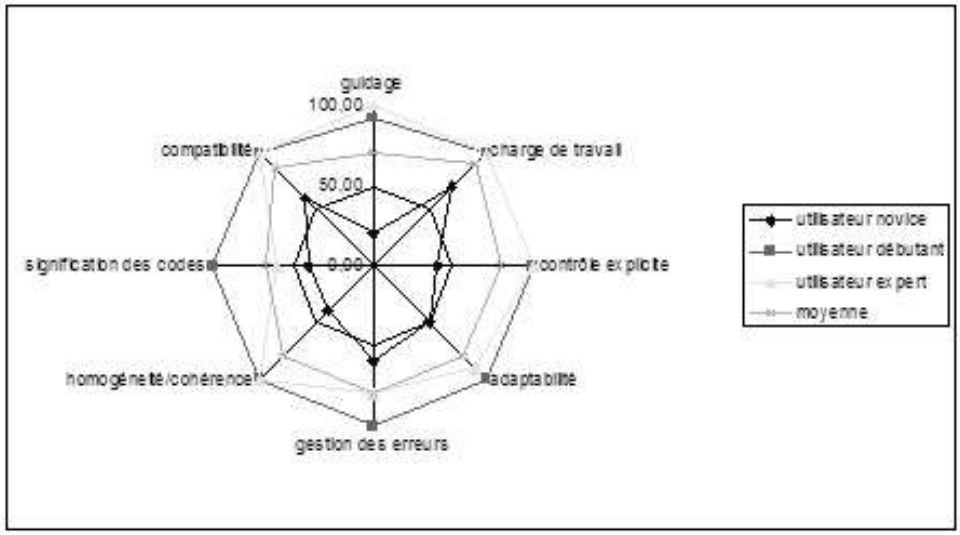


FIGURE 4.36 – Résultats globaux relatifs à l'utilisabilité du système

Une autre remarque négative provenant d'un utilisateur (utilisateur débutant) porte sur la forme des cases des tableaux fortement allongées qui oblige une lecture "verticale". Cette remarque était subjective car les autres utilisateurs ont apprécié la présentation. De plus les cases du tableau étaient redimensionnables donc cette remarque aurait pu être évitée.

Les utilisateurs ont été globalement satisfaits de l'interaction avec le système. On remarque que l'utilisateur novice s'est plus consacré à l'évaluation de l'utilisabilité qu'à l'utilité puisqu'il n'était pas réellement du domaine. Les autres utilisateurs (débutant et expert) ont été satisfaits par l'utilité de l'outil

et n'ont pas porté beaucoup d'importance à l'utilisabilité (les défauts leur semblaient mineurs), ce qui justifie l'écart entre les deux jugements.

En moyenne le critère ayant la plus petite moyenne est "la signification des codes" avec 2/3. Pour les utilisateurs les codes correspondaient aux énoncés des problèmes et des outils. C'est donc ce point qui devait être amélioré. Les meilleurs résultats concernent les critères sur la charge de travail et la compatibilité.

4.4.8 Conclusion sur le développement de SIADIF

A l'issue de la première itération, le système, partiellement terminé, a été accepté par les futurs utilisateurs. Ils ont apprécié l'offre en outils proposée par le système. L'idée de proposer un panel d'outils pour résoudre leur problème leur a plu. Ils disposent de cette manière d'un ensemble d'outils.

Malgré l'implication d'utilisateurs dont certains sont des experts de RFF au cours du cycle de développement, quelques défauts dans les IHM ont tout de même été remarqués. Ceux-ci n'avaient pas été pensés dans l'analyse globale du système. Ces défauts sont mineurs et ont rapidement été corrigés. Ils n'ont pas été la cause de rejet de SIADIF de la part des utilisateurs. La détection de ces défauts montrent l'intérêt d'une évaluation à la fin d'un cycle et montre également que les évaluations centrales seules ne seraient pas suffisantes.

Les besoins concernant les informations des présentations des outils aux utilisateurs, ont évolué. En effet, dans la forme actuelle, chaque outil est présenté avec son nom, ses paramètres et une description. Les utilisateurs ont suggéré qu'il y ait plus d'informations concernant ces outils de manière à faciliter la sélection et à former les utilisateurs les moins aguerris.

4.5 Conclusion

ADESIAD, Approche de Développement de SIAD proposée dans le cadre de ce mémoire a été appliquée à un cas concret dans le domaine ferroviaire pour le développement de SIADIF. Ce système est développé en collaboration avec RFF (Réseau Ferré de France), propriétaire du réseau ferroviaire français. Nous avons vu que le système a été partiellement développé et accepté par des utilisateurs représentatifs des utilisateurs finaux. La partie importante de décomposition du problème pour aboutir à un ensemble d'outils d'analyse correspond à leur besoin. Cette décomposition a également permis de fournir un support dans le choix d'outils. L'évaluation sans composants métier a montré que le système était correct du point de vue de l'utilité et de l'utilisabilité. Le système n'étant pas encore complet, il est difficile de conclure sur le respect du processus humain sur l'ensemble du processus de décision. *A contrario*, nous pouvons conclure qu'il respecte le processus pour la partie de sélection d'outils. La validation de SIADIF va permettre de valider certaines caractéristiques d'ADESIAD, dans le chapitre suivant. D'autres validations feront partie de nos perspectives. Celles-ci feront l'objet du chapitre suivant.

Chapitre 5

Evaluation globale d'ADESIAD et perspectives de recherche

Sommaire

5.1 Introduction

Ce chapitre propose un bilan global concernant ADESIAD ainsi que des perspectives de recherche. La première partie présentera, à partir du développement montré dans le chapitre précédent un bilan global de l'approche. Plusieurs points seront détaillés et les perspectives de recherche associées seront exposées. La seconde partie du chapitre sera axée sur les perspectives de recherche concernant l'apprentissage du système mais aussi des utilisateurs grâce aux SIAD, la personnalisation des informations appliquée aux SIAD et pour faciliter le transfert de compétence l'intégration de la collaboration aux SIAD. D'autres techniques voisines des techniques de composants telles que les fédérations d'outils ou les services seront également envisagées tout en soulignant leur intérêt potentiel à être intégrées aux SIAD.

5.2 Evaluation globale de l'approche ADESIAD

L'objectif de cette partie est de mettre en avant quelques résultats provenant de l'évaluation globale d'ADESIAD. Elle comprend trois sections. La première présente un bilan global d'ADESIAD en fonction des objectifs fixés. Ensuite, l'évaluation porte sur les répercussions de l'approche par rapport à l'utilisation de SIADIF en fonction du degré d'expertise des utilisateurs. Ensuite, l'apport des composants métier pour la prise de décision sera discuté et les perspectives permettant de valider certaines de nos hypothèses seront présentées.

5.2.1 Bilan d'ADESIAD par rapport aux objectifs fixés *a priori*

Le bilan de l'approche ADESIAD se base sur l'application concrète (SIADIF) présentée précédemment et sur les résultats de l'évaluation de SIADIF à la fin de la première itération. Pour chaque objectif d'ADESIAD, nous donnerons nos conclusions selon les constats établis durant l'utilisation d'ADESIAD lors du développement de SIADIF. Les trois parties d'ADESIAD seront traitées : le modèle, la méthode, l'architecture. Enfin, les objectifs globaux d'ADESIAD sont présentés ; de la même façon des constats et conclusions sont exprimées.

5.2.1.1 Bilan du modèle d'ADESIAD

Le principal objectif d'ADESIAD était d'être adaptée aux types de SIAD visés dans le cadre de cette recherche. Pour cela, une phase d'évaluation placée au centre du processus avait pour objectif de diminuer les risques en impliquant plus encore les utilisateurs et experts et devait également inciter à prévoir et effectuer des feedbacks à leur intention plus systématiques que dans un cycle "classique". L'objectif final étant de diminuer les risques, particulièrement ceux liés à un rejet des utilisateurs. Lors du développement de SIADIF, nous avons constaté des comportements qui permettent de valider au moins en partie ces objectifs. Ces constats sont présentés en Tableau 5.1. A partir de ces constats, le modèle est globalement validé. Il reste toutefois des points à évaluer qui font partie de nos perspectives de recherche. Parmi celles-ci, on cherchera à adapter ADESIAD à d'autres types de SIAD en vue d'étendre son domaine d'application.

Objectifs d'ADESIAD	**Constats**	**Validé**	**Non Validé**	**Non évalué**
Adapté au SIAD	Ce modèle a permis de développer SIADIF. SIADIF correspond à un certain type de SIAD (basé sur la connaissance).	X		
Evaluation centrale	Les évaluations et réponses des experts ont amené à de nombreux feedbacks. On remarque lors de la phase d'analyse de SIADIF que la décomposition du problème en sous-problèmes a nécessité de nombreuses réunions mais que cela n'a pas suffi car après validation il y a eu de nouveau un souhait des experts de peaufiner cette décomposition exprimé lors d'une évaluation.	X		
Centré utilisateurs et experts	Les utilisateurs et les experts ont participé de façon active à de nombreuses étapes du cycle ; à la fois dans les étapes de conception, d'évaluation centrale et d'évaluation terminale. La démarche était véritablement participative.	X		
Gestion des risques de rejet	SIADIF est accepté par les utilisateurs. Les évaluations centrales, le maquettage et le prototypage ont aidé à diminuer les risques de rejet, tandis que les évaluations terminales ont permis de valider le système en fin d'itération.	X		

TABLEAU 5.1 – Bilan concernant le modèle d'ADESIAD

5.2.1.2 Bilan de la méthode d'ADESIAD

De la même façon que pour le modèle, des constats ont été établis sur l'utilisation de la méthode issue d'ADESIAD. Ceux-ci, présentés en Tableau 5.2, permettent de valider la globalité des principes de la méthode. Nous retrouvons parmi les objectifs atteints l'établissement des activités pour les acteurs, en particulier experts et utilisateurs, l'intégration des connaissances expertes, la modélisation des utilisateurs, des interactions homme-machine et du système. Les points qui n'ont pas permis d'apporter des conclusions et qui restent à être évalués concernent la modélisation des activités et des experts.

5.2.1.3 Bilan de l'architecture d'ADESIAD

L'architecture a également été évaluée suivant les objectifs de départ à son sujet. Elle devait permettre l'intégration des composants métier. Des travaux issus de la gestion des connaissances nous ont amenés à y intégrer une base des problèmes (base de données relatives aux problèmes visés par le SIAD) pour

Objectifs de la méthode d'ADESIAD	**Constats**	**Validé**	**Non Validé**	**Non évalué**
Etablissement des activités pour les différents intervenants, en particulier pour les utilisateurs	Chaque étape a été détaillée en montrant les activités des utilisateurs lors de chaque étape. Leur rôle était bien défini et mis en pratique lors du développement de SIADIF.	X		
Etablissement des activités pour les différents intervenants, en particulier pour les experts	De la même façon que pour les utilisateurs, le rôle des experts était détaillé pour chaque étape. Lors du développement de SIADIF, ils ont accompli les tâches qui leur été destinées.	X		
Intégration des connaissances expertes	Les patrons proposés visant à la décomposition du problème ont beaucoup aidé les experts pour formaliser leur connaissance à intégrer.	X		
Modélisation des activités	Lors du développement de SIADIF, cette modélisation a été très difficile à mener car les activités des experts et des utilisateurs prévus évoluaient au cours du temps. RFF étant en pleine expansion, les activités et les rôles des employés n'étaient pas stabilisés.			X
Modélisation de l'utilisateur	L'évolution des rôles des personnels de RFF au cours du développement a rendu difficile la modélisation des utilisateurs. Au vu des résultats, on peut tout de même valider la modélisation effectuée car elle montre tout de même les principaux profils.	X		
Modélisation de l'expert	Les méthodes issues de la gestion des connaissances préconisent une modélisation des experts. Dans le cas de SIADIF, il n'a pas été nécessaire de modéliser l'expert avec un modèle spécifique différent de celui de l'utilisateur. Ce point sera donc à évaluer lors de nos perspectives de recherche.			X
Modélisation des interactions homme-machine	Les interactions homme-machine ont été modélisées avec la participation des utilisateurs et des experts et à partir des modèles de ceux-ci. Le chapitre 4 montre que les utilisateurs ont accepté le système avec ses interfaces et que l'évaluation de l'utilisabilité a donné des résultats satisfaisants. Le modèle de base a donc été validé.	X		
Modélisation du système	Le langage UML a été utilisé dans la majorité des étapes (de l'analyse à l'implémentation) pour modéliser le système, les acteurs et les comportements.	X		

TABLEAU 5.2 – Bilan concernant la méthode d'ADESIAD

la gestion des connaissances issues de la décomposition du problème en sous-problèmes (cf. Tableau 5.3). Le premier point n'a pas pu être validé puisque lors de la première itération du développement de SIADIF les composants métier n'ont pas été intégrés à ADESIAD. Le second objectif a pu, quant à lui, être validé.

Objectifs de l'architecture d'ADESIAD	**Constats**	**Validé**	**Non Validé**	**Non évalué**
Intégration des composants métier	L'architecture de SIAD place les composants métier à "l'extérieur" du système de manière à ce que le système soit évolutif et indépendant de ces composants. L'intégration n'a pas pu être évaluée dans SIADIF car les composants métier qui ont tous dû être développés, n'étaient pas prêts à être intégrés, cette intégration fera partie de l'itération suivante dans le cycle.			X
Base de données relatives aux problèmes	A partir de la décomposition du problème en sous-problèmes, une base relative aux problèmes a été implantée. Cette base est utilisée pour faire les liens entre les problèmes et sous-problèmes et entre les problèmes et les composants. Elle permet de proposer les outils associés à un problème. Elle est donc le premier moyen mis en œuvre pour fournir de l'aide à l'utilisateur en utilisant la connaissance des experts formalisée.	X		

TABLEAU 5.3 – Bilan concernant l'architecture d'ADESIAD

5.2.1.4 Bilan global d'ADESIAD

Les bilans se terminent par l'évaluation d'ADESIAD dans son ensemble en fonction des objectifs définis dans le second chapitre. Les objectifs a priori d'ADESIAD étaient de permettre le développement de SIAD en respectant leurs caractéristiques, de mieux prendre en compte les utilisateurs et les experts dans la démarche méthodologique, de placer le décideur au centre du système, de respecter le processus de décision et de faciliter la gestion des connaissances. Selon les constats que nous avons établis durant le développement de SIADIF, cf. Tableau 5.4, nous ne pouvons valider l'intérêt de SIADIF pour la gestion des connaissances. La décomposition du problème en utilisant les patrons constitue une amorce d'ADESIAD visant cet objectif ; cependant, il faut encore le renforcer pour aboutir à une gestion complète des connaissances. Le respect du processus de décision n'a pu être entièrement validé. Il a été validé "*a priori*" par les utilisateurs au niveau de développement de SIADIF mais il ne pourra être validé que lorsque le système sera complet. Enfin, le dernier point concernant la volonté de développer un système centré décideur n'a pas été réellement évalué. L'objectif dans SIADIF était plutôt de centrer le système

sur l'ensemble du processus de décision et donc sur la totalité des utilisateurs ayant un rôle dans ce processus. SIADIF n'est donc pas centré sur le décideur (seul) mais sur toute l'équipe intervenant dans cette décision.

5.2.2 Evaluation des différences d'utilisation de l'outil SIADIF selon le degré d'expertise de l'utilisateur

A partir des évaluations précédemment menées, un des constats était que l'expert sélectionne beaucoup moins d'outils pour mener une étude que les autres utilisateurs. On suppose que la stratégie de l'expert est de sélectionner quelques outils au départ et d'en choisir d'autres pour affiner son étude en fonction des premiers résultats (stratégie progressive) tandis que les autres utilisateurs sélectionnent la totalité des outils en rapport avec l'énoncé de problème. La personne novice dans le domaine (extérieure au domaine) n'a pas su restreindre ses choix : la quasi-totalité des outils ont été sélectionnés quels que soient le problème et le mode de présentation des outils. Cette personne a en fait suivi une stratégie de découverte. Ces différentes stratégies amènent à penser que l'outil SIADIF n'est pas adapté à des personnes hors du domaine, qui ne sont pas familiarisées avec ce type d'étude. Par contre l'utilisateur "chargé d'étude débutant" a apprécié l'outil et a su sélectionner un nombre moins important d'outils.

5.2.3 Apport des composants métier pour aboutir à une prise de décision

Nous avions pris pour hypothèse que la décomposition du problème en sous-problèmes permettrait de fournir les outils (composants métier) appropriés à l'utilisateur et de le guider dans ses choix s'intégrant dans le processus de prise de décision. Dans le cas de SIADIF, les différences en sélections d'outils en fonction du problème et du niveau d'expertise des utilisateurs tendent à valider le fait qu'une panoplie d'outils s'avère plus intéressante qu'un seul outil regroupant l'ensemble des fonctionnalités. De plus, cela permet au système d'être plus évolutif.

Cependant, ce résultat dépend de la granularité des outils. En effet, si la décomposition aboutit à des outils de granularité trop faible, les utilisateurs peuvent avoir des difficultés à imaginer ce qu'ils pourraient apporter réellement ou au contraire les utiliseront systématiquement. Cette granularité dépend de la décomposition. L'implication des experts et des utilisateurs dans cette étape est donc tout à fait indispensable pour obtenir une décomposition aboutissant à des composants utiles pour eux.

De plus, le courant actuel encourage l'utilisation de composants de manière à faciliter la réutilisation. Les utilisateurs ayant participé aux premières évaluations ont apprécié la quantité et la qualité de l'offre en outils avant leur utilisation (*a priori*). Les perspectives à court terme consisteront à évaluer *a posteriori*, dans le cas d'application SIADIF, l'utilité des composants métier dans le processus de décision.

A partir de ces bilans, la partie suivante propose un ensemble de perspectives à nos recherches.

Objectifs d'ADESIAD	**Constats**	**Validé**	**Non Validé**	**Non évalué**
Adapté au développement de SIAD	Le développement de SIADIF et les conclusions qui ont été tirées de cette mise en application permettent de conclure qu'ADESIAD est bien adapté au développement de SIAD. On pourrait toutefois préciser que cela n'a été vérifié que pour les SIAD basé sur la connaissance.	X		
Prise en compte des utilisateurs et des experts	Le modèle et la méthode d'ADESIAD ont montré qu'ils ont bien permis de prendre en compte les utilisateurs et les experts durant la majorité des étapes du développement. Ils interviennent dès les premières étapes d'analyse, de spécifications, lors des évaluations centrales et terminales et éventuellement lors des phases de conception.	X		
Intérêt pour la gestion des connaissances	ADESIAD a permis d'intégrer la connaissance des experts, en particulier lors de la décomposition du problème et grâce à l'utilisation de patrons. De nombreuses notions ont pu être formalisées. Cependant toutes ces connaissances n'ont pas encore été totalement exploitées dans SIADIF ce qui ne permet pas encore de fournir une aide suffisante pour les débutants. A ce stade de développement, les utilisateurs doivent être initiés aux terminologies et aux pratiques issues du monde ferroviaire.			X
Respect du processus de décision	Suite aux évaluations, les utilisateurs ont jugé que SIADIF respecte bien le processus de décision. Ce point est tout de même à vérifier une fois que la majorité des composants seront intégrés et qu'ils permettront une évaluation complète d'un problème menant à une décision.			X
Système centré décideur	L'objectif d'ADESIAD au départ était de permettre le développement d'un SIAD centré sur le décideur. A la suite de l'application, on se rend compte que le SIAD ne peut pas être uniquement centré sur le décideur car celui-ci n'intervient parfois qu'à la fin du processus de décision. Il fallait donc également centrer le système sur les autres acteurs du processus de décision qui seront des utilisateurs. Le système développé avec ADESIAD est donc plutôt centré sur tous les acteurs ayant un rôle dans la prise de décision. ADESIAD n'a pas seulement atteint son objectif, mais l'a en quelque sorte dépassé.		X	

TABLEAU 5.4 – Bilan global concernant l'ADESIAD

5.3 Perspectives de recherche

La première perspective de recherche concerne l'outillage d'ADESIAD. Ensuite, nous présenterons les perspectives liées à l'évolution possible d'ADESIAD pour permettre le développement de SIAD intégrant d'autres caractéristiques supplémentaires. En particulier, les SIAD peuvent jouer un rôle de formation. Ils peuvent être personnalisés pour mieux guider ou mieux renseigner l'utilisateur. Ils peuvent également évoluer vers des systèmes coopératifs.

5.3.1 Outillage d'ADESIAD

Suite à la validation d'ADESIAD, un manque a été ressenti concernant l'outillage favorisant son utilisation. De plus, toute méthode de génie logiciel doit être accompagnée d'un atelier (AGL) pour la supporter. Dans un premier temps, il serait intéressant d'ajouter un éditeur de patrons et de règles à un éditeur UML. Une fois les patrons établis, l'éditeur servirait de base aux concepteurs et experts pour créer les instances des patrons de manière à faciliter la décomposition du problème. Une maquette réalisée à l'aide d'un éditeur d'images a été réalisée (cf. Figure 5.1). Elle montre l'éditeur de diagrammes UML ArgoUml[22] auquel a été ajouté le diagramme de décomposition de problème. Cet éditeur devrait, dans la mesure du possible, permettre une génération automatique de code (ArgoUml le fait déjà sur la base du diagramme de classe). Dans un second temps, l'atelier devrait également guider l'équipe de conception durant tout le cycle en lui fournissant les outils nécessaires, tel que l'éditeur précédemment décrit.

5.3.2 Les SIAD comme moyen d'apprentissage des utilisateurs débutants et novices

Il existe deux types d'apprentissage : l'apprentissage du système pour "personnaliser" l'information à présenter [Anli et al., 2004a], qui sera présenté dans la perspective suivante concernant la personnalisation, et l'apprentissage des utilisateurs de par l'utilisation d'un système.

Le mode de recherche nommé dans SIADIF "par thème", basé sur la décomposition de problème est un premier pas vers l'apprentissage. En effet, ce mode permet à l'utilisateur de choisir les outils relatifs à son étude en se basant sur la répartition des outils faite suivant la décomposition du problème (validée par des experts). Cette aide n'est pas suffisante pour l'apprentissage car, au vu des résultats issus de l'évaluation de SIADIF concernant les modes préférés des utilisateurs, ce mode vise principalement les débutants déjà initiés. Par conséquent, le système pourrait être amélioré pour intégrer la formation des débutants.

Dans un premier temps, les patrons définis en conception, intégrant la connaissance des experts, pourraient être utilisés (puis enrichis) pour initier les débutants au vocabulaire du domaine. Les définitions et les exemples remplis par les experts lors de la définition du problème et complétés au fur et à mesure de l'apparition de nouveaux problèmes pourraient servir de base à la formation des débutants. L'apprentissage serait effectué par utilisation du système.

Dans un second temps, une collaboration entre utilisateurs de différents niveaux d'expérience via le SIAD faciliterait l'apprentissage. Selon ce principe, des travaux ont été effectués via un campus virtuel

[22] Accessible à l'adresse : http ://argouml.tigris.org/

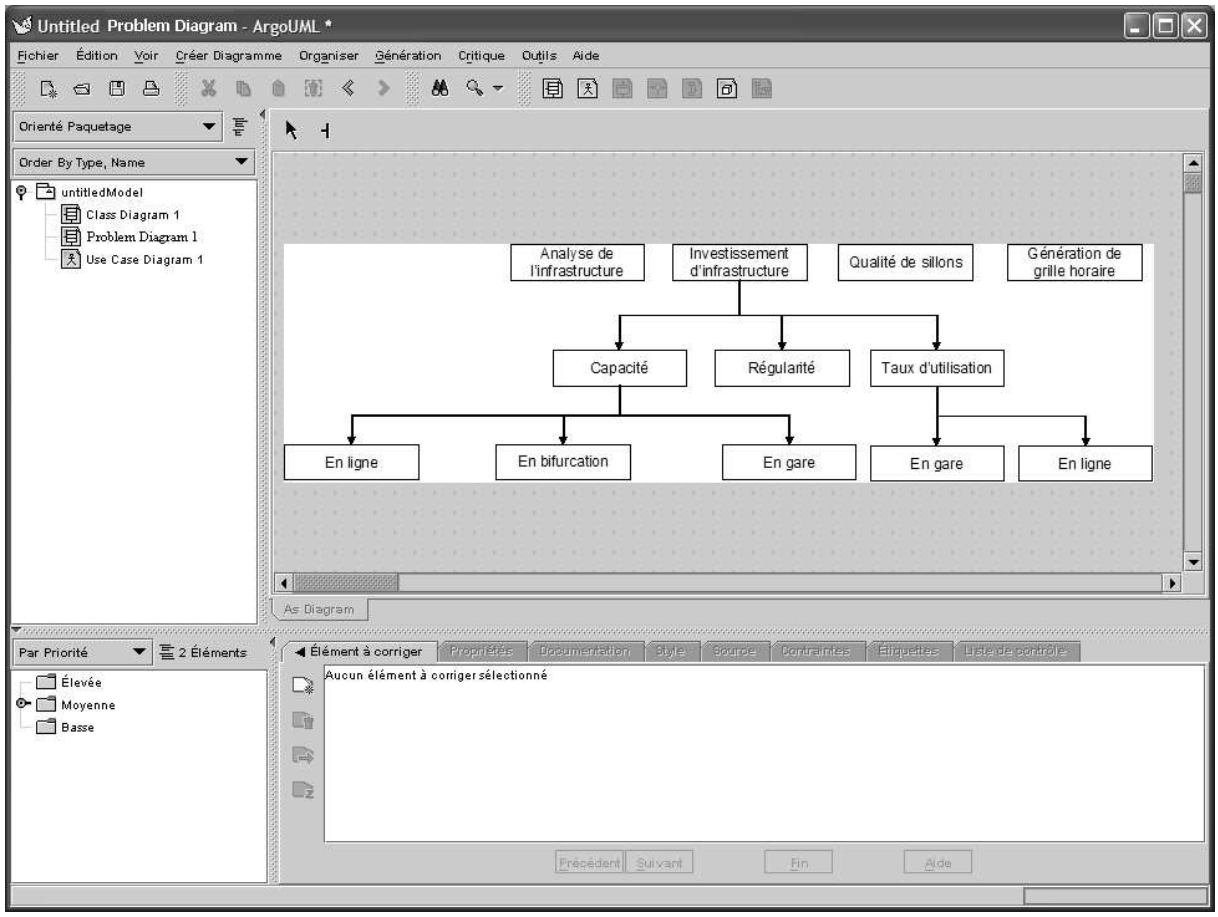


FIGURE 5.1 – Maquettage de l'outil support d'ADESIAD

[Bourguin and Derycke, 2001, Derycke et al., 2003]. Ces études ont donné lieu à des réflexions sur l'efficacité de la collaboration dans le cadre de l'apprentissage [Arnaud, 2003]. Cependant, ces conclusions se basent sur des expériences faites dans les milieux scolaires. Il serait intéressant d'analyser l'apprentissage par coopération/collaboration des utilisateurs dans le milieu professionnel. Des systèmes coopératifs ont déjà été implantés dans des milieux de ce type où le travail nécessite une forte coopération [Pacaux-Lemoine et al., 1996, Debernard et al., 2004]. Ces techniques pourraient être utilisées dans des milieux qui ne demandent pas a priori une forte coopération, c'est-à-dire où les acteurs ne sont pas obligés de coopérer pour réaliser leur tâche. La résolution de problème ne fait, par exemple, pas partie des tâches nécessitant une forte coopération mais la coopération pourrait apporter une aide à la résolution de problème.

Par exemple dans SIADIF, deux chargés d'études de niveaux différents pourraient collaborer pour réaliser une tâche commune de résolution de problème d'investissement. L'expert, en justifiant ses choix, aurait une influence sur l'apprentissage du novice. Pour mettre en place une telle structure, il faudrait intégrer la notion de collaboration dans le cycle d'ADESIAD. Pour cela, le modèle des tâches interactives devrait intégrer non seulement les interactions homme-machine comme cela a déjà été fait mais aussi homme-machine-homme ou hommes-machines [Karsenty, 1993]. La Figure 5.2 présente en fond

grisé les modifications à apporter aux étapes d'analyse et de spécification pour intégrer ces nouvelles interactions.

Enfin, une autre solution consisterait à intégrer au SIAD des outils de formation (issus de l'entreprise) sous la forme de composants métier.

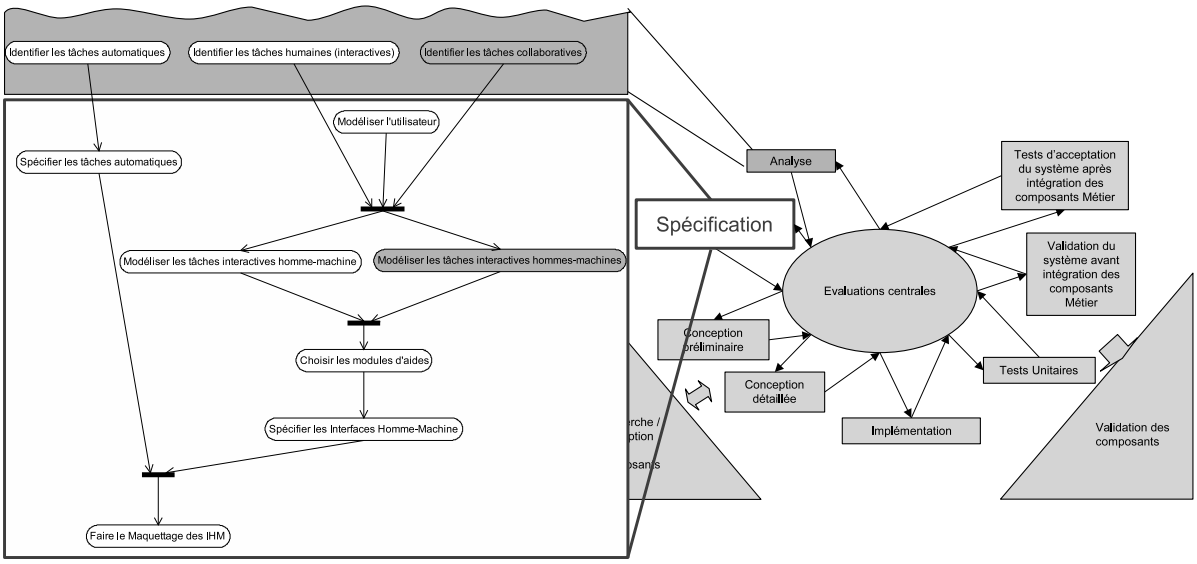


FIGURE 5.2 – Modification possible de la méthode d'ADESIAD pour intégrer les tâches collaboratives

5.3.3 Personnalisation de l'information dans les SIAD

Palma-dos-Reis et ses collègues [Palma-dos Reis and Zahedi, 1999] placent dans les perspectives d'évolution des SIAD la personnalisation. Par personnalisation, nous n'entendons pas simplement personnalisation de l'interface homme-machine mais personnalisation des informations. Elle est définie par [Anli et al., 2004a] par le filtrage d'un ensemble de données en fonction du profil utilisateur, la recommandation d'information, la classification des informations selon leur pertinence pour un utilisateur donné. Elle peut consister à adapter la présentation des informations au support d'interaction ou à proposer des informations connexes.

Nous pensons qu'une manière d'intégrer la personnalisation dans les SIAD serait de modifier ADESIAD en vue de l'utilisation d'un composant *utilisateur*, tel qu'il a été défini dans les travaux de Bouchet et ses collègues [Bouchet, 2003, Bouchet and Nigay, 2004]. Ces travaux visent une approche à composants pour l'interaction multimodale. Leur plateforme fournit divers types de composants :

- les composants élémentaires (de base), dont :
  - le composant *utilisateur* qui représente les caractéristiques de l'utilisateur. Ces caractéristiques correspondent à différents types d'informations qui sont soit statiques (nom, prénom), soit dynamiques (préférences).
  - le composant *dispositif* représente la réalité physique d'entrée ou de sortie du système. Ce composant est en contact direct avec l'utilisateur.

– les composants de composition sont présents pour assurer la multimodalité de l'application.
– le composant d'assignation a pour rôle de faire l'interface entre les composants et l'application.

Le composant *utilisateur*, défini par [Bouchet, 2003], serait ajouté à l'architecture. Ceci permettrait d'associer à chaque utilisateur les composants préférés ou, en tout cas, les plus adaptés. Il serait intéressant d'intégrer ce mode de personnalisation à notre approche de manière à ce qu'il soit systématiquement appliqué lors de la conception de systèmes interactifs. Le composant *utilisateur* serait issu du modèle de l'utilisateur présent dans la phase de spécification du modèle d'ADESIAD, cf. Figure 5.3.

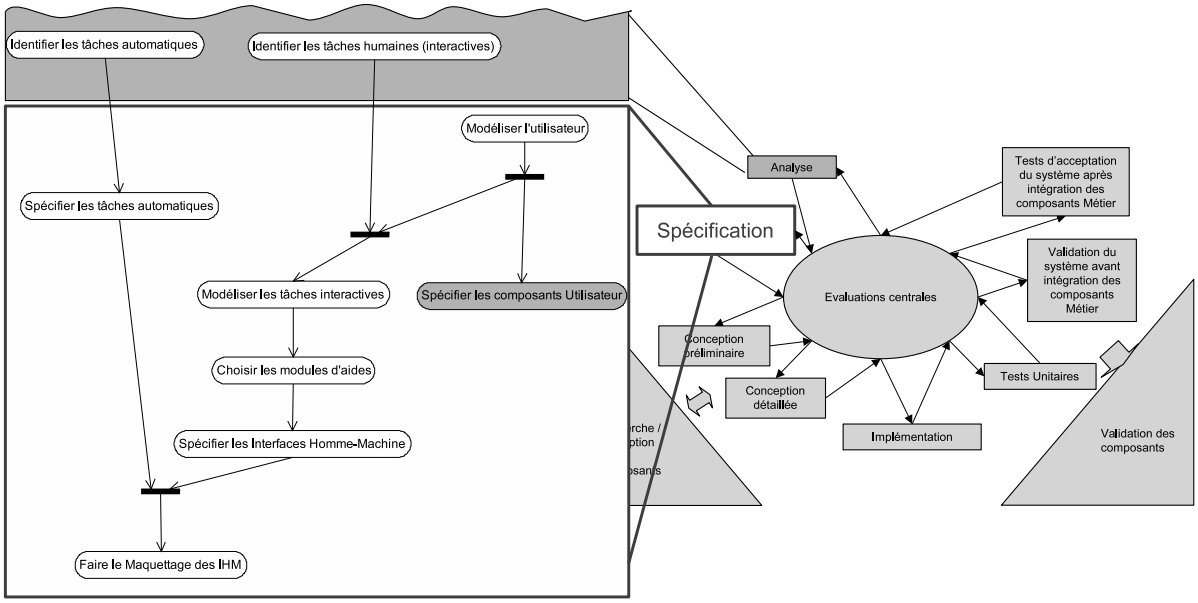


FIGURE 5.3 – Modification possible de la méthode d'ADESIAD pour intégrer les composants utilisateur

D'autres approches basées sur des Systèmes Multi-Agents (SMA) proposent aussi de personnaliser l'information. Par exemple, [Petit-Rozé, 2003] a proposé une organisation multi-agents pour la conception de Systèmes d'Information Personnalisée (SIP). Cette organisation intègre quatre types de modèles :

– d'*assistance*, responsable de la gestion des interactions avec l'utilisateur ;
– de *coordination*, responsable de la mise en adéquation entre la requête de l'utilisateur et les solutions à lui apporter ;
– de *recherche de données*, ayant pour mission de récupérer l'ensemble des solutions satisfaisant la requête et
– de gestion de profils, responsable de la maintenance des profils utilisateurs.

Cette approche pourrait être adaptée et intégrée à ADESIAD pour la personnalisation de l'information, particulièrement celle qui concerne les SIAD. Le modèle de recherche de données serait remplacé par un modèle de recherche de composants métier qui sélectionnerait les composants métier les plus en rapport avec le problème. Le modèle coordination sélectionnerait, parmi les composants sélectionnés par le modèle de recherche, les composants métier les plus appropriés en fonction de la réponse de l'agent gestion de profil, cf. Figure 5.4. L'architecture d'ADESIAD pourrait donc évoluer vers une architecture à base d'agents et de composants, cf. Figure 5.5. Cependant, cette organisation n'est intéressante qu'à

partir du moment où les composants métier sont suffisamment nombreux (ce qui ne sera, par exemple, pas le cas avant plusieurs années d'exploitation de SIADIF et d'intégration progressive de composants dans celui-ci).

Cette personnalisation peut être dangereuse dans un SIAD si elle prive l'utilisateur d'informations. Pour une utilisation de la personnalisation dans les SIAD, il faut vérifier à ne pas priver l'utilisateur d'information qui aurait pu sembler inadaptée car l'utilisateur (décideur) doit rester maître de ses choix.

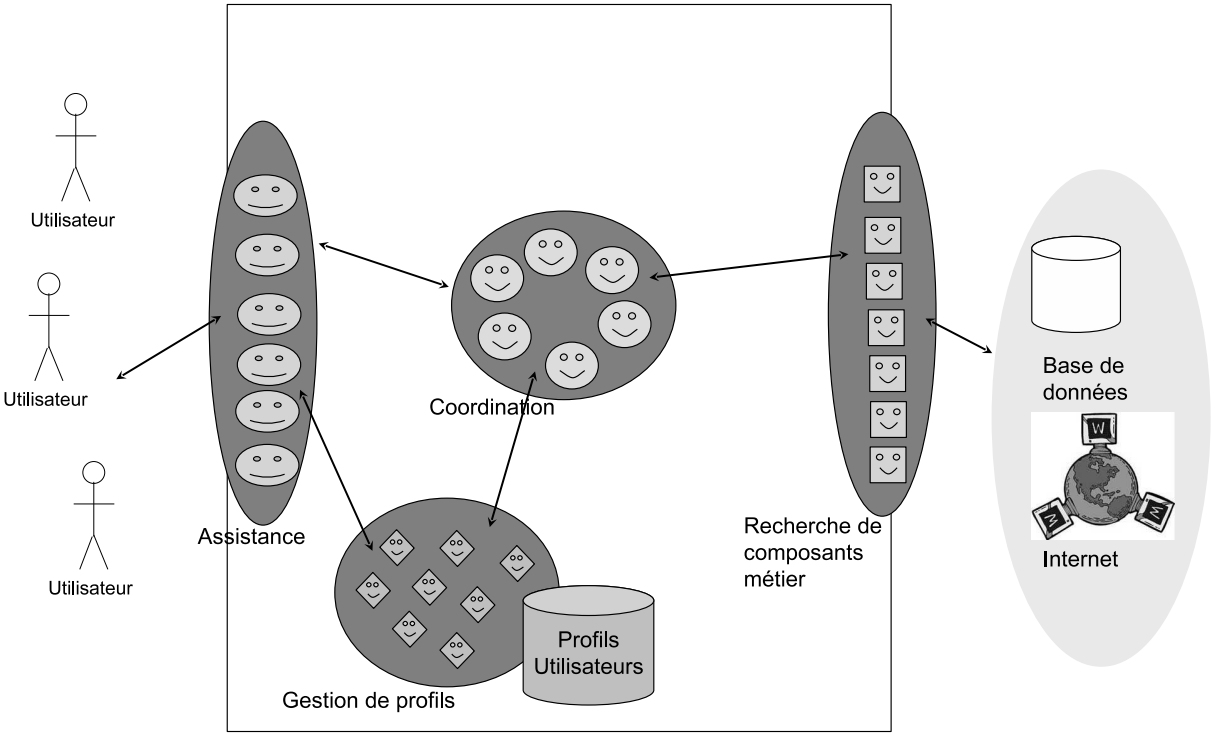


FIGURE 5.4 – Adaptation de MAPIS [Petit-Rozé, 2003] au SIAD

Notons que cette approche peut-être enrichie par des principes d'apprentissage pour une meilleure personnalisation de l'information [Anli et al., 2004a, Anli et al., 2004b]. L'apprentissage est alors utilisé pour enrichir le modèle utilisateur. Ainsi, un questionnaire peut être utile pour commencer à personnaliser l'information (mais il ne peut être long et doit contenir des questions correspondant aux informations essentielles qui ne peuvent pas être apprises facilement). Ensuite, au fur et à mesure des interactions homme-machine, l'apprentissage peut permettre de connaître les préférences de l'utilisateur et d'affiner son profil. Par exemple, dans le cas de SIADIF, les préférences des utilisateurs concernant les modes de sélection seraient vite apprises. De plus, à partir de l'intitulé du problème, il serait possible d'apprendre les préférences de l'utilisateur par rapport à ce type de problème.

5.3.4 Des SIAD aux GSIAD

Plusieurs acteurs peuvent intervenir dans la prise de décision concernant un problème. C'est à partir de ce principe que le SIAD en groupe (GSIAD) est né. Le GSIAD fait partie du domaine du Travail

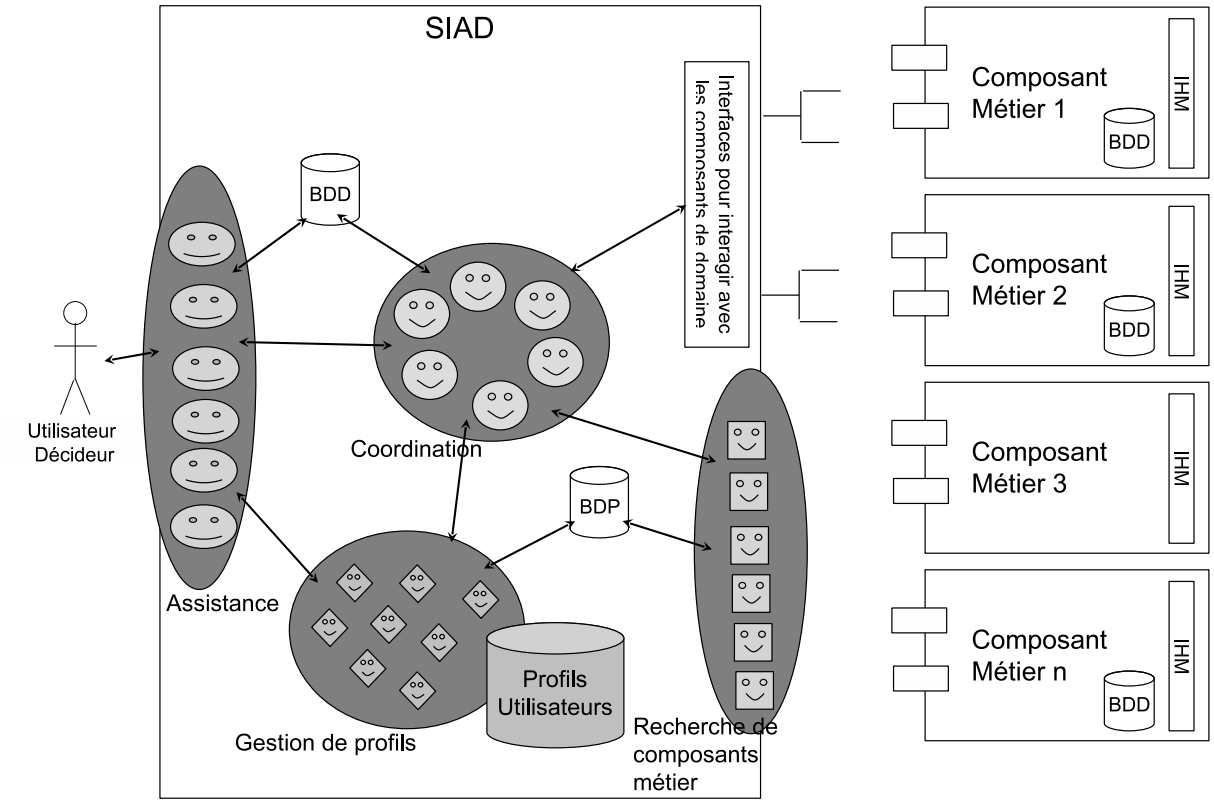


FIGURE 5.5 – Adaptation possible d'ADESIAD pour intégrer la personnalisation basée sur les SMA

Coopératif Assisté par Ordinateur (TCAO)[23]. Le principe est de rassembler les personnes d'un groupe via un espace de travail commun virtuel. Pour transformer un SIAD en un GSIAD, il faut intégrer au SIAD des composants permettant la coopération. Cela induit une modification du modèle de développement d'ADESIAD. En effet, il ne faut plus prendre en compte le seul modèle utilisateur mais le modèle de la coopération. Pour que le système soit accepté, il faut que le système leur permette de travailler en coopération.

Certains travaux, axés sur les collecticiels, ont pour objectif de faciliter la composition logicielle [Payet, 2003]. La composition est vue comme la donnée d'un plan de circulation des messages entre les composants. Le résultat d'une composition est un réseau de composants qui décrit le plan de circulation des messages d'un composant émetteur à un composant récepteur. L'enrichissement de messages est utilisé comme support à la composabilité[24], cf. Figure 5.6a. L'enrichissement est établi par l'ensemble des acteurs intervenants dans l'environnement collaboratif. Chaque participant a la possibilité d'intervenir dans le processus de composition et d'influer sur l'adaptation globale de l'application. Il peut également être utilisé comme support à l'interopérabilité[25], cf. Figure 5.6b. Son objectif est alors d'aider au processus d'adaptation du message pour qu'il soit compatible avec chacun des destinataires qui lui ont été

[23]Le terme anglophone correspondant au TCAO est CSCW : Computer-Supported Collaborative Work.

[24]Composabilité : pouvoir d'associer et de confi gurer un ensemble de composants en vu de la réalisation d'un traitement donné.

[25]Intéropérabilité : aptitude d'un ensemble de composants à se comprendre et à travailler ensemble.

désignés.

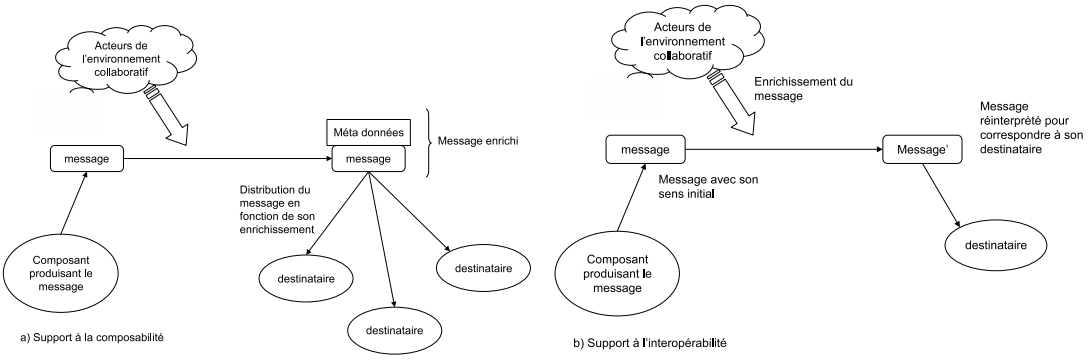


FIGURE 5.6 – Enrichissement de message d'après [Payet, 2003]

Ces travaux pourraient être utiles pour faciliter la composition dans les SIAD, notamment dans le cas où la collaboration y serait intégrée. L'architecture d'ADESIAD s'est voulue globale pour ne pas se restreindre à un type de composant. De cette manière en intégrant les principes de ces travaux, il est possible d'intégrer les composants spécifiques à l'enrichissement de message tels que les composants *producteurs de message*, *récepteurs de message* et *présents* dans le système. Ceux-ci jouent un rôle dans le "ciblage collectif", cf. Figure 5.7. Ce ciblage permet à chaque entité du système logiciel de prendre part collectivement à la gestion de la composition.

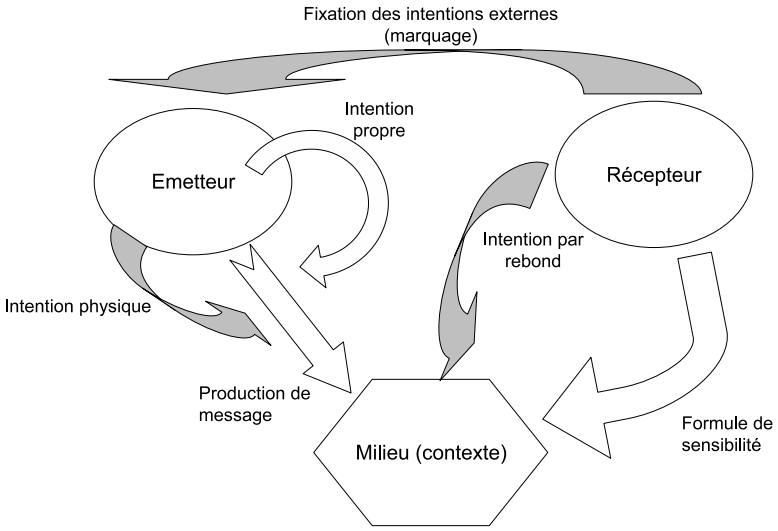


FIGURE 5.7 – Champs d'action émetteur/récepteur/autres composants dans le ciblage collectif [Payet, 2003]

5.3.5 Les principes de la co-évolution adaptés aux systèmes non coopératifs

Un système coopératif doit pouvoir s'adapter aux besoins émergeants, à la transformation de besoins ayant servi à l'analyse initiale, ainsi qu'aux évolutions des contextes d'usages et des technologies uti-

lisées [Benali et al., 2002]. Le terme co-évolution a été choisi pour traduire le fait que les systèmes coopératifs doivent être continûment évolutifs, sans l'être toutefois de façon autonome ou auto-adaptative, car ils doivent rendre compte, de manière consciente, des évolutions des besoins, des attitudes et des compétences des usagers, individuellement ou collectivement [Benali et al., 2002].

La notion de co-évolution peut être adaptée aux systèmes non coopératifs (nous considérons que nous nous trouvons en présence de tels systèmes). En effet, on peut supposer que quel que soit le système (coopératif ou non) les besoins évoluent. La Figure 5.8a illustrant la co-évolution pourrait devenir plus générale (cf. Figure 5.8b), adaptée à des systèmes non coopératifs. Dans la définition de la co-évolution, les nouveaux besoins sont engendrés par la coopération. D'autres contextes de travail peuvent eux aussi induire une évolution.

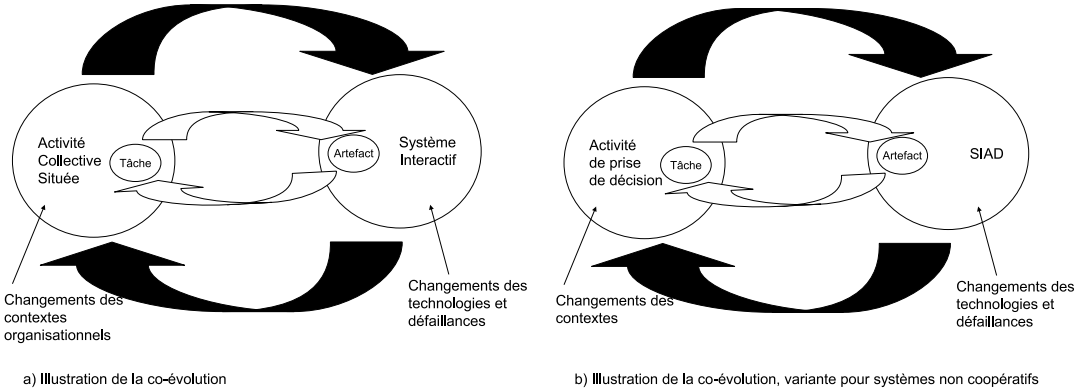


FIGURE 5.8 – Application de la co-évolution à des systèmes non coopératifs de type SIAD

Un des moyens utilisés pour répondre à la co-évolution est la malléabilité ; elle intègre la flexibilité et l'adaptabilité d'un système [Bourguin, 2000, Payet, 2003]. Elle peut être de trois niveaux [Morch, 1997] :

- le paramétrage, il est réalisable par une sélection à établir à travers un ensemble d'options de configurations prédéfinies.
- l'intégration, elle consiste à lier des composants prédéfinis à l'intérieur d'une application ou entre des applications disjointes.
- l'extension, elle correspond à une modification au niveau de l'application, par exemple en ajoutant du code à son programme.

Selon nous, ces notions sont identiques pour des systèmes non coopératifs. Les travaux de Bourguin sont basées sur des technologies par composants pour permettre cette malléabilité [Bourguin, 2000]. Ils peuvent servir de base à nos perspectives pour l'évolution des SIAD.

Dans ces travaux, ce sont les utilisateurs qui font évoluer le système. Dans le cas de systèmes coopératifs, la problématique consiste à ce qu'ils puissent tout de même garder une base commune de manière à garder leurs moyens de collaboration. Dans les SIAD, que l'on suppose à la base non collaboratifs, le problème persiste au niveau de l'entreprise. En effet, si un utilisateur fait évoluer un composant de telle sorte que l'apparence des résultats ne change pas mais qu'ils ne signifient plus tout à fait la même chose, cela peut aboutir à des problèmes de collaboration dans l'entreprise.

Par exemple, si un chargé d'étude utilise souvent trois composants pour résoudre un type de problème,

le décideur connaît les composants et se fait une idée des répercussions et des décisions à prendre. Si par la suite, le chargé d'étude fait évoluer un des trois composants, le décideur sera induit en erreur par cette évolution. L'évolution qui intervient dans un système non collaboratif mais servant de base à des collaborations, suivra la même problématique que la co-évolution.

5.3.6 Technologies proches des composants : fédération d'outils et services

La prise en compte de technologies dérivées des composants telles que les fédérations d'outils ou les services web fait partie de nos perspectives. Nous souhaiterions à court terme étudier ce que peuvent apporter ces technologies pour le développement de SIAD.

Nous avons vu dans le premier chapitre qu'une certaine cohérence commence à émerger dans le domaine des composants sur la notion de composants mais des différences au niveau des définitions perdurent : aucun consensus n'est encore établi. De la même façon que pour les composants électroniques, lorsqu'il y a plusieurs standards en cours, les fabricants doivent choisir parmi ceux-ci. Ce choix présente un risque étant donné que le standard choisi ne persistera peut-être pas. D'autres proposent des adaptateurs ou se conforment à plusieurs standards. La problématique consiste alors à regrouper des composants issus de divers modèles ayant des standards différents ; c'est dans ce cadre que les travaux sur les fédérations d'outils ont émergé.

5.3.6.1 Des composants aux fédérations d'outils

Une fédération d'outils correspond à une vision d'assemblage d'outils, de constituants dont l'objectif est de construire des environnements de progiciels existants qui doivent coopérer [Verjus, 2001, Villalobos, 2003, Le, 2004]. Les fédérations font partie des systèmes à base de composants. Elles intègrent en supplément l'indépendance des participants (outils) et l'autorité commune. Celle-ci a pour rôle de contrôler la synchronisation des participants avec l'Univers Commun (UC). Cet univers commun est partagé entre l'univers commun de l'application (UCA) et l'univers commun du contrôle (UCC). Les outils sont hétérogènes au sens où ce sont des composants suivant divers standards. Pour qu'ils puissent participer à une fédération, il faut qu'ils soient adaptés ; il existe alors deux types d'adaptations :

- L'adaptation conceptuelle a pour but de résoudre ces incohérences par l'association, à chaque participant de la fédération d'un représentant.
- L'adaptation technique doit permettre la communication entre les outils et l'univers commun de l'application en utilisant une façade.

L'architecture permettant ces deux types d'adaptation est présentée en Figure 5.9.

Ces travaux pourraient être utilisés dans le développement de SIAD si on considère que le SIAD est l'univers commun et que les outils correspondent aux composants métier que nous avons définis dans ADESIAD. L'intérêt de cette approche est qu'elle permet d'intégrer des composants quel que soit le modèle par l'utilisation des façades et des adaptations. Les composants métier subiraient alors moins de contraintes lors du développement. En conclusion, les composants métier intégrables seraient plus nombreux et la réutilisation serait plus efficace.

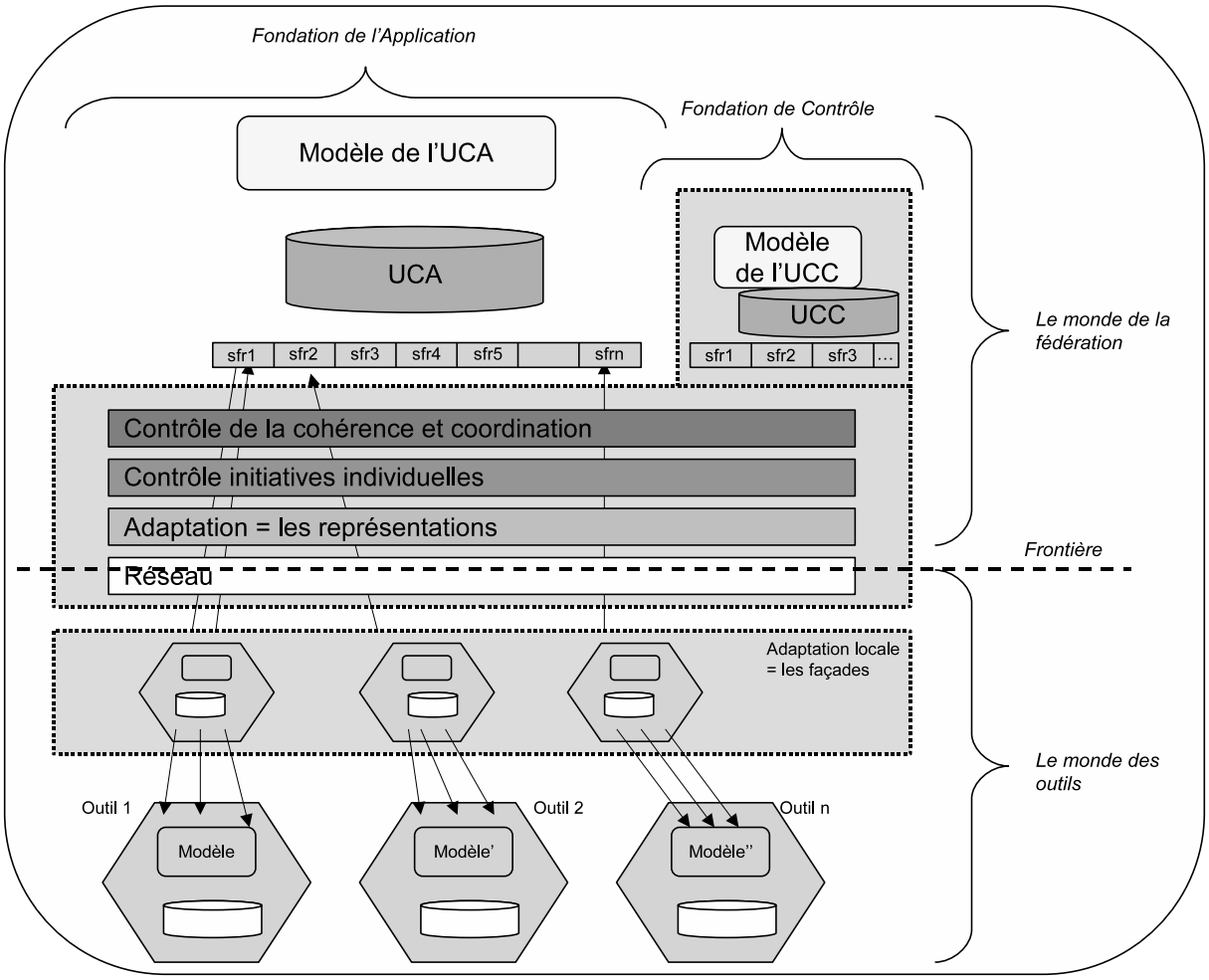


FIGURE 5.9 – Architecture pour fédération d'outils [Le et al., 2003]

5.3.6.2 Des composants aux services

Une notion proche de la notion de composant est la notion de services. [Cervantes, 2004] donne la définition suivante : un service est une fonctionnalité réutilisable qui est décrite de façon contractuelle dans un descripteur de services. Un tel descripteur inclut des informations syntaxiques, c'est-à-dire une interface de service, mais peut aussi contenir des informations qui décrivent le comportement ou qui caractérisent le service. Un fournisseur de services fournit des objets de services qui implémentent l'interface du service permettant à un client d'utiliser sa fonctionnalité. Une caractéristique fondamentale de l'approche à services est que l'assemblage d'une application à base de services est réalisé à partir de descripteur de services. La découverte et la liaison avec des fournisseurs de services n'ont lieu que de façon tardive, c'est-à-dire avant ou pendant l'exécution de l'application.

C'est cette liaison tardive, qui se fait pendant l'exécution, qui nous semble intéressante pour être intégrée aux SIAD. Nous avons choisi dans nos travaux d'utiliser des composants. Les composants métier dans ADESIAD sont principalement des outils. Nous avons choisi d'intégrer ces composants métier dans le cycle d'ADESIAD car ils font partie des aides aux utilisateurs pour résoudre leurs problèmes. Si on

sort ces outils du cycle, ils deviennent des services. Dans ce cas, le SIAD ne peut être évalué avec ces outils. Les outils supports à la formation sont quand à eux des "bonus" par rapport à la prise de décision. Ils pourraient donc être intégrés sous la forme de service en fonction des besoins des utilisateurs.

Une alternative serait de baser ADESIAD sur un modèle à composants orienté services proposé par [Cervantes, 2004]. Ce modèle permet la disponibilité dynamique dans un modèle à composants (cf. Figure 5.10). Dans ce modèle, des concepts sont équivalents à ceux d'un composant "classique" (propriétés de configuration, d'interface de contrôle, les propriétés et dépendances de déploiement). Parmi les interfaces de services, on retrouve :

- La cardinalité qui exprime le caractère optionnel ou obligatoire de la création d'une connexion avec un fournisseur de l'interface de service requise ainsi que la multiplicité.
- La politique qui définit si une fois que les connexions ont été créées, elles peuvent être changées (connexion dynamique) ou non (connexion statique).
- Le filtre qui permet de contraindre la réalisation de connexions vers un sous-ensemble de fournisseurs de services ou bien vers un fournisseur de service particulier, pour réduire le problème de l'imprévisibilité.

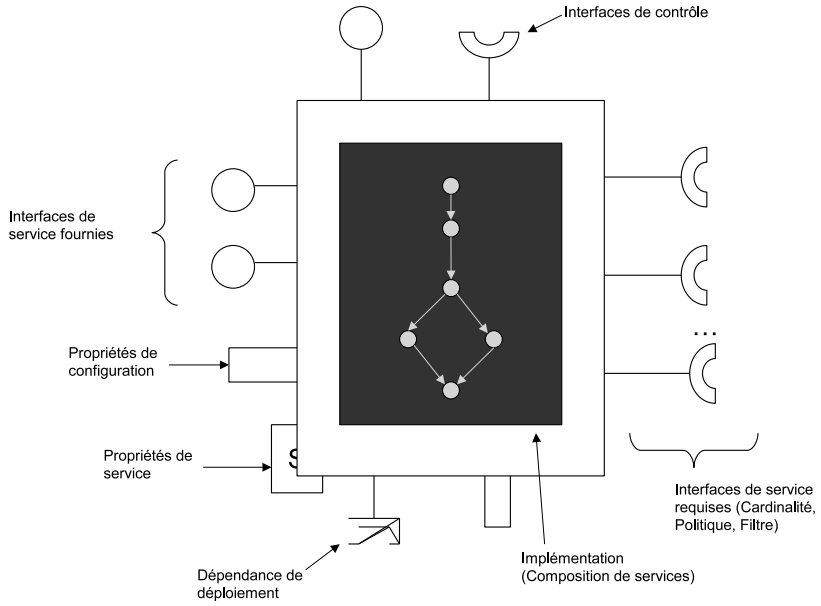


FIGURE 5.10 – Classe de composant de service selon [Cervantes, 2004]

Le composant à service diffère d'un composant par certains aspects [Cervantes, 2004] :

- Les interfaces fonctionnelles fournies correspondent à des interfaces de services fournies.
- Des propriétés de service sont associées à l'implémentation du composant.
- L'implémentation du composant à services peut réaliser une composition de services. Dans ce cas, le composant requiert des interfaces de service ; ces interfaces de service sont caractérisées par un certain nombre d'informations destinées à configurer la logique de l'adaptation.

Les travaux présentés dans cette section peuvent être utilisés pour intégrer les composants de type métier en phase d'exécution qui seraient alors des composants orientés service, basé sur la définition de

Cervantes, cf. Figure 5.10. Cela aurait pour incidence d'extraire les composants métier devenus composants orientés services, de la validation du système, cf. Figure 5.11. En effet, selon ce principe, puisque les composants sont intégrés après la recette cela implique que le système ne peut être évalué avec. Ceci peut être gênant dans le cas des SIAD car on ne peut plus s'assurer que les outils apportent une aide dans le processus de décision, ce qui était le but de l'évaluation du système avec les composants métier dans ADESIAD. Cependant, on peut supposer que les utilisateurs intégreront à l'exécution des outils qui les supportent dans cette tâche. Pour remédier à ce problème, il faudrait peut-être distinguer les composants métier jouant un rôle dans le processus de décision et les intégrer aux évaluations et garder les autres composants (par exemple les composants de formation) comme composants orientés services. Ce choix doit être fait en fonction du cas d'application, en accord avec les experts et utilisateurs.

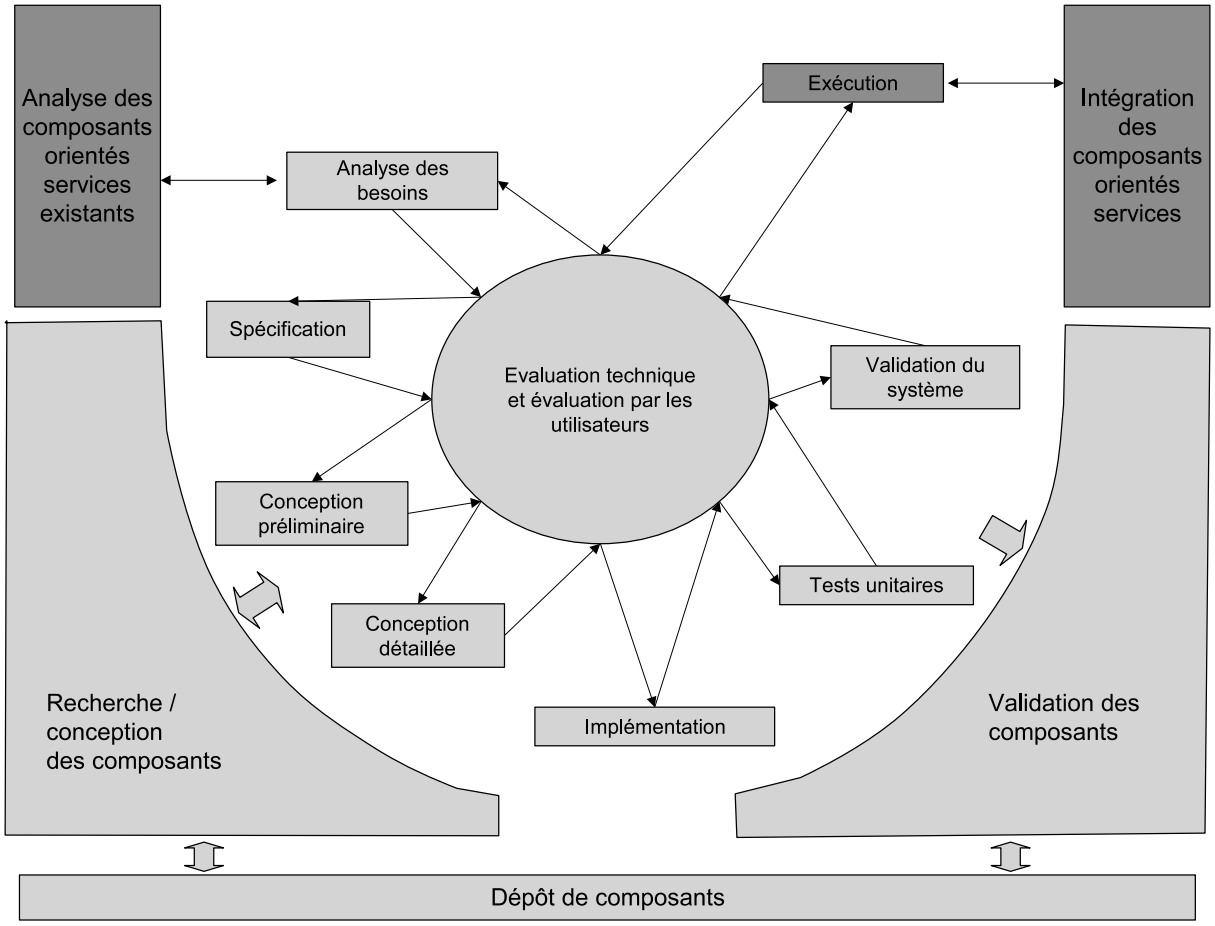


FIGURE 5.11 – ADESIAD modifiée pour l'intégration de composants orientés services

5.3.7 Applications d'ADESIAD

ADESIAD a été appliqué pour le développement de SIADIF, cela n'est pas suffisant pour la valider complètement. Cette section présente les perspectives d'application d'ADESIAD. Avant toute chose, l'application d'ADESIAD à SIADIF doit être terminée ; il reste de nombreuses itérations à effectuer. De

plus, l'aide à la prise de décision ne pourra être réellement évaluée que lorsque les composants métier seront majoritairement intégrés.

Ensuite, il serait intéressant pour évaluer chacune des particularités d'ADESIAD de l'appliquer à la fois complètement et partiellement de manière à s'assurer de l'utilité de chacune des propositions. Cela permettrait de valider chacun des apports de la méthode.

Pour finir, l'application d'ADESIAD à d'autres types de SIAD moins "dépendants des connaissances" (selon le modèle de Rasmussen), tels que les systèmes de régulation, de supervision permettrait d'étendre son action. L'application d'ADESIAD à des systèmes qui ne sont pas centrés sur une prise de décision mais tout de même basés sur la connaissance permettrait encore d'étendre le champ d'action de cette méthode.

5.4 Conclusion

La première partie de ce chapitre a présenté une évaluation globale de l'approche ADESIAD. Sur la base du développement de SIADIF effectué dans le chapitre précédent, plusieurs bilans sur le modèle, la méthode, l'architecture et l'approche complète ont enrichi cette évaluation. Ces bilans ont permis de valider globalement les principes intégrés dans ADESIAD. Il reste cependant de nombreuses perspectives qui ont été exposées en seconde partie. On compte parmi elles l'application complète d'ADESIAD permettant de valider le principe de décomposition pour l'aide à la décision. L'approche proposée sera complètement utilisable lorsqu'un atelier de génie logiciel (AGL) l'accompagnera. D'autres perspectives visent à intégrer l'apprentissage, la personnalisation, la coopération au modèle. D'autres technologies proches de celles des composants ont été analysées pour montrer leur intérêt à être intégrées au développement de SIAD.

Conclusion générale

Le thème principal abordé dans ce mémoire concernait les Systèmes Interactifs d'Aide à la Décision (SIAD). Le premier chapitre présentait donc ces systèmes, en intégrant les définitions issues du domaine et la notre. A partir de l'évolution des définitions, nous avons pu remarquer que les interactions homme-machine, qui n'y étaient pas du tout présentes au début (on parlait plutôt de SAD), ont pris une ampleur telle que l'on peut désormais considérer les SIAD comme des systèmes qui devraient être systématiquement centrés sur l'utilisateur. Les SIAD qui nous intéressent sont ciblés sur la connaissance ; ce qui nous a amené à présenter les systèmes de gestion de connaissance ou systèmes de connaissance. L'aspect de réutilisation à la fois des connaissances et des artefacts nécessaires au développement, nous a conduit à étudier les systèmes à base de composants et à déduire que les SIAD devaient être également basés sur les composants pour intégrer ces notions. Une présentation succincte des systèmes d'aide dans le milieu ferroviaire, à la fin du premier chapitre, nous a permis de montrer qu'il n'y avait pas réellement de système efficace pour les prises de décision relatives aux investissements dans les infrastructures ferroviaires ; le besoin d'un SIAD réalisant cet objectif est donc ressorti de cette étude.

Le second chapitre était, quant à lui, centré sur les approches de développement de systèmes correspondant aux domaines énoncés dans le premier chapitre. Les approches (modèles, méthodes et architectures) classiques du génie logiciel, qui sont souvent à la base d'autres approches, ont été présentées en premier lieu. Ensuite, nous avons montré que ces modèles ont évolué pour s'adapter aux systèmes particuliers dans le domaine des Interaction Homme-Machine (IHM), de la gestion de la connaissance et de la réutilisation. Nous avons également présenté les approches de développement spécifiques aux SIAD. L'étude des approches présentées dans les divers domaines, nous a permis de conclure que les approches spécifiques aux SIAD ne prennent pas réellement en compte les spécificités des autres domaines dans lesquels, selon nous, les SIAD s'intègrent ; il nous fallait alors proposer une approche que nous permette de développer un SIAD en intégrant les principes des domaines auxquels ils se rattachent.

Notre contribution a visé à combler cette lacune en approche de développement pour SIAD. Le troisième chapitre avait alors pour but de présenter une nouvelle approche nommée Approche de Développement de Systèmes Interactifs d'Aide à la Décision (ADESIAD). Celle-ci regroupe des principes d'autres approches de manière à intégrer les notions essentielles au développement de SIAD. ADESIAD propose un modèle itératif, incrémental et parallèle. Ce modèle intègre la notion d'évaluation centrale et terminale, ce qui est rare dans les approches provenant d'autres domaines. Dans ses étapes, les participations à la fois d'utilisateurs et d'experts sont centrales. Ce modèle est la base de la méthode présentant les étapes de manière plus détaillée. Dans cette méthode, le principe de décomposition de problème à l'aide de patron est la base du développement pour formaliser la connaissance liée au problème. Cette décomposition permet aussi de faire ressortir le besoin en composants métier ; ce type de composants étant alors relié aux étapes d'analyse et de spécification, contrairement aux composants de conception

qui interviennent en phase de conception préliminaire et aux composants logiciels qui n'interviennent qu'à l'étape de conception détaillée. Ces composants sont soit déjà existants, soit devront être développés en parallèle au SIAD. Enfin, l'architecture proposée dans l'approche et utilisée dans la méthode montre bien la séparation des composants métier du système. En effet, le système est conçu pour être indépendant des composants métier, ce qui permet de rendre le système plus évolutif. Ils peuvent être ajoutés et supprimés sans que cela n'ait de répercussion sur l'architecture du SIAD.

Le développement d'un Système Interactif d'Aide à la Décision relative aux Investissement dans les infrastructures Ferroviaire (SIADIF) fut supporté par ADESIAD. Cette mise en application a permis de montrer que la participation des experts et des utilisateurs était fondamentale. La décomposition du problème de décision en sous-problèmes n'était pas évidente puisqu'il a fallu trois décompositions avant que les experts et utilisateurs n'en valident une. L'utilisation de patrons a apporté une aide précieuse aux experts pour expliciter leurs connaissances. Par contre, quelle que soit la décomposition, les experts et utilisateurs avaient une idée très précise des outils dont ils avaient besoin pour résoudre leur problème. L'évaluation du système à la fin de la première itération a montré l'utilité de la décomposition du problème pour apporter une aide aux utilisateurs considérés comme novice. Les utilisateurs ont accepté le système, ils attendent maintenant que les composants métier soient prêts à être intégrés pour le valider complètement.

Pour finir, le cinquième chapitre a présenté d'abord les conclusions que nous avons pu tirer sur ADESIAD au vu du développement de SIADIF et des évaluations. En effet, les évaluations ayant eu lieu à la fin de la première itération, accompagnés des constats faits au long du développement ont permis de valider les principes intégrés à ADESIAD. La décomposition du problème en sous-problèmes a été acceptée, validée et a pu montrer son intérêt au cours du développement du SIAD. Les évaluations centrales ont permis la remise en question de certains choix ; cela a légèrement ralenti le développement au cours du premier cycle mais a assuré que le système soit accepté en phase terminale. La séparation du système et des composants a permis de valider le système alors que les composants métier n'étaient pas encore prêts à être intégrés. Ceux-ci peuvent être ajoutés ultérieurement. Il est vrai que l'évaluation du système avec les composants est en attente mais le développement séparé et itératif permet de continuer le développement du système selon l'itération suivante qui n'est pas "bloquée" par le manque de composants métier. Ensuite, les perspectives de recherche de nos travaux concernant les améliorations et les évolutions possibles d'ADESIAD ont été présentées. Nous avons commencé ces perspectives par l'outillage nécessaire de l'approche maintenant qu'elle a été validée. Les perspectives de recherche sur l'amélioration d'ADESIAD pour intégrer de la personnalisation, de l'apprentissage et de la collaboration aux SIAD ont été présentées. Des technologies proches de celles des composants ont été présentées de manière à montrer leur intérêt à être intégrées dans le développement de SIAD. D'autres perspectives concernant les applications d'ADESIAD à d'autres types de SIAD ont terminé ce mémoire.

Pour finir rappelons que, selon [Coriat and Weinstein, 1995] et [David and Foray, 1995], dans une hypothèse évolutionniste nommée "la dépendance du sentier", l'innovation est un processus de "création technologique endogène et cumulatif", c'est-à-dire que c'est la nature même du patrimoine de connaissances accumulé dans une organisation qui prédétermine le sentier d'évolution des connaissances. Les

travaux de ce mémoire avaient pour but d'innover et ont suivi cette définition. En partant des connaissances de l'organisation et en apportant des connaissances d'autres organisations, nous avons tenté d'améliorer les connaissances passées. Ces connaissances résultantes ont été utilisées pour le développement d'un système pour RFF dont les premiers utilisateurs ont été globalement satisfaits et qui attendent la suite pour améliorer eux aussi leurs connaissances.

Bibliographie générale

[Abed, 2001] Abed, M. (2001). *Méthodes et Modèles formels et semi-formels de conception et évaluation des systèmes homme-machine*. Mémoire d'HDR, Université de Valenciennes et du Hainaut-Cambrésis.

[Abed et al., 1991] Abed, M., Bernard, J. M., and Angué, J. C. (1991). Task analysis and modelization by using SADT and Petri networks. In *Proceedings Tenth European Annual Conference on Human Decision Making and Manual Control*. Liège.

[Abed et al., 1995] Abed, M., Ezzedine, H., and Angué, J. C. (1995). Méthodologie d'analyse et de modélisation de tâches d'interaction Homme-Machine avec des outils de spécification. *Revue Européenne Diagnostic et Sureté de fonctionnement*, 5(2) :159–179.

[Abed et al., 2001] Abed, M., Ezzedine, H., and Kolski, C. (2001). Modélisation des tâches dans la conception et l'évaluation des systèmes interactifs : la méthode SADT/Petri. In Kolski, C., editor, *Analyse et Conception de l'IHM. Interaction Homme-Machine pour les SI*, pages 145–174. éditions Hermès.

[Abed and Tabary, 1998] Abed, M. and Tabary, D. (1998). TOOD : Une méthodologie de Description Orientée Objet des Tâches utilisateur pour la spécification et la conception des Interfaces Homme-Machine. In *Acte des journées "Automatique et Homme" du club EEA*, pages 23–24.

[Abrial, 1996] Abrial, J.-R. (1996). *The B Book - Assigning Programs to Meanings*. Cambridge University Press.

[Adam, 2000] Adam, E. (2000). *Modèle d'organisation multi-agent pour l'aide au travail coopératif dans les processus d'entreprise : application aux systèmes administratifs complexes*. PhD thesis, Université de Valenciennes et du Hainaut-Cambrésis, Valenciennes.

[Alexander et al., 1977] Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiskdahl-King, I., and Angel, S. (1977). *A pattern language*. Oxford University Press, New York.

[Allen and Frost, 1998] Allen, P. and Frost, S. (1998). *Component-based Development for Enterprise Systems, Applying the Select Perspective*. Cambridge University Press.

[Almeida et al., 2003] Almeida, E. S., Bianchini, C. d. P., Prado, A. F., and Trevelin, L. C. (2003). IPM : An incremental process model for distributed component-based software development. In *The 5th International Conference on Enterprise Information System (ICEIS)*, pages 221–232, Angers. ACM Press.

[Alter, 1977] Alter, S. (1977). A taxonomy of decision support systems. *Sloan Management Review*, 19(1) :39–56.

[André, 1993] André, J. (1993). Le cycle de vie en y. rapport, Arche SQL.

[Aniorté, 2002] Aniorté, P. (2002). La dimension processus dans l'ingénierie par réutilisation de composants, de l'infrastructure de réutilisation à l'architecture logicielle distribuée. In *Actes du Workshop OCM-SI "objets, Composants, Modèles dans l'Ingénierie des Systèmes d'information"*, Nantes. GDR I3 PRC.

[Anli et al., 2004a] Anli, A., Petit-Rozé, C., and Grislin-Le Strugeon, E. (2004a). Plate-forme d'intégration de services personnalisés à base d'agents logiciels. *Génie Logiciel*, 71 :34–39.

[Anli et al., 2004b] Anli, A., Petit-Rozé, C., and Grislin-Le Strugeon, E. (2004b). User-based decision support system. In *International Conference on Advances in Intelligent Systems - Theory and Applications in cooperation with IEEE Computer Society, AISTA'2004*.

[Anthony, 1965] Anthony, R. (1965). *Planning and Control Systems : A Framework for Analysis*. Harvard University Graduate School of Business Administration, Cambridge, MA.

[Arnaud, 2003] Arnaud, M. (2003). Les limites actuelles de l'apprentissage collaboratif en ligne. *STICEF*, 10.

[Balasubramanian et al., 1999] Balasubramanian, P., Nochur, K., Henderson, J. C., and Millie Kwan, M. (1999). Managing process knowledge for decision support. *Decision Support Systems*, 27(1-2) :145–162.

[Barais and Duchien, 2004] Barais, O. and Duchien, L. (2004). TranSAT : Maîtriser l'Evolution d'une Architecture Logicielle. *L'Objet*, 10(2-3) :103–116.

[Barbier et al., 2002] Barbier, F., Cauvet, C., Oussalah, M., Rieu, D., Bennasri, S., and Souveyet, C. (2002). Composants dans l'ingénierie des systèmes d'information : concepts clés et techniques de réutilisation. In *Information Interaction Intelligence - Actes des 2e Assises nationales du GdR I3*, pages 95–117, Nancy. Cepaduès.

[Barthet, 1986] Barthet, M.-F. (1986). *Conceptions d'applications conversationnelles adaptées à l'utilisateur*. Thèse de doctorat, Université de Toulouse.

[Bass et al., 1998] Bass, L., Clements, P., and Kazman, R. (1998). *Software Architecture in Practice*. Addison Wesley.

[Bass et al., 1991] Bass, L., Mittle, R., Pellegrino, R., Reed, S., Seacord, S., Sheppard, S., and Szesur, M. (1991). The Arch Model : Seeheim revisited. In *Proceedings of User Interface Developper's Workshop*. Seeheim.

[Bastien and Scapin, 2001] Bastien, J. and Scapin, D. (2001). Evaluation des systèmes d'information et critères ergonomiques. In Kolski, C., editor, *Environnements évolués et évaluation de l'IHM. Interaction Homme-Machine pour les SI*, pages 53–79. éditions Hermès, Paris.

[Beck, 1999] Beck, K. (1999). *Extreme Programming Explained : EMbrace Change*. Addison-Wesley Publishing Company.

[Benali et al., 2002] Benali, K., Bourguin, G., David, B., Derycke, A., and Ferraris, C. (2002). Collaboration / coopération. In *gdr I3*.

[Biebow and Szulman, 2000] Biebow, B. and Szulman, S. (2000). Une approche terminologique pour catégoriser les concepts d'une ontologie. In Charlet, J., Zacklad, M., Kassel, G., and Bourrigault, D., editors, *Ingénierie des connaissances : évolutions récentes et nouveaux défis*. Eyrolles, Paris, France.

[Bodart et al., 1995] Bodart, F., Hennebert, A.-M., Leheureux, J.-M., Provot, I., Vanderdonckt, J., and Zucchinetti, G. (1995). Key activities for a development methodology of interactive applications. In *Critical Issues in User Interface Systems Engineering*, pages 109–134. Springer Verlag, Berlin.

[Boehm, 1981] Boehm, B. (1981). *Software Engineering Economics*. Prentice Hall, Englewoods Cliffs N.J.

[Boehm et al., 1984] Boehm, B. W., Gray, T. E., and Seewaldt, T. (1984). Prototyping versus specifying : a multiproject experiment. *IEEE transactions on Software Engineering*, 10(3) :290–302.

[Bonczek et al., 1981a] Bonczek, R., Holsapple, C., and Watson, A. (1981a). *Foundations of decision support systems*. Academic Press, NY, USA.

[Bonczek et al., 1981b] Bonczek, R. H., Holsapple, C. W., and Whinston, A. B. (1981b). The evolving roles of models in the decision support systems. *Sci.*, 11 :337–356.

[Booch et al., 2000] Booch, G., Rumbaugh, J., and Jacobson, I. (2000). *Le guide de l'utilisateur UML*. Eyrolles, Paris.

[Borchers, 2001] Borchers, J. (2001). *A pattern approach to interaction design*. WILEY, Chichester, England.

[Bouchet, 2003] Bouchet, J. (2003). Approche à composants pour la conception et le développement d'interfaces multimodales. Rapport de DEA, Joseph-Fournier, Grenoble.

[Bouchet and Nigay, 2004] Bouchet, J. and Nigay, L. (2004). ICARE : A component-based approach for the design and development of multimodal interfaces. In *Extended Abstracts of CHI'04*, pages 1325–1328. Vienna, Austria.

[Bourguin, 2000] Bourguin, G. (2000). *Un support informatique à l'activité coopérative fondé sur la théorie de l'activité : le projet DARE*. Thèse de doctorat, Université de Lille.

[Bourguin and Derycke, 2001] Bourguin, G. and Derycke, A. (2001). Integrating the CSCL activities into vitrual campuses : Foudations of a new infrastructure for distributed collective activities. In Dillenbourg, E. H., editor, *euro - CSCL 2001*, pages 123–130.

[Bret et al., 2001] Bret, F., Cruanes, T., Guessarian, I., Metais, E., Rousset, M.-C., Schwer, S., Teste, O., and Zurfluh, G. (2001). Entrepôt de données pour l'aide à la décision. In Cauvet, C. and Rosenthal-Sabroux, C., editors, *Ingénierie des systèmes d'information*, pages 175–208. Hermès Science Publications, Paris.

[Buchanan et al., 1983] Buchanan, B., Barstow, D., Bechtel, R., Bennet, J., Clacey, W., Kulikowski, C., Mitchell, T., and Watermen, D. (1983). Constructing expert system. In Hayes-Roth, F., Waterman, D., and Lenat, D., editors, *Building Expert Systems*. Addison Wesley, Mass.

[Buisine, 1999] Buisine, A. (1999). *Vers une démarche industrielle pour le développement d'Interfaces Homme-Machine, De l'analyse de l'activité à la génération du code*. Thèse de doctorat, Université de Rouen.

[Caulier, 1997] Caulier, P. (1997). *Méthodologie de capitalisation et de réutilisation de connaissances pour l'aide à la supervision des procédés automatisés complexes - Application à la supervision du trafic téléphonique de l'Ile de France*. Thèse de doctorat, Université de Valenciennes et du Hainaut-Cambrésis.

[Caulier, 2001] Caulier, P. (2001). Méthodologie de conception de système d'information. In Zacklad, M. and Grunsdtein, M., editors, *Ingénierie et capitalisation des connaissances*, pages 109–132. Hermès Sciences Publication.

[Cauvet and Semmak, 1999] Cauvet, C. and Semmak, F. (1999). La réutilisation dans l'ingénierie des systèmes d'information. In Oussalah, C., editor, *Génie objet - Analyse et conception de l'évolution*, pages 25–55. Hermès Sciences Publications, Paris.

[Cervantes, 2004] Cervantes, H. (2004). *Vers un modèle à composants orienté services pour supporter la disponibilité dynamique*. Thèse de doctorat, Université Joseph-Fourier.

[Checroun, 1992] Checroun, A. (1992). *Comprendre et utiliser les SIAD*. MASSON, Paris, Milan, Barcelone, Bonn.

[Christiansson et al., 2002] Christiansson, B., Jakobsson, L., and Crnkovic, I. (2002). CBD process. In Crnkovic, I. and Larsson, M., editors, *buiding reliable component-based Software*, pages 89–113. Artech House, MA.

[Cloux, 2003] Cloux, P.-Y. (2003). *RUP, XP architectures et outils - industrialiser le processus de développement*. Dunod, Paris.

[Cooke, 1994] Cooke, N. J. (1994). Varieties of knowledge elicitation techniques. *International journal of human-computer studies*, 41 :801–849.

[Coriat and Weinstein, 1995] Coriat, B. and Weinstein, O. (1995). *Les nouvelles théories de l'entreprise*. LGF - Livre de Poche, Paris, France.

[CORYS TESS, 2004] CORYS TESS (2004). SAMURAIL. http ://www.corys.fr/technologie/corail.php.

[Coulange, 1996] Coulange, B. (1996). *Réutilisation du logiciel*. MASSON, Paris, France.

[Courbon et al., 1981] Courbon, J.-C., Grageof, J., and Tomasi, J. (1981). L'approche évolutive. *Informatique et Gestion*, 21 :29–34.

[Courtney, 2001] Courtney, J. F. (2001). Decision making and knowledge management in inquiring organization : toward a new decision-making paradigm for DSS. *Decision Support Systems*, 31(1) :17–38.

[Coutaz, 1987] Coutaz, J. (1987). PAC : an implemantation model for dialog design. In Bullinger, H. and Shakel, B., editors, *Proceedings of Interact'87*, pages 431–436, Stuttgart, North Holland. Elsevier Science Publishers.

[Coutaz, 1990] Coutaz, J. (1990). *Interfaces homme-ordinateur*. Bordas, Paris.

[Coutaz and Nigay, 2001] Coutaz, J. and Nigay, L. (2001). Architecture logicielle conceptuelle des systèmes interactifs. In Kolski, C., editor, *Analyse et conception de l'IHM. Informatique et systèmes d'information*. Hermès Science Publications, Paris.

[Curtis and Hefley, 1994] Curtis, B. and Hefley, B. (1994). a WIMP no more, the maturing of user interface engineering. *Interactions*, pages 22–34.

[David and Foray, 1995] David, P. and Foray, D. (1995). Dépendance du sentier et économie de l'innovation : un rapide tour d'horizon. *Revue d'Economie industrielle*, pages 27–52.

[Debernard et al., 2004] Debernard, S., Poulain, T., and Guiost, B. (2004). Assister les opérateurs humains de supervision : une approche expérimentale de coopération homme-machine dans le contrôle aérien. *REE*, pages 36–42.

[Delorme, 2003] Delorme, X. (2003). *Modélisation et résolution de problèmes liés à l'exploitation d'infrastructures ferroviaires*. PhD thesis, Université de Valenciennes et du Hainaut-Cambrésis.

[Demeure and Najm, 2002] Demeure, I. and Najm, E. (2002). *Les intergiciels : développements récents dans CORBA, Java RMI et les agents mobiles*. Hermès - Lavoisier.

[Derycke et al., 2003] Derycke, A., Hoogstoël, F., and Le Pallec, X. (2003). The reciprocity project : a P2P approach to collaborative environnements for life-long learning. In *Frontiers in Education, FIE 2003*.

[Détienne and Burkhardt, 2001] Détienne, F. and Burkhardt, J.-M. (2001). Des aspects d'ergonomie cognitive dans la réutilisation en génie logiciel. In Dao, M. and Dony, C., editors, *La réutilisation*, Technique et sciences informatiques, pages 461–487. Hermès, Paris.

[Diaper and Stanton, 2004] Diaper, D. and Stanton, N. (2004). *The handbook of task Analysis for Human-Computer Interaction*. Lawrence Erlbaum Associates.

[D'Souza, 2003] D'Souza, D. (2003). Component and Catalysis Cost-Effective Software Development. www.catalysis.org.

[Duribreux-Cocquebert, 1995] Duribreux-Cocquebert, M. (1995). *MODESTI : vers une méthodologie interactive de développement de Systèmes à Base de Connaissances*. PhD thesis, Université de Valenciennes et du Hainaut-Cambrésis.

[EPFL, 1991] EPFL (1991). Fasta analyse de la stabilité d'horaires sur un réseau ferroviaire maillé. Technical report, ITEP de l'Ecole Polytechnique Fédérale de Lausanne.

[Ermine, 1993] Ermine, J.-L. (1993). *Génie Logiciel et Génie Cognitif pour les systèmes à base de connaissances*. Lavoisier, Paris.

[Ermine, 1996] Ermine, J.-L. (1996). *Les systèmes de connaissances*. Hermès, Paris.

[Ermine, 2000] Ermine, J.-L. (2000). *Les systèmes de connaissances*. Hermès science publications, Paris.

[Ermine, 2001] Ermine, J.-L. (2001). Capitaliser et partager les connaissances avec la méthode MASK. In Zacklad, M. and Grundstein, M., editors, *Ingénierie et capitalisation des connaissances*, pages 67–105. Hermes Sciences Publication, Paris, France.

[Ermine, 2003] Ermine, J.-L. (2003). *La gestion des connaissances*. Hermès, Lavoisier, Paris.

[ESA, 1991] ESA (1991). Software engineering standards. Technical Report 2, European Space Agency.

[Farenc, 1997] Farenc, C. (1997). *ERGOVAL : une méthode de structuration des règles ergonomiques permettant l'évaluation automatique d'interfaces graphiques*. PhD thesis, Université Toulouse 1.

[Fontaine and Gauyacq, 2001] Fontaine, M. and Gauyacq, D. (2001). SISYFE : a toolbox to simulate the railway network functioning for many purposes. Some cases of application. In *World Congress on Railway Research*, Köln.

[Front-conte et al., 1999] Front-conte, A., Giraudin, J. P., Rieu, D., and Saint-Marcel, C. (1999). Réutilisation et patrons d'ingénierie. In *Génie Objet : Analyse et Conception de l'évolution*, pages 91–136. Hermès, Paris.

[Frost, 1995] Frost, S. (1995). The select perspective, version 4.0. Technical report.

[Gabay, 2001] Gabay, J. (2001). *Merise et UML pour la modélisation des systèmes d'information*. Dunod, Paris.

[Gacek et al., 1995] Gacek, C., Abd-Allah, A., Clark, B., and Boehm, B. (1995). On the definition of software system architecture. In *Proceedings of the first International Workshop on Architectures for Software Systems - In Cooperation with the 17th International Conference on Software Engineering*, Seattle.

[Gamboa-Rodriguez, 1998] Gamboa-Rodriguez, F. (1998). *Spécification et implémentation d'ALACIE : Atelier Logiciel d'Aide à la Conception d'Interfaces Ergonomiques*. PhD thesis, Université de Paris Sud.

[Gamma et al., 1994] Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1994). *Design Patterns : Elements of reusable Object-Oriented Software*. Addison Wesley, Massachusetts.

[Gamma et al., 1999] Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1999). *Design Patterns, Catalogue de modèles de conception réutilisables*. Vuibert, Paris.

[Gorry and Scott Morton, 1971] Gorry, G. A. and Scott Morton, M. S. (1971). A framework for management information systems. *Sloan Management Review*, 13(1) :50–70.

[Goverde and Odijk, 2002] Goverde, R. M. P. and Odijk, M. A. (2002). Performance evaluation of network timetables using PETER. In Allan, J., Andersson, E., Brebbia, C. A., Hill, R. J., Sciutto, G., and Sone, S., editors, *Computers in Railway VIII*, Southampton. Wit Press.

[Gray, 2001] Gray, P. H. (2001). A problem-solving perspective on knowledge management practices. *Decision Support Systems*, 31 :87–102.

[Grislin, 1995] Grislin, M. (1995). *Définition d'un cadre pour l'évaluation a priori des interfaces homme-machine dans les systèmes industriels de supervision*. PhD thesis, Université de Valenciennes et du Hainaut-Cambrésis.

[Grislin and Kolski, 1996] Grislin, M. and Kolski, C. (1996). Evaluation des interfaces homme-machine lors du développement de système interactif. *Technique Science Informatiques TSI*, 2(265-296).

[Grudin, 1992] Grudin, J. (1992). Utility and usability : research issues and development contexts. *Interacting with Computers*, 4(2) :209–217.

[Hachemane, 1997] Hachemane, P. (1997). *Evaluation de la capacité de réseaux ferroviaires*. PhD thesis, Ecole Polytechnique Fédérale de Lausanne.

[Hachette 2004] Hachette 2004 (2003). *Dictionnaire*. Hachette Livre, Paris.

[Hart, 1988] Hart, A. (1988). *Acquisition du savoir pour les sytemes experts*. MASSON, Paris.

[Hartson and Hix, 1989] Hartson, H. and Hix, D. (1989). Towards empirically derived methodologies and tools for human-computer development. *International Journal of Man-Machines Studies*, 31 :477–494.

[Hassine et al., 2002a] Hassine, I., Rieu, D., Bounaas, F., and Seghrouchni, O. (2002a). Towards a reusable business components model. In *Workshop on Reuse in Object Oriented Information Systems Design*, Montpellier, France. OOIS 2002.

[Hassine et al., 2002b] Hassine, I., Rieu, D., Bousnaas, F., and Seghrouchni, O. (2002b). Symphony : un modèle conceptuel de composants métier. In Cauvet, C., editor, *Connaissances métier en ingénierie des systèmes d'information*, Ingénierie des systèmes d'information, pages 35–59. hermès science, Paris.

[Herault, 2003] Herault, C. (2003). Adaptabilité des services techniques dans un modèle à composants. In *3eme Conférence Française sur les Systèmes d'Exploitation (CFSE)*, La Lolle Sur Loup, France.

[Hickman et al., 1989] Hickman, F., Killin, J., Land, L., Mulhall, T., Porter, D., and Taylor, R. (1989). *Analysis for Knowledge-based systems, a Practical guide to the KADS methodology*. Ellis Horwood.

[Hill, 1992] Hill, R. D. (1992). The abstraction -link-view paradigm : Using constraints to connect user interfaces to applications. In *Proceedings CHI'92*, pages 335–342. ACM Press.

[Hix and Hartson, 1993] Hix, D. and Hartson, H. (1993). *Developing User Interfaces : Ensuring Usability Through Product & Process*. John Wiley & Sons.

[Hodgson, 1991] Hodgson, R. (1991). The X-model : a process model for object-oriented software development. In *Actes du congrès Le génie Logiciel et ses applications*. Toulouse.

[Hoogstriema and Teunisse, 1998] Hoogstriema, J. S. and Teunisse, M. (1998). The use of simulation in the planning of the dutch railway services. In Medeiros, D. J., Watson, E. F., Carson, J. S., and Manivannan, M. S., editors, *Proceedings of the Winter Simulation Conference*, pages 1139–1145.

[Houriez, 1994] Houriez, B. (1994). *Acquisition de connaissances pour l'aide à la conduite et la supervision de procédés industriels*. Mémoire d'HDR, Université de Valenciennes et du Hainaut-Cambrésis.

[Hugues, 2000] Hugues, A.-M. (2000). Gestion de connaissances dans l'industrie du logiciel : contribution à la réutilisation ? In *Séminaire Systèmes Distribués et Connaissances*. INRIA Sophia Antipolis.

[Hugues et al., 1996] Hugues, B., Leblanc, B., and Morley, C. (1996). *RAD : une méthode pour développer plus vite*. InterEditions, Paris.

[ivt, 2005] ivt (2005). opentrack. http ://www.ivt.baug.ethz.ch/oev/opentrack_e.html.

[Jacobson et al., 1999] Jacobson, I., Booch, G., and Rumbaugh, J. (1999). *Le processus Unifié de Développement logiciel*. Editions Eyrolles, Paris.

[Jacobson et al., 1993] Jacobson, I., Christerson, M., Jonsson, P., and Overgaard, G. (1993). *Le génie logiciel orienté objet, une approche fondée sur les cas d'utilisation*. Addison-Wesley, Paris.

[Jambu, 1999] Jambu, M. (1999). *Introduction au Data Mining, analyse intelligente des données*. Eyrolles, Paris.

[James, 1991] James, M. G. (1991). *Taking Software Design Seriously*. Academic Press.

[Jaulent, 1994] Jaulent, P. (1994). *Génie Logiciel, les méthodes*. A. Colin, Paris.

[Jones, 1990] Jones, C. B. (1990). *Systematic software development using VDM*. Prentice-Hall, New York.

[Karsenty, 1993] Karsenty, A. (1993). Le collecticiel : de l'interaction homme-machine à la communication homme-machine-homme. *Technique et Science Informatique*.

[Keen and Scott-Morton, 1978] Keen, P. and Scott-Morton, M. (1978). *Decision Support System : an Organisational Perspective*. Addsion Wesley Publication Company, Reading, Mass.

[Kettner and Sewcyk, 2001] Kettner, M. and Sewcyk, B. (2001). NEMO - the network evaluation model. In *World Congress on Railway Research, WCRR 2001*. Köln.

[Khayati et al., 2003] Khayati, O., Conte, A., and Giraudin, J. P. (2003). Vers un modèle de recherche de composants. In *Colloque ALCAA2003*, pages 130–141, Bayonne, France.

[Khayati and Giraudin, 2002] Khayati, O. and Giraudin, J.-P. (2002). Components retrieval systems. In *OOIS, Workshop on Reuse in Object-Oriented Information Systems Design*.

[Kolski, 1995] Kolski, C. (1995). *Méthodes et modèles de conception et d'évaluation des interfaces homme-machine*. Mémoire d'HDR, Université de Valenciennes et du Hainaut-Cambrésis.

[Kolski, 1997] Kolski, C. (1997). *Interfaces homme-machine application aux systèmes industriels complexes*. Hermès, Paris, France.

[Kolski, 2001] Kolski, C. (2001). *Analyse et conception de l'IHM- Interaction homme-machine pour les SI 1*. Hermés Science Publication, Paris, France.

[Krasner and Pope, 1988] Krasner, G. and Pope, S. (1988). A cookbook for using the Model-View-Controller User Interface Paradigm in Smalltalk-80. *Journal of Object Oriented Programming*, pages 26–49.

[Kruchten, 2000] Kruchten, P. (2000). *Introduction au Rational Unified Process*. Eyrolles, Paris.

[Labbé, 2001] Labbé, R. (2001). *Méthodes d'aide à la décision. Approche théorique et études de cas développés*. Ellipses Edition Marketing, Paris.

[Labouisse and Djellab, 2001] Labouisse, V. and Djellab, H. (2001). Demiurge : A tool for the optimisation and capacity assessment for railway infrastructure. In *World Congress on Railway Research, WCRR 2001*, Köln.

[Lai, 1991] Lai, M. (1991). *Conception orientée objet : pratique de la méthode HOOD*. Dunod, Paris.

[Lapassat, 2003] Lapassat, G. (2003). *Urbanisme informatique et architectures applicatives*. Hermès-Lavoisier, Paris.

[Laprie et al., 1995] Laprie, J.-C., Arlat, J., Blanquart, J.-P., Costes, A., Crouzet, Y., Deswarte, Y., Fabre, J.-C., Guillermain, H., Kaâniche, M., Kanoun, K., Mazet, C., Powell, D., Rabéjac, C., and Thévenod-Fosse, P. (1995). Guide de la sureté de fonctionnement. rapport No 94090, Laboratoire d'Ingénierie de la Sureté de fonctionnement (LIS), Toulouse.

[Larvet, 1994] Larvet, P. (1994). *Analyse des systèmes : de l'approche fonctionnelle à l'approche objet*. InterEditions, Paris.

[Le, 2004] Le, A. T. (2004). *Fédération : une architecture logicielle pour la construction d'applications dirigée par les modèles*. Thèse de doctorat, Université Joseph Fourier.

[Le et al., 2003] Le, A. T., Villalobos, J., and Estublier, J. (2003). Multi-level composition for software federations. In *ETAPS-2003 Conference, Workshop on Software Composition (SC'2003)*.

[Lepreux, 2001] Lepreux, S. (2001). Conception de système interactif d'aide à l'évaluation de la capacité ferroviaire. rapport de DEA, Université de Valenciennes.

[Lepreux, 2003] Lepreux, S. (2003). Méthode de conception de SIAD centré sur le décideur et utilisant la programmation par composants. In *IHM 03, 15e conférence francophone sur l'Interaction Homme-Machine*, pages 295–298, Caen, France. ACM Press.

[Lepreux et al., 2001a] Lepreux, S., Abed, M., Kolski, C., Jung, S., and Legendre, M. (2001a). A methodology for decision support system design in railway capacity evaluation. In Lind, M., editor, *20th European Annual Conference on Human Decision Making and Manual Control, EAM2001*, pages 123–130, Copenhague. DTU.

[Lepreux et al., 2001b] Lepreux, S., Abed, M., Kolski, C., Jung, S., and Legendre, M. (2001b). Vers un système d'aide à l'évaluation de la capacité des réseaux ferroviaires. In *2ème Conférence Annuelle d'Ingénierie Système*, pages 193–202, Toulouse, France. Association Française d'Ingénierie Système.

[Lepreux et al., 2003a] Lepreux, S., Abed, M., and Kolski, C. (2003a). A human-centred methodology applied to decision support system design and evaluation in a railway network context. *Cognition, Technology & Work*, 5 :248–271.

[Lepreux et al., 2003b] Lepreux, S., Kolski, C., and Abed, M. (2003b). Design method for component-based DSS. In *International Workshop on Component-Based Business Information Systems Engineering (CBBISE)*. Geneve, Suisse.

[Lepreux et al., 2003c] Lepreux, S., Kolski, C., and Queric, G. (2003c). Towards a methodology for DSS user-centered design. In Harris, D., Duffy, V., Smith, M., and Stephanidis, C., editors, *Human-center computing : cognitive, social and ergonomic aspects*, pages 1289–1292, Crete. Lawrence Erlbaun Associates.

[Lepreux et al., 2003d] Lepreux, S., Queric, G., Legendre, M., and Kolski, C. (2003d). Decision support system for the investment in railway architecture. In *Proceedings WCRR2003 The world congress on Railway Research*, Edimbourg, Scotland.

[Lepreux et al., 2004a] Lepreux, S., Kolski, C., and Abed, M. (2004a). Decision support systems designs as knowledge-based tool integration. In Memon, A. and Zhao, N., editors, *Proceedings of the 2004*

IEEE International Conference on Information Reuse and Integration (IRI2004), pages 516–521, Las Vegas, USA. IEEE.

[Lepreux et al., 2004b] Lepreux, S., Kolski, C., and Abed, M. (2004b). IHM et SIAD : vers une composition d'outils interactifs pour l'aide à la décision. In *Proceddings of IHM 2004*, pages 227–230. International Conference Proceedings Series, ACM Press.

[Lévine and Pomerol, 1989] Lévine, P. and Pomerol, J.-C. (1989). *Systèmes interactifs d'aide à la décision et systèmes experts*. Hermès, Paris, France.

[Liao, 2003] Liao, S.-h. (2003). Knowledge management technologies and applications - litterature review from 1995-2002. *Expert Systems with applications*, 25 :155–164.

[Lim and Long, 1994] Lim, K. and Long, J. (1994). *The Muse Method for Usability Engineering*. Cambridge University Press.

[Lind, 1991] Lind, M. (1991). Decision models and the design of knowledge-based systems. In Rasmussen, J., Brehmer, B., and Leplat, J., editors, *Distributed Decision Making, Cognitive models for cooperative work*. John Wiley & sons, chichester, England.

[Mahfoudi, 1997] Mahfoudi, A. (1997). *TOOD : Une méthodologie de description orientée objet des tâches utilisateur pour la spécification et la conception des interfaces homme-machine*. PhD thesis, Université de Valenciennes et du Hainaut-Cambrésis.

[Maret and Pinon, 1997] Maret, P. and Pinon, J.-M. (1997). *Ingénierie des savoir-faire - compétences individuelles et mémoire collective*. Hermès, Paris.

[Maret et al., 1996] Maret, P., Poullet, L., and Pinon, J.-M. (1996). Des modèles conceptuels pour capitaliser la connaissance au sein d'une organisation. *Ingénierie des systèmes d'information*, 4(4) :491–540.

[McDermid and Ripkin, 1984] McDermid, J. and Ripkin, K. (1984). Life cycle support in the ADA environment. *University Press*.

[Meinadier, 2002] Meinadier, J.-P. (2002). *Le métier d'intégration de systèmes*. Hermes Science Publications, Paris, France.

[Meyer, 2000] Meyer, B. (2000). *conception et programmation oreintés objet*. Eyrolles, Paris.

[Microsoft, 2004] Microsoft (2004). .net. http ://www.microsoft.com/france/vstudio/communaute/.

[Middelkoop and Bouwman, 2001] Middelkoop, D. and Bouwman, M. (2001). Simone : Large scale train network simulations. In Peters, B., Smith, J., Medeiros, D., and Rohrer, M. W., editors, *2001 Winter Simulation Conference*, Arlington, VA. ACM.

[Millie Kwan and Balasubramanian, 2003] Millie Kwan, M. and Balasubramanian, P. (2003). Knowledgescope : managing knowledge in context. *Decision Support Systems*, 35 :467–486.

[Millot and Roussillon, 1991] Millot, P. and Roussillon, E. (1991). Man-machine cooperation in telerobotics : Problematics and methodologies. In *Second Symposium on Robotics*, Gif-sur-Yvette.

[Morch, 1997] Morch, A. (1997). Three levels of end-user tailoring customization, integration, and extension. In Kyng, M. and Mathiassen, L., editors, *Computers and Design in Context*, pages 51–76. The MIT Press, Cambridge, MA.

[Nanard, 2002] Nanard, J. (2002). Les patrons dans la conception et la réalisation d'hypermédias, vers une opérationalisation. *Revue d'Interaction Homme-Machine*, 3(1) :41–78.

[Nielsen, 1993] Nielsen, J. (1993). *Usability engineering*. Academic Press, Boston.

[Nigay, 1994] Nigay, L. (1994). *Conception et modélisation logicielles des systèmes interactifs : application aux interfaces multimodales*. Thèse de doctorat, Université Joseph-Fourier.

[Nonaka and Takeushi, 1995] Nonaka, I. and Takeushi, H. (1995). *The knowledge Creating Company : How Japanese companies create the dynamics of innovation*. Oxford University Press, New York.

[Nonaka et al., 2000] Nonaka, I., Toyama, R., and Ko, o. N. (2000). Seci, ba, leadership : a unifed model of dynamic knowledge creation. *Long Range Planning*, 33 :5–34.

[Nykanen, 2000] Nykanen, P. (2000). *Decision Support System from a Health Informatics Perspective*. PhD thesis, University of tempere.

[objectweb, 2005] objectweb (2005). Fractal. http ://www.objectweb.org/fractal/index.html.

[OMG, 2003] OMG (2003). Model Driver Architecture (MDA) guide version 1.0.1. Technical Report omg TC Document omg/2003-06-01, Object Management Group.

[OMG, 2004] OMG (2004). CORBA. http ://www.omg.org/#corba_corner.

[Pacaux-Lemoine et al., 1996] Pacaux-Lemoine, M.-P., Debernard, S., Crevits, I., and Millot, P. (1996). Cooperation between humans and machines : first results of an experimentation of a multi-level cooperative organisation in air traffic control. *Cooperative Supported Cooperative Work : The journal of Collaborative Computing*, 5 :299–321.

[Palma-dos Reis and Zahedi, 1999] Palma-dos Reis, A. and Zahedi, F. M. (1999). Designing personalized intelligent financial decision support systems. *Decision Support Systems*, 26(1) :31–47.

[Pascot and Bernadas, 1993] Pascot, D. and Bernadas, C. (1993). L'essence des méthodes : étude comparative de six méthodes de conception de systèmes d'information informatisés. In *Actes INFORSID'93 Systemes d'information, systèmes à base de connaissances*, Lille.

[Paterno, 2001] Paterno, F. (2001). Tasks models in interactive software systems. In Chang, S. K., editor, *Handbook of Software Engineering and Knowledge Engineering*, pages 817–836. World Scientific Publishing Co.

[Paulhac et al., 2002] Paulhac, G., Jung, S., Legendre, M., Abed, M., Kolski, C., and Lepreux, S. (2002). Infrafer, outil d'aide à la conception des infrastructures ferroviaires. Technical Report final de projet PREDIT, CORYS-TESS, RFF, LAMIH.

[Payet, 2003] Payet, D. (2003). *L'enrichissement de message comme support pour la composition logicielle*. Thèse de doctorat, Universite de Montpellier 2.

[Petit, 2003] Petit, D. (2003). *Génération automatique de composants logiciels sûrs à partir de spécifications formelles B*. Thèse de doctorat, Université de Valenciennes et du Hainaut-Cambrésis.

[Petit-Rozé, 2003] Petit-Rozé, C. (2003). *Organisation multi-agents au service de la personnalisation de l'information*. Thèse de doctorat, Université de Valenciennes et du Hainaut-Cambrésis.

[Pfaff, 1985] Pfaff, G. (1985). *User Interface Management Systems*. Springer-Verlag, New York.

[Poltrock and Grundin, 1995] Poltrock, S. and Grundin, J. (1995). Groupeware and workflow : A survey of systems and behavioral issues. In *CHI'95 Proceedings of the conference on Human Factors in Computing Systems : Mosaic of Creativity.*, pages 355–356. ACM Press.

[Power, 2002] Power, D. J. (2002). A brief history of decision support systems. http ://dssresources.com/history/dsshistory.html.

[Prax, 2003] Prax, J.-Y. (2003). *Le manuel du Knowledge Management*. Dunod, Paris.

[Prieto-Diaz and Freeman, 1987] Prieto-Diaz, R. and Freeman, P. (1987). Classifying software for reusability. *IEEE Software*, pages 6–17.

[Printz, 1996] Printz, J. (1996). Genie logiciel. In *Techniques de l'ingénieur*, pages 1–32. Paris, France.

[Puerta and Maulsby, 1997] Puerta, A. R. and Maulsby, D. (1997). Management of interface design knowledge with mobi-d. In *IUI'97*, pages 249–252.

[Radtke and Bendefeldt, 2001] Radtke, A. and Bendefeldt, J.-P. (2001). Handling of railway operation with railsys. In *World Congress on Railway Research*, Köln.

[Ramadour, 2002] Ramadour, P. (2002). Approche et modèle pour la spécification de composants métier. In Cauvet, C., editor, *Connaissances métier en ingénierie des systemes d'information*, pages 61–81. Hermès, Paris.

[Rasmussen, 1983] Rasmussen, J. (1983). Skill, rules and knowledge ; signals, signs, and symbols, and other distinctions in human performance models. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-13(3) :257–267.

[Rational Software, 2004] Rational Software (2004). Rup. http ://www-136.ibm.com/developerworks/rational/products/rup/.

[RFF, 2000] RFF (2000). Recherche sur la saturation des lignes ferroviaires. Technical Report 2166/ESF/317-97/RA, RFF.

[RFF, 2005] RFF (2005). site du réseau ferré de france. www.rff.fr.

[Robert, 2003] Robert, J.-M. (2003). Que faut-il savoir sur les utilisateurs pour réaliser des interfaces de qualité ? In Boy, G., editor, *Ingénierie cognitive, IHM et cognition*, pages 249–283. Hermès Lavoisier, Paris.

[Roche and Million-Rousseau, 2003] Roche, C. and Million-Rousseau, C. (2003). Construction de terminologies métier : l'importance du modele ontologique. *Extraction des connaissances et apprentissage*, 17(1-2-3/2003) :313–318.

[Rolland et al., 2000] Rolland, C., Nurcan, S., and Grosz, G. (2000). A decision-making pattern for guiding the enterprise knowledge development process. *Information and software technology*, 42 :313–331.

[Sandoval, 1997] Sandoval, V. (1997). *L'informatique décisionnelle*. Hermès, Paris.

[Sanlaville, 2002] Sanlaville, R. (2002). *Architecture logicielle : une expérimentation industrielle avec Dassault Systemes*. Thèse de doctorat, Université Joseph-Fourier.

[Satzinger et al., 2002] Satzinger, J., Jackson, R., and Burd, S. (2002). *Analyse et conception de systèmes d'information*. Editions Reynald Goulet Inc., Canada.

[Saumont and Mirecourt, 2003] Saumont, P.-Y. and Mirecourt, A. (2003). *Le guide du développeur Java2. Meilleures pratiques avec Ant, JUnit et les design patterns*. Eyrolles, Paris.

[Scapin and Bastien, 1997] Scapin, D. and Bastien, J. (1997). Ergonomic criteria for evaluating the ergonomic quality of interactive systems. *Behaviour & Information Technology*, 16(4/5) :220–231.

[Sen, 1998] Sen, A. (1998). From DSS to DSP : A taxonomic retrospection. *Communications of the ACM*, 41(5) :206–216.

[Senach, 1990] Senach, B. (1990). Evaluation ergonomique des interfaces homme-machine une revue de la littérature. rapport 1180, Institut National de Recherche en Informatique et en Automatique, Sophia Antipolis, France.

[Shackel, 1991] Shackel, B. (1991). Usability - context, framework, design and evaluation. In *Human Factors for informatics Usability*, pages 21–38. Cambridge University Press, Cambridge.

[Shalloway and Trott, 2002] Shalloway, A. and Trott, J. R. (2002). *Design Pattern par la pratique*. Editions Eyrolles.

[Shamsfard and Barforoush, 2004] Shamsfard, M. and Barforoush, A. A. (2004). Learning ontologies from natural language textes. *International Journal of Human-Computer Studies*, 60 :17–63.

[Shim et al., 2002] Shim, J. P., Warkentin, M., Courtney, J. F., Power, D. J., Sharda, R., and Carlsson, C. (2002). Past, present and future of decision support technology. *Decision Support Systems*, 33(2) :111–126.

[Silvestre and Verlhac, 1996] Silvestre, P. and Verlhac, D. (1996). Stratégie de conception des systèmes d'information. In *Techniques de l'ingénieur*, pages 1–24. Paris, France.

[Simon, 1960] Simon, H. (1960). *The New Science of Management Decision*. Harper Brothers, New York.

[Simon, 1972] Simon, H. A. (1972). Theories of bounded rationality. In Rather, C. and Radner, R., editors, *Decision and Organisation*. North Holland, Amsterdam.

[SMA, 1999] SMA (1999). Viriato- le logiciel pour la planification de concepts d'offre. Technical report, Zurich, Suisse.

[SMA, 2002] SMA (2002). Viriato. http ://www.sma-partner.ch/vp/vstart.php ?lang=e.

[Spivey, 1989] Spivey, J. M. (1989). *The Z notation : a reference manual*. Prentice-Hall, Englewood Cliffs, N. J.

[Sprague, 1987] Sprague, R. (1987). DSS in context. *Decision Support Systems*, 3 :197–202.

[Stary, 2003] Stary, C. (2003). Activiy theory and task-based user-interface development. *Revue d'Interaction Homme-Machine*, 5(1) :65–87.

[sun microsystems, 2004] sun microsystems (2004). J2EE. http ://java.sun.com/index.jsp.

[SYSTRA Consulting, 2005] SYSTRA Consulting (2005). railsim simulation software suite. http ://www.railsim.com/default.htm.

[Szekely, 1996] Szekely, P. (1996). Retrospective and challenges for model-based interface development. In *Design, Specification and Verification of Interactive systems '96 : Proc. of 3rd Eurographics Workshop on Design, Specification, Verification of Interactive Systems DSV-IS'96 (Namur, 5-7 juin 1996)*, EG Books, pages 1–27. Springer-Verlag, Vienne.

[Szyperski, 1998] Szyperski, C. (1998). *Component-Software-Beyond Object-Oriented Programming*. Addison Wesley, MA.

[Tabary, 2001] Tabary, D. (2001). *Contribution à TOOD, une méthode à base de modèles pour la spécification et la conception des systèmes interactifs*. Thèse de doctorat, Université de Valenciennes et du Hainaut-Cambrésis.

[Tarby and Barthet, 1996] Tarby, J. C. and Barthet, M. F. (1996). The DIANE + method. In Vanderdonckt, J., editor, *Proceedings of the 2nd International Worshop on Computer-aided Design of User Interfaces CADUI'96*, pages 95–119, Namur. Presses Universitaires de Namur.

[Tarby and Barthet, 2001] Tarby, J. C. and Barthet, M. F. (2001). Analyse et modélisation des tâches dans la conception des systèmes d'information : la méthode diane +. In Kolski, C., editor, *Analyse et conception de l'IHM*, pages 117–144. Hermès.

[Tardieu et al., 1986] Tardieu, H., Rochfield, A., and Colletti, R. (1986). *La méthode MERISE 1*. Editions d'organisation, Paris.

[Tardieu et al., 2000] Tardieu, H., Rochfield, A., and Colletti, R. (2000). *La Méthode MERISE : principes et outils*. Editions d'organisation, Paris.

[Thayer and McGettrick, 1993] Thayer, R. and McGettrick, A. (1993). *Software Engineering : a European Perspective*. Los Alamitos, IEEE Computer Society Press.

[Vahidov and Kersten, 2004] Vahidov, R. and Kersten, G. E. (2004). Decision station : situating decision support systems. *Decision Support Systems*, 38(2) :283–303.

[Verjus, 2001] Verjus, H. (2001). *Conception et construction de fédérations de progiciels*. Thèse de doctorat, Université de Savoie.

[Villalobos, 2003] Villalobos, J. (2003). *Fédération de composants : une architecture logicielle pour la composition par coordination*. Thèse de doctorat, Université Joseph - Fourier.

[Vogel, 1988] Vogel, C. (1988). *Génie cognitif*. MASSON.

[Warkentin et al., 1997] Warkentin, M. E., Sayeed, L., and Hightower, R. (1997). Virtual teams versus face-ti face teams : an explonatory study of a web-based conference system. *Decision Sciences*, 28(4) :975–996.

[Wielinga et al., 1992a] Wielinga, B., Schreiber, G., and Breuker, J. (1992a). KADS : a modelling approach to knowledge engineering. *Knowledge Acquisition*, 4 :5–53.

[Wielinga et al., 1992b] Wielinga, B., Van de Velde, W., Schreiber, G., and Akkermans, H. (1992b). The KADS knowledge modelling approach. In Mizoguchi, R. and Motoda, H., editors, *Proceedings of the 2nd Japaneses Knowledge Acquisition for Knowledge-Based Systems Workshop*, pages 23–42, Hatoyama, Saitama, Japan.

[Wielinga et al., 1993] Wielinga, B., Van de Velde, W., Schreiber, G., and Akkermans, H. (1993). Towards a unification of knowledge modelling approaches. In Davide, J.-M., Krivine, J.-P., and Simmons, R., editors, *Second Generation Expert Systems*, pages 299–335. Springer Verlag.

[Xanthakis et al., 2000] Xanthakis, S., Régnier, P., and Karapoulios, C. (2000). *Le test des logiciels*. Hermès, Paris, France.

[Yim et al., 2004] Yim, N.-H., Kim, S.-H., Kim, H.-W., and Kwahk, K.-H. (2004). Knowledge based decision making on higher level strategic concerns : system dynamics approach. *Experts Systems with Applications*, 27(1) :143–158.

[Zacklad and Grundstein, 2001a] Zacklad, M. and Grundstein, M. (2001a). *Ingénierie et capitalisation des connaissances*. Hermès Science Publications, Paris, France.

[Zacklad and Grundstein, 2001b] Zacklad, M. and Grundstein, M. (2001b). *Management des connaissances*. Hermes Science Publication, Paris, France.

[Zighed and Rakotomalala, 2002] Zighed, D. A. and Rakotomalala, R. (2002). Extraction de connaissance à partir de données. *Techniques de l'ingénieur*, HA(H3744).

[Zopounidis et al., 1997] Zopounidis, C., Doumpos, M., and Matsatsinis, N. F. (1997). On the use of knowledge-based decision support systems in financial management : A survey. *Decision Support Systems*, 20(3) :259–277.

[Zwaneveld et al., 2001] Zwaneveld, P. J., Kroon, J. G., and van Hoesel, S. P. M. (2001). Routing trains through a railway station based on a node packing model. *European journal of Operational Research*, 128 :14–33.

Annexe A

Documents relatifs aux évaluations

Cette annexe a pour objectif de présenter les documents qui ont été utilisés lors des évaluations de SIADIF.

Sommaire

A.1 Protocole d'évaluation du SIAD (une demi-journée par sujet)

Etape 1 : Présentation théorique du système dans son ensemble (0h05)

- Temps : 5 minutes
- But : Présenter les éléments suivants :
  - But du SIAD
  - Différents modes de présentation d'outils :
    - présentation par thèmes
    - présentation par liste
    - présentation par mots-clés
  - Interfaces Homme-Machine
  - Outils accessibles
- Moyens techniques : diaporama commenté
- Questionnaire : pa de questionnaire particulier, il faut juste demander au sujet s'il a des questions
- Consigne : pas de consigne particulière

Etape 2 : Familiarisation avec les modes de recherche du système (0h20)

- Temps : 15 minutes
- But : entraînement à l'utilisation des 3 modes de recherche par rapport à des scénarios courts, assimiliation des outils existants dans le système.
- Moyens techniques : interaction avec le système ; le sujet peut être assisté par l'évaluateur.
- Questionnaire : pas de questionnaire particulier, il faut juste demander au sujet si il a des remarques.
- Consignes : utiliser le système sans appréhension

Etape 3 : Analyse des modes de choix d'outils de mesure (1h20)

1. Présentation du problème à traiter

- Temps : 5 minutes
- But : présenter le problème : augmentation de la capacité du réseau au niveau de Hazebrouck (Problème 1)
- Moyens techniques : le sujet peut être assisté par l'évaluateur.
- Questionnaire 1

2. Mise en situation (3fois)

- Consigne : choix d'une méthode et justification de son choix
- Temps : 15 minutes
- But : utilisation d'une aide au choix
- Moyens techniques : interaction avec le système ; magnétoscope complété d'un micro

- Consigne : n'utiliser que le mode que le sujet a choisi pour aboutir à une sélection d'outils
- Questionnaire 2
- Questionnaire 3 à a seconde itération
- Questionnaire 4 à la troisième itération

Etape 4 : Auto-confrontation (1h50)

- Temps : 30 minutes
- But : commentaires sur les parties critiques
- Moyens techniques : magnétoscope, télé, micro
- Consigne : le sujet doit commenter ses diificultés
- Questions orales :
  - Quelles sont les causes de vos difficultés ?
  - Comment améliorer SIADIF pour que vous ne soyez plus en difficulté ?

Pause (2h00)

Etape 5 : Mise en situation pour d'autres problèmes ferroviaires (2 fois : 0h30)

1. Présentation du problème à traiter

- Temps : 5 minutes
- But : présenter le problème
- Moyens techniques : le sujet peut être assisté par l'évaluateur.
- Questionnaire 5

2. Mise en situation

- Consigne : utilisation du mode liste, puis thèmes puis Mots-clés
- Temps : 15 minutes
- But : comparaison des résultats sortis des trois modes
- Moyens techniques : interaction avec le système ; grille de comparaison, repérage des mots-clés saisis par le sujet car ils seront difficilement lisible sur l'enregistrement ; magnétoscope complété d'un micro
- Questionnaire 6

A.2 Enoncé du problème 1

HAZEBROUCK : augmentation de la capacité du réseau

Finalités :

– Renforcer les dessertes entre Lille ou Arras et le littoral, éventuellement sous la forme d'un cadencement.
– Permettre un écoulement du fret au niveau national, ce projet s'inscrivant dans la logique du développement sur l'artère Nord-Est.

Consistance :

Les travaux comprennent :

1. L'aménagement des voies :
   (a) Suppression de la bifurcation de Haute Loge.
   (b) Création de 2 nouvelles voies de circulation traversant la gare d'Hazebrouck et permettant de séparer les axes Lille-Dunkerque et Arras-Calais et où les trains circulent à 90 km/h.
   (c) Dédoublement des jonctions vers Calais afin de permettre des simultanéités.
   (d) Maintien à la bifurcation d'Hazebrouck du raccordement à niveau des voies de Béthune aux voies de Lille-Calais.
2. La création d'un poste d'aiguillage unique de technologie moderne offrant aux agents chargés de la circulation une vision d'ensemble de toute la zone du nœud ferroviaire.

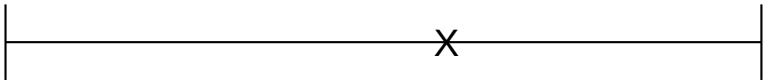
A.3 Questionnaire 1

Nom :	**Prénom :**
Emploi :	**Service :**
Age :	**Date :**

Exemple :

Aimez-vous les vacances ?

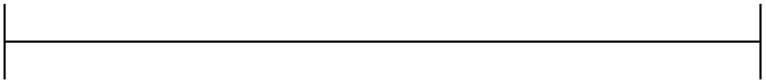
Pas du Tout — Parfaitement



Position par rapport au problème présenté

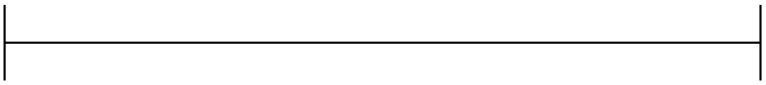
Avez-vous compris le problème ?

Pas du Tout — Parfaitement



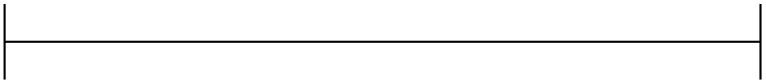
Ce problème vous semble-t-il complexe à traiter ?

Pas du Tout — Parfaitement



Vous considérez-vous comme expert pour la résolution de ce problème ?

Pas du Tout — Parfaitement



Quelles sont les études qui permettraient de vérifier le bien fondé de cet investissement ?

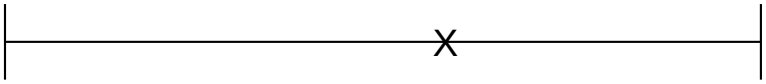
A.4 Questionnaire 2

Nom :	**Prénom :**
Emploi :	**Service :**
Age :	**Date :**

Exemple :

Aimez-vous les vacances ?

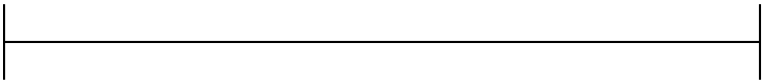
Pas du Tout — Parfaitement



Position par rapport au mode de choix retenu

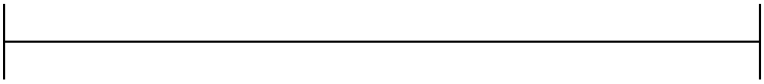
Etes-vous satisfait des outils qui ont été sélectionnés (des propositions du système s'il y en a eu) ?

Pas du Tout — Parfaitement



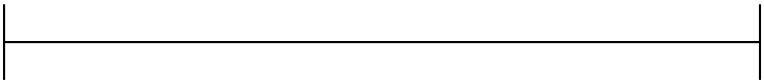
Avez-vous trouvé complexe le mode de recherche que vous avez choisi ?

Pas du Tout — Parfaitement



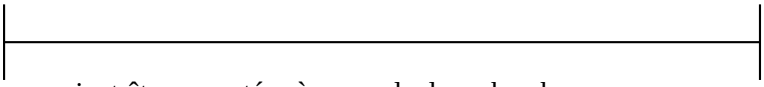
Parmi les outils proposés et sélectionnés, manque-t-il des outils de mesure pour résoudre ce problème ?

Pas du Tout — Parfaitement



Etes-vous satisfait de l'interaction avec le système ?

Pas du Tout — Parfaitement



Quelles améliorations pourraient être apportées à ce mode de recherche,

– En terme d'interaction ? (réponse orale)
– En terme de fonctionnalités ? (réponse orale)

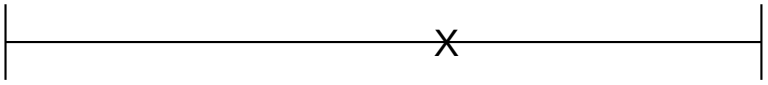
A.5 Questionnaire 3

Nom :	**Prénom :**
Emploi :	**Service :**
Age :	**Date :**

Exemple :

Aimez-vous les vacances ?

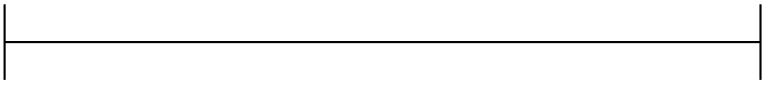
Pas du Tout — Parfaitement



Position par rapport au mode de choix retenu

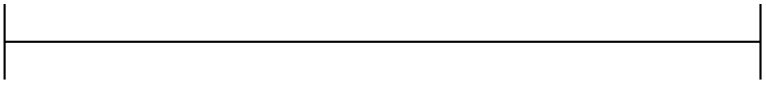
Etes-vous satisfait des outils qui ont été sélectionnés (des propositions du système s'il y en a eu) ?

Pas du Tout — Parfaitement



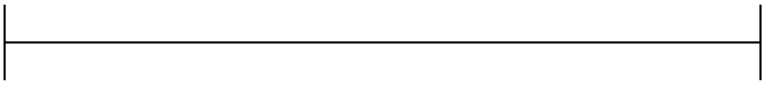
Avez-vous trouvé complexe le mode de recherche que vous avez choisi ?

Pas du Tout — Parfaitement



Parmi les outils proposés et sélectionnés, manque-t-il des outils de mesure pour résoudre ce problème ?

Pas du Tout — Parfaitement



Est-ce que les outils que vous avez sélectionnés sont les mêmes qu'avec le mode précédent ?

OUI NON

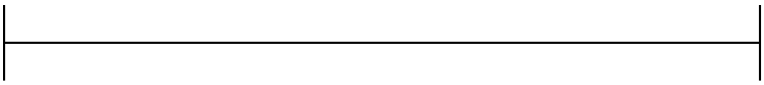
Si non, la différence se situe-t-elle quantitativement ou au niveau du type d'outil ?
Quelle est la raison de cette différence ?

- Difficulté à trouver l'outil recherché
- L'outil manquant n'a pas été proposé par le mode de recherche
- Autre _____

Parmi les modes que vous avez utilisés, quel mode de recherche avez-vous préféré, pourquoi ?

Etes-vous satisfait de l'interaction avec le système ?

Pas du Tout Parfaitement



Quelles améliorations pourraient être apportées à ce mode de recherche,

– En terme d'interaction ? (réponse orale)
– En terme de fonctionnalités ? (réponse orale)

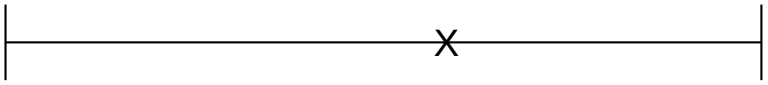
A.6 Questionnaire 4

Nom :	**Prénom :**
Emploi :	**Service :**
Age :	**Date :**

Exemple :

Aimez-vous les vacances ?

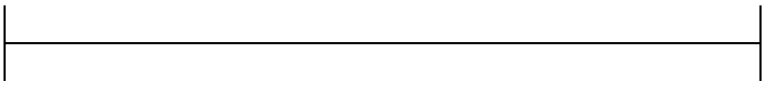
Pas du Tout — Parfaitement



Position par rapport au mode de choix retenu

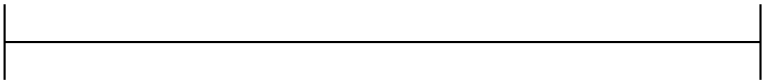
Etes-vous satisfait des outils qui ont été sélectionnés (des propositions du système s'il y en a eu) ?

Pas du Tout — Parfaitement



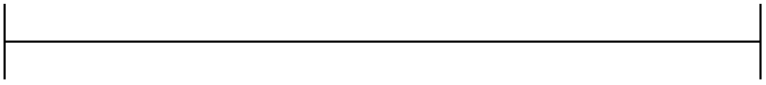
Avez-vous trouvé complexe le mode de recherche que vous avez choisi ?

Pas du Tout — Parfaitement



Parmi les outils proposés et sélectionnés, manque-t-il des outils de mesure pour résoudre ce problème ?

Pas du Tout — Parfaitement



Est-ce que les outils que vous avez sélectionnés sont les mêmes qu'avec les modes précédents ?

Oui ❒

Non ❒

Sinon, la différence se situe-t-elle :

– Quantitativement ❒

– Au niveau du type d'outil ? ❒

Quelle est la raison de cette différence ?

– Difficulté à trouver l'outil recherché. ❒

– L'outil manquant n'a pas été proposé par le mode de recherche. ❒

– Autres _____

Parmi les modes de recherche que vous avez utilisés, quel mode avez-vous préféré, pourquoi ?

_____

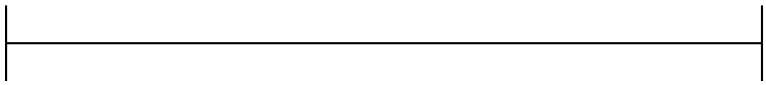
_____

_____

_____

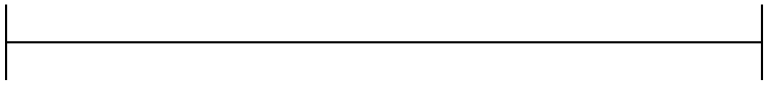
Etes-vous satisfait de l'interaction avec le système ?

Pas du Tout Parfaitement



Etes-vous satisfait de la présentation des outils par le système ?

Pas du Tout Parfaitement



Quelles améliorations pourraient être apportées à ce mode de recherche,

– En terme d'interaction ? (réponse orale)

– En terme de fonctionnalités ? (réponse orale)

Quelles améliorations pourraient être faites de manière globale pour la recherche d'outils

– En terme d'interaction ? (réponse orale)

– En terme de fonctionnalités ? (réponse orale)

A.7 Questionnaire Critères Ergonomiques

Nom :	**Prénom :**

Pour chaque critère d'ergonomie, vous donnerez un score traduisant de votre niveau d'accord et d'importance, en utilisant le principe de l'échelle de mesure suivante, à 5 points.

Pas de tout d'accord	Plutôt pas d'accord	Neutre	Plutôt d'accord	Tout à fait d'accord
1	2	3	4	5

1.Guidage					
1.1. Incitation	1	2	3	4	5
1.2.1. Groupement/ distinction entre items par localisation	1	2	3	4	5
1.2.2. Groupement /distinction entre items par le format	1	2	3	4	5
1.3. Feedback immédiat	1	2	3	4	5
1.4. Lisibilité	1	2	3	4	5
2. Charge de travail					
2.1. Brièveté	1	2	3	4	5
2.2. Densité informationnelles	1	2	3	4	5
3. Contrôle Explicite					
3.1. Actions explicites	1	2	3	4	5
3.2. Contrôle utilisateur	1	2	3	4	5
4. Adaptabilité					
4.1. Flexibilité	1	2	3	4	5
4.2. Prise en comte de l'expérience de l'utilisateur	1	2	3	4	5
5. Gestion des erreurs					
5.1. Protection contre les erreurs	1	2	3	4	5
5.2. Qualité des messages d'erreur	1	2	3	4	5
5.3. Correction des erreurs	1	2	3	4	5
6. Homogénéité/Cohérence	1	2	3	4	5
7. Signifiance des codes et dénominations	1	2	3	4	5
8. Compatibilité	1	2	3	4	5

Critères d'ergonomie	**Définitions**	**Exemples**
1.Guidage	**L'ensemble des moyens mis en œuvre pour conseiller, orienter informer, et conduire l'utilisateur lors de ses interactions avec l'utilisateur**	
1.1. Incitation	Recouvre les moyens mis en œuvre pour amener les utilisateurs à effectuer des actions spécifiques	Afficher les unités de mesures des données à saisir. Donner un titre à chaque fenêtre Guider les entrées de données en indiquant le format adéquat
1.2.1. Groupement/ distinction entre items par localisation	Positionnement des items les uns par rapport aux autres dans le but d'indiquer leur appartenance ou non appartenance à une même classe.	Organiser des items en listes hiérarchiques Grouper les options de menus en fonction des objets sur lesquelles ils s'appliquent
1.2.2. Groupement/ distinction entre items par le format	Il concerne les caractéristiques graphiques permettant de faire apparaître l'appartenance ou non appartenance à une même classe.	Etablir une distinction visuelle entre des aires ayant des fonctions différentes (commande, message…) Etablir une distinction visuelle entre les labels et les champs d'entrée
1.3. Feedback immédiat	Il concerne les réponses de l'ordinateur consécutives aux actions des utilisateurs.	Toujours faire apparaître sur l'écran les entrées effectuées par les utilisateurs.
1.4. Lisibilité	Il concerne les caractéristiques lexicales de présentation des informations sur l'écran pouvant entraver ou faciliter la lecture de ces informations.	Les titres doivent être centrés, Les labels doivent être en majuscule Le curseur doit être facilement repérable.
2. Charge de travail	**Il concerne l'ensemble des éléments de l'interface qui ont un rôle dans la réduction de la charge perceptive ou mnésique des utilisateurs et dans l'augmentation de l'efficacité du dialogue**	

Critères d'ergonomie	**Définitions**	**Exemples**
2.1. Brièveté	Il concerne la charge de travail au niveau perceptif et mnésique pour les éléments d'entrée pu de sortie et les séquences d'entrées	Permettre aux utilisateurs des entrées de données courtes Minimiser le nombre d'étapes dans la sélection de menus Pour la saisie de données, afficher des champs appropriés, les valeurs par défaut
2.2. Densité informationnelles	Il concerne la charge de travail du point de vue perceptif, pour des ensembles et non pour des items	Limiter la densité informationnelle de l'écran en affichant seulement les informations nécessaires
3. Contrôle Explicite	**Il concerne à la fois la prise en compte par le système des actions explicites des utilisateurs et le contrôle qu'ont les utilisateurs sur le traitement de leurs actions**	
3.1. Actions explicites	Il concerne la relation pouvant exister entre le fonctionnement de l'application et les actions des utilisateurs.	Le système doit requérir une action explicite (ex : Entrée, validation, OK. . .) par l'utilisateur suite à une entrée de données ; aucun traitement ne devrait être la conséquence d'une autre action
3.2. Contrôle utilisateur	L'utilisateur doit toujours avoir la main, pouvoir contrôler le déroulement des traitements informatiques en cours.	Le curseur ne doit pas être déplacé sans contrôle des utilisateurs Ne pas passer d'une page écran à une autre sans contrôle de l'utilisateur
4. Adaptabilité	**Il concerne sa capacité à réagir selon le contexte, et selon les besoins et préférences des utilisateur**	
4.1. Flexibilité	Il concerne les moyens mis à la disposition des utilisateurs pour personnaliser l'interface afin de rendre compte de leurs stratégies ou habitudes et des exigences de la tâche	Quand les exigences sont imprécises, fournir aux utilisateurs une certaine latitude dans les possibilités de contrôle des affichages
4.2. Prise en comte de l'expérience de l'utilisateur	Il concerne les moyens mis en œuvre pour respecter le niveau d'expérience de l'utilisateur	Autoriser les utilisateurs expérimentés à contourner une série de sélections par menu en formulant directement des commandes ou par raccourcis clavier

Critères d'ergonomie	**Définitions**	**Exemples**
5. Gestion des erreurs	**Il concerne tous les moyens permettant d'une part d'éviter ou de réduire les erreurs, et d'autre part de les corriger lorsqu'elles surviennent**	
5.1. Protection contre les erreurs	Il concerne les moyens mis en place pour détecter et prévenir les erreurs d'entrées de données ou de commande	Quand l'utilisateur quitte une session et qu'il y a risque de perte de données, il doit y avoir un message le signalant. Les labels des champs doivent être protégés
5.2. Qualité des messages d'erreur	Il concerne la pertinence, la facilité de lecture et l'exactitude de l'information donnée aux utilisateurs sur la nature des erreurs commises et sur les actions à entreprendre pour les corriger	Fournir des messages d'erreurs orientés tâches Utiliser des messages d'erreurs aussi brefs que possible
5.3. Correction des erreurs	Il concerne les moyens mis à la disposition des utilisateurs pour leur permettre de corriger leurs erreurs	Fournir la possibilité de modifier les commandes lors de leur saisie
6. Homogénéité/Cohérence	**Il se réfère à la façon avec laquelle les choix de conception de l'interface sont conservés pour des contextes identiques, et sont différents pour des contextes différents**	**Localisation similaire des titres de fenêtres, Formats d'écrans similaires, Procédures similaires d'accès aux options des menus**
7. Signifiance des codes et dénominations	**Il concerne l'adéquation entre l'objet ou l'information affichée ou entrée, et son référent.**	**Les titres doivent véhiculer ce qu'ils représentent, et être distincts, rendre les règles d'abréviations explicites**
8. Compatibilité	**Il se réfère à l'accord pouvant exister entre les caractéristiques des utilisateurs et des tâches, et à l'organisation des sorties, des entrées et du dialogue d'une application donnée.**	**L'organisation des informations affichées doit être conforme à l'organisation des données à entrer.**

A.8 Enoncé du problème 2

VALENCIENNES-HIRSON : AMELIORATION DE LA REGULARITE

Finalités :

Augmenter e débit de la circulation et améliorer la régularité grâce à la mise en place d'un système d'espacement et des équipements nouveaux correspondants.

Consistance :

Equipement de la ligne en block automatique luminues moderne et redécoupage de système d'espacement des trains.

A.9 Enoncé du problème 3

DOUAI : AMENAGEMENT DE LA GARE

Finalités :

- Améliorer la fluidité du trafic fret sur le tronçon de ligne Ostricourt-Arras et Ostricourt-Valenciennes
- Améliorer la fluidité du trafic TER sur l'axe TER Lille-Arras.
- Optimiser l'exploitation de la gare de Douai en permettant les passages des trains fret par les voies A et B de l'itinéraire alternatif.
- Obtenir une souplesse d'exploitation générale plus importante de la gare de Douai.
- Améliorer la régularité des circulations.
- Augmenter la capacité de la gare de Douai en créant un itinéraire alternatif permettant de réaliser la séparation des flux TER et fret entre l'axe Paris/Lille et Douai/Valenciennes, apte à recevoir un trafic lourd à la vitesse de 60 km/h

Consistance :

L'aménagement consiste à séparer les courants voyageurs et fret en provenance et à destination de Valenciennes et Cambrai.

- Adaptation des voies (appareils de voie, traversées obliques) à la vitesse de 60 km/h en gare d'Arras.
- Création d'une bifurcation de voies pour supporter la vitesse de 60 km/h en gare de Lille.
- Remaniement de la ligne Paris-Nord à Lille.
- Réduction de la longueur des voies et la suppression d'une impasse.
- Création d'une piste de circulation pour le personnel.
- Renouvellement des deux voies pour permettre la circulation de trafic lourd à 60 km/h
- Adaptation des installations de télécommunication et caténaire.
- Adaptation du système de signalisation.
- Réchauffage électrique des aiguilles.

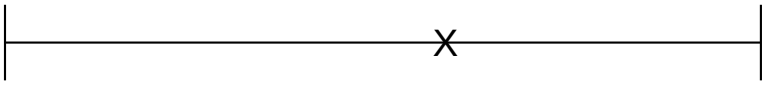
A.10 Questionnaire 5

Nom :	**Prénom :**
Emploi :	**Service :**
Age :	**Date :**

Exemple :

Aimez-vous les vacances ?

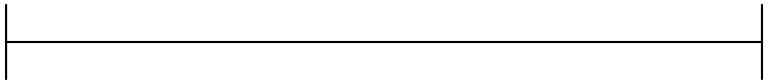
Pas du Tout — Parfaitement



Position par rapport au problème présenté

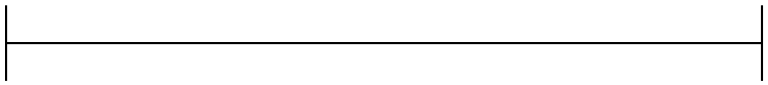
Avez-vous compris le problème ?

Pas du Tout — Parfaitement



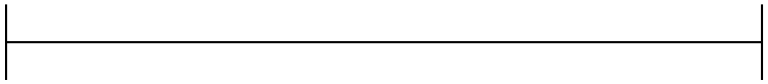
Ce problème vous semble-t-il complexe à traiter ?

Pas du Tout — Parfaitement



Vous considérez-vous comme expert pour la résolution de ce problème ?

Pas du Tout — Parfaitement



Quelles sont les études qui permettraient de vérifier le bien fondé de cet investissement ?

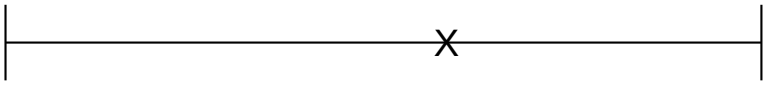
A.11 Questionnaire 6

Nom :	**Prénom :**
Emploi :	**Service :**
Age :	**Date :**

Exemple :

Aimez-vous les vacances ?

Pas du Tout Parfaitement

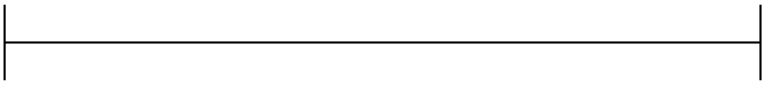


Position par rapport aux résultats des modes :

Etes-vous satisfait des outils qui ont été sélectionnés (des propositions du système s'il y en a eu) ?

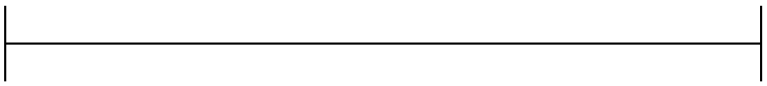
Pour le mode par Liste

Pas du Tout Parfaitement



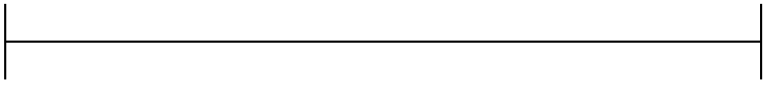
Pour le mode par Thèmes

Pas du Tout Parfaitement



Pour le mode par Mots clés

Pas du Tout Parfaitement



Utilisation de la grille de comparaison :

Que pensez vous des résultats, des différences ? (Réponse orale)

A.12 Grille de Comparaison

Nom :	**Prénom :**
Emploi :	**Service :**
Age :	**Date :**

Problème :

Outils disponibles	Choisi avant utilisation du SIAD	Choisi en utilisant le mode Liste	Choisi en utilisant le mode Thèmes	Choisi en utilisant le mode Mots clés
Insertion d'un sillon standard				
Planification par cadencement				
Insertion d'un ensemble de sillons				
Mesure du temps de voie libre des signaux				
Analyse de l'occupation des voies à quai				
Etude des conflits dans les nœuds				
Analyse de l'effet d'un cisaillement dans un nœud				
Calcul des temps de parcours, temps de garage et temps de circulation				
Association de sillons				
Taux d'utilisation de l'infrastructure sur le linéaire				
Taux d'utilisation de l'infrastructure des voies à quai				
Taux d'utilisation de l'infrastructure en fonction de l'ordonnancement des sillons				
Recherche des annonces longues				
Recherche des cantons pénalisants				
Gestion des IPCS				
Simulateur de trafic avec infrastructure				
Ordonnancement des sillons				

Glossaire

Aléas dans l'execution du service : ces aléas correspondent à des retards imprévisibles, dus à des événements extérieurs (incidents voyageurs, signaux d'alarme, dérangements d'installation, pannes

Bifurcation : lieu où les circulations vont ou viennent de directions différentes. Deux catégories de bifurcations ; L'une où un aiguillage permet aux trains de se diriger vers des destinations différentes : c'est la divergence. L'autre où les trains d'origine différente empruntent un itinéraire commun, c'est la convergence.

Cadencement : le cadencement s'applique aux horaires des circulations. Ce sont par exemple au départ d'une localité le départ des trains vers une même destination à horaire régulier (un train chaque $\frac{1}{4}$ d'heure au minute 1, 16, 31, 46)

Capacité ferroviaire : nombre des sillons définis que l'on peut placer sur une infrastructure ferroviaire durant un intervalle de temps donné.

Compactage : Le compactage des sillons permet de déterminer le taux d'utilisation d'une infrastructure. Il s'opère en rapprochant d'une manière théorique les circulations d'un graphique en ne laissant que les temps minimums d'espacement entre chaque circulation.

Complexe ferroviaire : cf. Nœud complexe

Croisement : Ce terme correspond à deux voies qui se croisent en un même point. Pour qu'un train puisse emprunter ce poinr (cisailler une voie), il faut que son itinéraire soit protégé par la fermeture de signaux de protection. Durant le passage du convoi, l'artère empruntée n'admet aucune circulation.

Dédoublement de jonctions : intégration d'aiguillages nouveaux pour permettre des mouvements simultanés de circulation

Double voie : deux voies permettent de relier deux points géographiques. En général, chacune est associée à un sens de circulation.

DSS : Decision Support System

ECD : Extraction de connaissance à partir de données.

Espacement : Deux trains successifs doivent toujours être séparés par une distance permettant l'arrêt du second train sans percuter le premier. Des signaux jalonnent les voies pour gérer cet espacement.

Graphique espace/temps : permet de visualiser une grille horaire

Grille horaire : sur un graphique espace/temps, la grille horaire fournit l'ensemble des circulations prévues sur une infrastructure donnée. Chaque circulation est représentée par un sillon.

GSS : Group Support system

Hétérogénéité des sillons : indique les sillons concernant des types de circulations différentes en particulier en temps de parcours.

Insertion d'un train : un train s'insère dans une ligne au niveau d'un point remarquable.

Ligne : ensemble des infrastructures ferroviaires permettant d'aller d'une origine à une destination.

Nœud : cf. Nœud ferroviaire

Nœud complexe : cf. Nœud ferroviaire

Nœud ferroviaire : correspond à un endroit particulier du réseau où l'enchevêtrement des voies est important. Exemples : les gares, les bifurcations à plusieurs directions.

Ordonnancement des sillons : l'ordre des sillons de type différents dans un graphique a des répercussions sur la capacité de la ligne.

Performances du matériel : ces performances correspondent le plus souvent aux performances d'accélération et de freinage. Le freinage en particulier est un freinage de confort, il ne doit pas provoquer de désagréments aux personnes et aux marchandises.

Perturbations : cf. Aléas dans l'éxecution du service

Plan de voies : représentation schématique des voies et des quais d'une gare. Sur ce plan de voie, on trouve la position des aiguillages et des signaux avec leur identification et leur point kilométrique.

Point kilométrique : chaque ligne est bornée. Les éléments de la ligne sont repérables par leur point kilométrique

Point remarquable : point facilement identifiable par les conducteurs de train. Ils servent au jalonnement des horaires.

Régularité : conception d'une grille horaire qui permet aux circulations d'être à l'heure. Souvent associée à la robustesse de la grille.

RFF : Réseau Ferré de France

Robustesse d'une grille : En cas de petits aléas sur une circulation, la robustesse permet d'éviter que les autres circulations soient gênées.

SAD : Système d'Aide à la Décision

SBC : Système à Base de Connaissance

Section de ligne : une partie de ligne où il y a des modifications du nombre de circulations ou de leur ordonnacement.

SIAD : Système Interactif d'Aide à la Décision

Signaux d'espacement : signaux utilisés pour assurer l'espacement entre deux trains successifs. Les signaux d'espacement sont utilisés pour éviter le rattrapage d'une circulation sur celle qui la précéde.

Signaux de protection : Signaux d'arrêt absolu présentés pour garantir les risques de prise en écharpe, d'affrontement, ou pour la protection des obstacles et du personnel.

Sillon : espace temporel réservé pour la circulation d'un train d'une nature donné (TER, FRET, GL) sur un itinéraire donné. Le sillon est sur l'infrastructure à la circulation pour se rendre d'un point à un autre.

SNCF : Société Nationale des Chemins de Fer

Taux d'utilisation de l'infrastructure : ce taux représente le pourcentage d'utilisation d'une infrastructure. Il se détermine par la méthode de compactage.

Temps de retour à voie libre : Le temps mis par un signal après le passage d'une circulation pour présenter à nouveau la voie libre. Les principaux paramètres qui déterminent ce temps sont la vitesse la longueur du train, la distance entre les signaux d'espacement etc.)

Tronçon : partie d'une ligne ne comprenant pas d'insertion ni d'échappement de train.

Voie : ensemble comprenant les rails, les traverses, le ballast. L'infrastructure comprend les voies, la caténaire, les installations de signalisation.

Voie libre : état d'un signal autorisant le passage d'une circulation sans aucune contrainte (Feu vert).

Voie unique : Il n'y a qu'une voie pour les deux sens de circulations sur une ligne.

Résumé

Les Systèmes Interactifs d'Aide à la Décision (SIAD) adaptés aux prises de décisions basées sur les connaissances sont au coeur de ces travaux. Un ensemble d'approches de développement (modèles, méthodes et architectures) provenant du génie logiciel, des IHM, des systèmes de connaissance, des systèmes à base de composants et des SIAD n'ont pas semblées correspondre au développement de ces SIAD. Une Approche de DEveloppement de Système Interactif d'Aide à la Décision (ADESIAD) a été proposée. Elle a pour particularité d'être centrée sur le décideur et d'utiliser des patrons pour formaliser et structurer la connaissance et définir les besoins en composants métier. SIADIF (Système Interactif d'Aide à la Décision en Investissement dans une infrastructure Ferroviaire) a été développé suivant ADESIAD pour Réseau Ferré de France. Les premières évaluations de SIADIF ont permis de valider globalement ADESIAD. Les perspectives de recherche visant à faire évoluer ADESIAD terminent cette thèse.

Mots-clés : Approche de développement, Système Interactif d'Aide à la Décision (SIAD), Interaction Homme-Machine (IHM), Système de connaissance, Système à base de composants

Abstract

This thesis presents the problematic of the choice between investment projects and focus to the need of Decision Support Systems (DSS) to help the railway authorities. We considere the case where the decision-making is mainly knowledge-based. In order to capitalize and reuse this knowledge, the component-based development (CBD) was studied. The approaches found in various domains such as software engineering, HCI, DSS, knowledge-based systems and CBD were investigated to match the development requirements. So, our work aims at the proposal of a new approach which includes a model, a method and an architecture as well. The approach aims at the decision-maker centered development and using patterns. It was applied to develop a DSS in the railway infrastructure investment domain. The first evaluation allowed to globally validate the approach and the end users agreed with the developed system. The perspectives of this work aims at extending the approach to other kinds of DSS.

Keywords: Development Approach, Decision Support System (DSS), Human-Computer Interaction (HCI), Knowledge System, Component-based System