HAL
open science

Ingénierie des Besoins : L'Approche L'Ecritoire

Colette Rolland

To cite this version:

Colette Rolland. Ingénierie des Besoins : L'Approche L'Ecritoire. Les Techniques de l'Ingenieur, 2003, pp.1. hal-00706492

HAL Id: hal-00706492

https://hal.science/hal-00706492v1

Submitted on 16 Jun 2012

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.

L'INGENIERIE DES BESOINS : L'APPROCHE L'ECRITOIRE

Colette Rolland

CRI Université de Paris 1 - Sorbonne
90 rue de Tolbiac
75013 Paris - France
rolland@univ-paris1.fr

Cet article adresse les problèmes posés par l'Ingénierie des Besoins (IB) et propose une approche, *L'Ecritoire*, pour conduire le processus d'IB. L'article est organisé en cinq chapitres. Le chapitre 1 définit l'IB et présente l'état de l'art du domaine. Le chapitre 2 est un survol de l'approche *L'Ecritoire*. Le chapitre 3 présente le modèle des besoins de l'approche tandis que le chapitre 4 décrit le processus d'IB et l'illustre par un exemple. On conclue dans le chapitre 5.

1. Introduction et Etat de l'Art

L'Ingénierie des Besoins (IB) est concernée par l'identification de buts assignés au système envisagé et l'opérationalisation de ces buts en contraintes et exigences imposées au système et aux agents qui en assureront le fonctionnement. L'IB peut être vue comme le processus qui permet de transformer une idée floue en spécification précise des besoins servant de support à la spécification du système et de ses interfaces avec l'environnement. Le processus d'IB inclue donc des activités de découverte, de spécification, de négociation et de validation des besoins. Au cours de l'IB, les besoins à l'égard du système sont découverts, négociés, validés, spécifiés et répertoriés dans des documents de spécification des besoins. Produire un document des besoins de qualité est difficile et crucial. De nombreuses enquêtes confirment en effet l'importance de l'IB dans la réussite des projets de systèmes d'information.

Ce chapitre est organisé en trois sections. Les motivations pour un processus d'ingénierie des besoins systématique et formalisé sont présentées d'abord (1.1). Ensuite, l'ingénierie des besoins est définie (1.2) et les approches émergentes pour l'ingénierie des besoins sont introduites (1.3).

1.1 Motivations pour l'ingénierie des besoins

Enquêtes et chiffres

La crise du logiciel est récurrente. Depuis 25 ans on ne cesse d'observer que le développement des systèmes logiciels est problématique. En 1976 déjà, Bell et Tayer ont fait observer que l'inadéquation des fonctionnalités du système aux besoins des usagers, l'incohérence, l'incomplétude et l'ambiguïté des documents de besoins avaient une influence majeure sur la qualité du logiciel final [Bell76]. Ces auteurs concluent que « the requirements for a system do not arise naturally, instead they need to be engineered and have continuing review and revision ». Barry Boehm estime quant à lui, que la correction d'erreurs coûte 200 fois plus cher en phase de tests que pendant la phase préalable d'IB [Boehm81]. Dans son papier réputé, Brooks énonce « the hardest single part of building a software system is deciding precisely what to build… Therefore, the most important function that the system builder performs for the client is the iterative extraction and refinement of the product requirements » [Brooks87]. Dans son étude des erreurs du programme Voyager & Galileo de la NASA, Lutz [Lutz93] rapporte que la principale cause vient des erreurs d'expression de besoins fonctionnels et d'interface.

Des enquêtes plus récentes ont confirmé l'ampleur des désastres dus à l'insuffisante qualité des documents de besoins. Une enquête menée auprès de 800 projets conduits dans 350 compagnies américaines par le Standish Group [Standish95] et présentée dans deux rapports, intitulés *'Chaos'* et *'Unfinished Voyages'*, a révélé que 31% des projets sont annulés avant même d'être terminés. En 1995, cela a coûté 81 milliards de dollars aux compagnies américaines. Ce même rapport montre que 50% d'entre eux n'avaient que partiellement réussi dans le sens où ils avaient nécessité des budgets et des délais très fortement majorés. La mauvaise qualité des documents de besoins constitue 47% des causes d'échecs citées. Comme l'indique la Figure 1.1, ce pourcentage est distribué de la façon suivante : manque de participation des utilisateurs (13%), besoins mal exprimés (ou incomplets) (12%), besoins changés entre le début et la fin du projet (11%), besoins qui manquent de réalisme (6%), et objectifs peu clairs (5%).

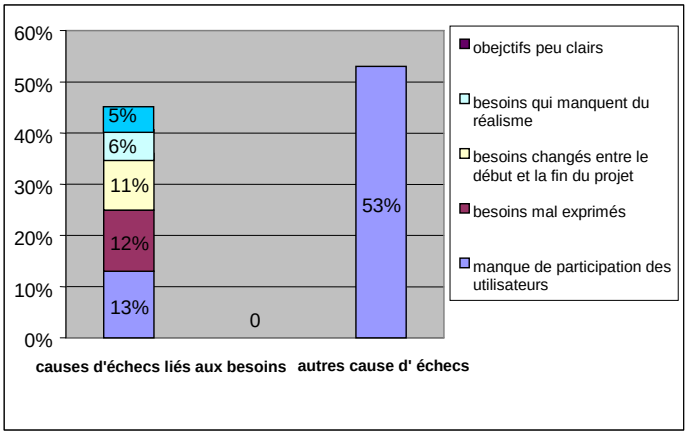


Figure 1.1 : Distribution des causes d'échecs selon l'étude du Standish Group

De plus, parmi les projets menés à terme, près de 67% des coûts de maintenance résultent de l'incomplétude des documents des besoins : « *La plupart des efforts de maintenance consistent à fournir des fonctionnalisés nécessaires mais manquantes* » [McGraw97].

En Europe une enquête de grande envergure auprès de 3800 organisations dans 17 pays différents [ESI96] a conclu dans le même sens : les principaux problèmes sont liés à la spécification des besoins (>50% des réponses) et à la documentation des besoins (50%).

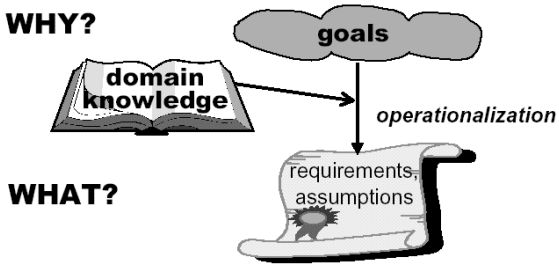
Conséquences
On comprend à travers ces faits et chiffres, l'importance de l'ingénierie des besoins et son caractère crucial pour la production de systèmes de qualité. Pour remédier à cette situation d'insuccès il semble nécessaire de s'interroger sur la mission de l'ingénierie des besoins mais peut être aussi, de comprendre pourquoi les approches d'IB pratiquées traditionnellement échouent dans cette mission. C'est ce que nous tentons de faire dans la section suivante.

1.2 Mission et objectifs de l'ingénierie des besoins

De l'importance du POURQUOI
La plus ancienne définition de l'IB contient les « ingrédients » essentiels. Dans leur article prémonitoire, Ross et Schoman écrivent « requirements definition is a careful assessment of the needs that a system is to fulfil. It must say why a system is needed, based on current and foreseen conditions, which may be internal operations or an external forces. It must say what system features will serve and satisfy this context. And it must say how the system is to be constructed » [Ross77].

En d'autres termes, l'ingénierie des besoins doit de prime abord s'intéresser aux buts du contexte organisationnel du système à développer parce qu'ils permettent de comprendre les raisons justifiant son développement. L'IB doit poser la question POURQUOI développer un système et aider les parties prenantes du projet à y répondre. Le rôle de l 'IB est ensuite de déterminer les fonctionnalités que le système doit mettre en œuvre pour aider à la satisfaction de ces buts et identifier les contraintes qui restreignent la mise en œuvre de ces fonctions. Ces buts, fonctions et contraintes, constituent les '*besoins*' qui doivent ultérieurement être convertis en une spécification précise permettant le développement du système. L'évolution des besoins au cours du temps et la répercussion de leur évolution sur la spécification du système doit aussi être prise en compte [Zave97].

La Figure 1.2 empruntée à Axel van Lamsweerde [Lamsweerde00] schématise cette vue de l'ingénierie des besoins centrée sur les questions POURQUOI (WHY) et QUOI (WHAT) ainsi que la mise en correspondance des réponses inhérentes à l'une et à l'autre.



(from Axel van Lamsweerde)

Figure 1.2 : Le fondement de l'ingénierie des besoins

Se centrer sur le POURQUOI doit permettre de découvrir les besoins correspondants à la mission et aux objectifs de l'organisation. Si l'on veut éviter de développer des systèmes techniquement parfaits mais inutilisés parce qu'inadaptés aux besoins réels de leurs utilisateurs, il faut se donner les moyens de comprendre à quoi le système va servir dans son contexte organisationnel. C'est la mission de l'IB. Mais ce n'est pas la pratique courante comme nous tentons de le montrer dans la section suivante.

De la nécessité de revisiter les approches traditionnelles

Dans les méthodes héritées des années 80 telles que MERISE, l'ingénierie des besoins est partie intégrante de l'étape 'd'analyse' aboutissant à la construction d'une représentation abstraite des données, des traitements et des interfaces en suivant une approche de modélisation conceptuelle. L'objectif principal est de décrire ce que le système doit faire, c'est à dire ses fonctionnalités, dans un *schéma conceptuel*. Comme le montre la Figure 1.3, l'IB fondée sur la modélisation conceptuelle est centrée sur la question QUOI à laquelle on répond par le cycle <acquisition, abstraction, validation>. La tâche d'acquisition de connaissance de domaine et des besoins s'appuie sur un cahier des charges et des interviews. Elle permet d'abstraire de la connaissance acquise, la spécification des fonctionnalités attendues du système. Le schéma conceptuel résultant sert de support à la validation des besoins par des techniques telles que le maquettage et le prototypage.

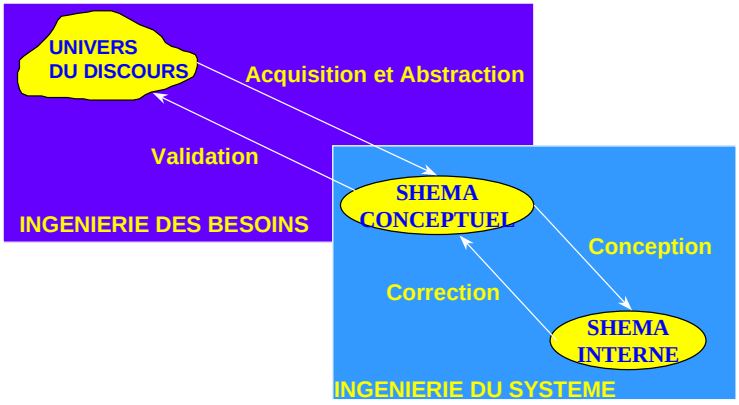


Figure 1.3 : Le cycle <acquisition, abstraction, validation>

L'IB fondée sur la modélisation conceptuelle n'est pas réellement conforme à la définition que nous en avons donnée. On peut comprendre qu'un centrage exclusif sur le QUOI inhibe les vraies questions du POURQUOI et aboutisse à la production de la spécification conceptuelle d'un système inadapté aux réelles attentes de ses usagers. Par ailleurs, le schéma conceptuel est l'expression d'une solution fonctionnelle et non des besoins à l'égard du système. L'absence d'une expression des besoins eux-mêmes ne peut que rendre difficile la validation par les parties concernées. En outre la pratique du cycle précédent repose sur des hypothèses (le plus souvent implicites) qui ne semblent plus du tout valides aujourd'hui :

- Les fonctionnalités d'un système sont stables, elles n'évoluent que peu dans le temps. En conséquence, le schéma conceptuel d'un système est lui aussi stable.
- Les besoins relatifs à un système sont donnés au départ. Les utilisateurs expriment leurs besoins dans un cahier des charges, le problème est donc de construire le système qui répond à ce cahier des charges ; ce sont les analystes (la maîtrise d'œuvre) qui sont en charge de ce travail.
- La validation des besoins peut se faire en référence aux fonctionnalités du système. En d'autres termes, le schéma conceptuel est le support privilégié pour communiquer, négocier et aboutir au consensus nécessaire entre les différentes parties impliquées dans le développement du système.

Si dans les années 80/90, ces hypothèses étaient valides, elles ne le sont plus aujourd'hui. En réponse aux pressions économiques et à l'émergence constante de nouvelles technologies, les organisations changent plus rapidement que par le passé. En conséquence, ce que les utilisateurs attendent du système évolue bien plus rapidement qu'auparavant. Leurs besoins ne sont donc pas stables [Harker93]. On sait même que les besoins évoluent au cours du projet [Lubars93] et qu'il est donc nécessaire de se doter de moyens permettant d'établir un lien conceptuel entre les buts et objectifs de l'organisation (réponse au POURQUOI), les besoins qui en résultent (réponse au QUOI) et les spécifications fonctionnelles du système qui les supportent .

On sait que le rôle central de l'analyste système a été reconsidéré et que la maîtrise d'ouvrage vient contrebalancer le rôle de la maîtrise d'œuvre. En fait, il est clair aujourd'hui que l'ingénierie des besoins requiert la participation d'un grand nombre d'acteurs de l'organisation, chacun apportant sa vision sur ce que le système devrait faire [Finkelstein90]. On distingue par exemple, les utilisateurs finaux du système – ceux qui utiliseront le système pour mener à bien leurs activités au sein de l'organisation –, les responsables de l'organisation qui ont décidé de la mise en place du système, les personnes responsables de la mise en place et de la maintenance du système, etc. ; en fait tous ceux pour qui le développement du système constitue un enjeu (les « stakeholders » dans la terminologie anglo-saxonne).

Enfin, la validation des besoins sur la base du schéma conceptuel n'est pas satisfaisante. L'expérience l'a montré. On peut accepter l'idée que cette validation soit bien plus efficiente si elle est basée sur l'expression des besoins eux-mêmes et non sur la spécification abstraite et difficile à comprendre des spécifications fonctionnelles du système qui en résultent. Ceci justifie l'introduction d'un document des besoins dans lequel les besoins spécifiés peuvent être discutés, négociés et validés.

Pour s'attaquer à ces problèmes il est essentiel de comprendre que l'ingénierie des besoins

(a) n'a pas vocation à se centrer sur le *QUOI* mais sur le *POURQUOI* pour en dériver un *QUOI* motivé,
(b) doit décrire le *QUOI* en termes de besoins que le système doit satisfaire et non de spécification des fonctionnalités du système.

La Figure 1.1 schématise cette vue de l'Ingénierie des Besoins. L'IB cherche à répondre à la question POURQUOI et à associer un QUOI compatible et motivé par le POURQUOI. Elle fait appel à des modèles aussi bien pour exprimer la dimension POURQUOI que pour la description du QUOI et préconise d'établir et de tracer le lien entre les éléments du premier modèle et ceux du second. Ce lien doit faciliter la répercussion d'un changement organisationnel sur les besoins.

A propos du processus d'ingénierie des besoins

Notre définition de l'IB laisse deviner pourquoi le processus d'ingénierie des besoins est si complexe et encore mal maîtrisé.

- Le champ est large puisqu'il s'étend du domaine social des organisations à celui des lois de la physique aux artefacts qui doivent y être intégrés; des objectifs stratégiques aux prescriptions logicielles fines; de l'informel au formel. Le système cible n'est pas seulement un logiciel mais comprend l'environnement organisationnel dans lequel il doit opérer. Celui-ci est complexe car il est constitué d'êtres humains, de machines et/ou de logiciels. Le système doit être considéré selon de multiples points de vue: socio-économiques, physiques, opérationnels, évolutifs etc.
- Il y a d'autres aspects à prendre en compte en plus des aspects fonctionnels du système qui viennent à en premier lieu à l'esprit : les aspects liés à la sécurité, la performance, la robustesse, la facilité d'utilisation, l'interopérabilité, etc. Tous ces aspects sont appelés *non fonctionnels* et ont la particularité d'être souvent en conflit les uns avec les autres.
- Il y a de nombreuses parties prenantes comme nous l'avons déjà mentionné. Chacune a son champ de compétence et de connaissance, sa propre culture, ses propres points de vue et intérêts. Citons, les ingénieurs des besoins, les clients, les experts du domaine, les utilisateurs, les commanditaires, les ingénieurs de maintenance et les ingénieurs d'application. Ces différentes parties prenantes ont des points de vue divergents, voire conflictuels.
- Les spécifications des besoins peuvent être entachées de nombreuses erreurs. Certaines d'entre elles peuvent être terriblement pénalisantes et avoir des effets désastreux sur les étapes suivantes du développement et/ou sur la qualité du produit final. Ce sont par exemple les incomplétudes et incohérences, ambiguïtés, inadaptations aux besoins réels; d'autres sont des faiblesses de la spécification qui peuvent aboutir à des pertes de temps ou la génération de nouvelles erreurs: ce sont par exemple, les sur-spécifications, les bruits, les références en arrière ou des besoins irréalistes.

- Le processus couvre de multiples activités qui sont inter-reliées. La Figure 1.4 en propose quatre :
  - *l'extraction* ou *découverte* des besoins,
  - *la spécification* des besoins dans une forme précise et non ambiguë,
  - *la validation* des besoins et
  - *la négociation* conduisant au choix des besoins qui seront implémentés.

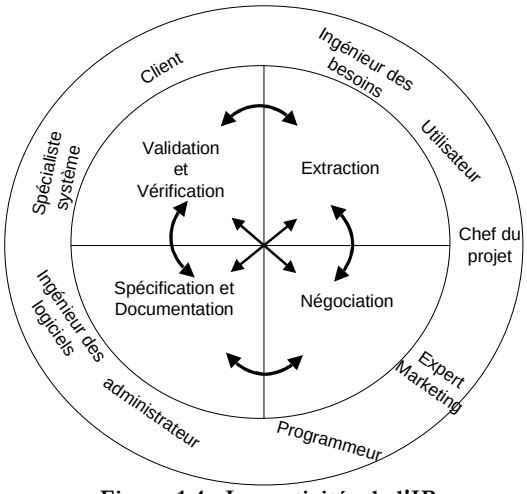


Figure 1.4 : Les activités de l'IB

A ces quatre activités principales d'un cycle d'IB s'ajoute l'activité *d'évolution* des besoins permettant de corriger des erreurs et/ou de prendre en compte de nouveaux objectifs.

Pour faire face à la complexité des activités du processus d'ingénierie des besoins, il est nécessaire de disposer de techniques, approches et outils appropriés. La section suivante présente l'état de l'art en la matière.

1.3 Approches d'ingénierie des besoins : état de l'art

On propose un cadre de référence aidant à comprendre quelles sont les sources des besoins et la nature des besoins qu'elles permettent de découvrir. Ce cadre permet d'identifier des classes ou familles d'approches d 'IB dont on décrit ensuite l'état de leur développement.

Cadre de référence

Comme le montre la Figure 1.5, cet article suggère de séparer en deux parties ce que l'on appelle traditionnellement *l'environnement* du système : le *monde du sujet* et le *monde de l'usage* [Jarke93]. L'une et l'autre sont sources de besoins mais de natures différentes.

Le monde de l'usage décrit l'ensemble des tâches, procédures, interactions etc. accomplies par les agents ainsi que la façon dont les systèmes d'information sont utilisés dans ces tâches. Ce monde doit être considéré comme celui où les objectifs de l'organisation doivent être atteints au travers de l'exécution des tâches réalisées par les agents. Le monde de l'usage décrit représente (1) les activités des agents et (2) de quelle manière les activités génèrent de la valeur ajoutée.

Le monde du sujet contient les connaissances du domaine pour lequel le futur système doit apporter de l'information. Ce monde contient les objets réels qui doivent être représentés dans le système.

C'est évidemment dans le monde de l'usage que l'on peut identifier les intentions, souhaits et buts des usagers du futur système. Au contraire, le monde du sujet génère des exigences qui reflètent les lois du domaine telles que les lois de la physique et ceci, indépendamment des ambitions, souhaits et besoins des usagers.

Il y a un troisième monde, le *monde du système* qui est celui des spécifications du système et dans lequel les besoins issus des deux autres mondes doivent être formulés et documentés. Ce monde détient aussi les représentations des entités, événements, processus des mondes du sujet et de l'usage ainsi que leur transformation en spécifications techniques et implémentations logicielles.

Tous ces mondes sont en interaction comme le montre la Figure 1.5. Les besoins issus du monde de l'usage sont capturés par la *Relation intentionnelle* et la relation *d'Adéquation à l'usage*. Les besoins imposés par les lois du domaine sont capturés par la relation *d'Adéquation au domaine*.

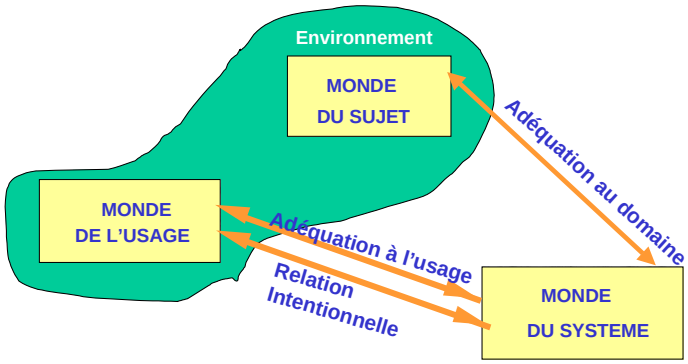


Figure 1.5 : Les trois mondes et leurs interactions

Le monde de l'usage est constitué des individus, des groupes sociaux et des cadres organisationnels dans lesquels le futur système va fonctionner [Gougen93]. C'est donc en explorant ce monde que l'on a une chance de répondre à la question du POURQUOI ce système plutôt qu'un autre. Toutefois on peut identifier entre le monde de l'usage et le monde du système différents types de relations qui sont sources de différents types de besoins. Les relations individuelles, *pragmatiques*, entre le monde de l'usage et le monde du système sont associées à la relation *Adéquation à l'usage*, tandis que les relations sociales, *sémiotiques* sont fournies par la *Relation intentionnelle*.

Les objectifs qui façonnent le développement d'un système se trouvent dans la tête des acteurs du contexte de fonctionnement du système. C'est donc en explorant la *relation intentionnelle* entre le monde de l'usage et le monde du système que l'on peut établir le lien entre les intentions des acteurs, les objectifs assignés au système qui en découlent et les fonctionnalités qui doivent être les siennes pour répondre à ces objectifs. Modéliser ce lien permet d'établir la connexion entre le futur système et l'environnent dans lequel il va fonctionner. Cela peut aussi permettre d'expliquer pourquoi un système d'une certaine configuration est utile. Les *approches dirigées par les buts* ont été développées dans cette perspective.

Le monde de l'usage est aussi celui des utilisateurs finaux du système, ceux qui vont utiliser individuellement le système pour réaliser les tâches qui leur permettent d'atteindre les objectifs que l'organisation leur a assignés. Chacun des ces utilisateurs a son propre point de vue sur ce qu'il attend du système, sur ses exigences fonctionnelles et non fonctionnelles à l'égard du système. Prendre en considération ces différents points de vue devrait permettre de construire un système plus proche des attentes de ses usagers. Les *approches à base de scénarios* ont privilégié cette perspective.

Approches dirigées par les buts

Les approches conventionnelles centrées sur le QUOI formalisent les contraintes imposées aux opérations, et objets du système mais ne disent pas POURQUOI ces contraintes sont imposées et si elles sont suffisantes pour garantir que les objectifs que l'organisation place dans le système seront satisfaits [Munford81, Berzins91, Rubin92]. Yue a sans doute été le premier à argumenter en faveur de l'introduction de modèles de buts dans le document de spécification des besoins [Yue87] afin d'introduire un critère de complétude des besoins : une spécification de besoins est complète si elle permet de satisfaire le but qu'elle affine [Yue87].

D'une façon générale un *but* se définit comme un objectif que le système futur doit garantir par la coopération appropriée d'agents avec le système. Un but est donc une propriété 'envisagée' (*optative* selon Jackson [Jackson95] par opposition à 'descriptive' de l'état courant).

Les buts sont formulés à différents *niveaux d'abstraction*, allant de buts stratégiques de haut niveau (tels que '*Servir davantage de passagers*' pour un service de transport de passagers par trains ou '*Fournir un service de retrait d'argent*' pour un réseau de Guichets Automatiques Bancaires (GAB)) à des buts techniques de bas niveau (tels que '*Activer la commande d'accélération à temps*' pour le système de transport ou '*Absorber la carte de crédit après trois tentatives de saisie de code incorrectes*' pour le GAB). Les modèles de buts contiennent donc la décomposition d'un but en sous buts dans des graphes de réduction ET/OU inspirés de

l'Intelligence Artificielle [Nilsson71]. Une réduction ET associe un but B à un ensemble de sous buts B1,B2…Bn qui doivent tous être satisfaits pour que B le soit. Une réduction OU associe un but B à un ensemble B1,B2,..Bn de sous buts tels que la satisfaction de l'un d'entre eux assure la satisfaction de B. De nombreuses approches dirigées par les objectifs incluent des graphes ET/OU avec certaines variations d'une approche à l'autre [Dardenne93, Bubenko94, Rolland98a, Loucopoulos94, Loucopoulos97, Mylopoulos99]. Ce mécanisme de réduction est fondamental pour assurer le passage du POURQUOI (les buts stratégiques de haut niveau) au QUOI (les buts de bas niveau qui s'opérationnalisent en objets et opérations du SI).

Les graphes ET/OU précédents comportent des buts que l'on dit *'durs'* (hard goals) par opposition aux buts *'mous'* (soft goals) introduits par Mylopoulos [Mylopoulos92]. Les premiers sont des buts faisant référence à des fonctions du système ; ils sont *fonctionnels* alors que les seconds font référence à des qualités attendues des fonctions; ce sont les buts *non fonctionnels*. De plus, Mylopoulos a assoupli la notion de *satisfaction* d'un but considérée dans les graphes ET/OU par celle de *satisfaisabilité*. Dans une relation de satisfaisabilité, un sous but contribue à la satisfaction du but père dans une certaine limite plutôt que de manière absolue. Un tel sous but contribue positivement ou négativement à la satisfaction du père indépendamment des autres sous buts. L'approche comprend des règles de propagation de la satisfaisabilité des buts dans un graphe ET/OU. Les buts non fonctionnels sont utiles pour raisonner sur les besoins qualitatifs imposés au système. Par exemple, Nixon [Nixon93] a montré que les buts non fonctionnels permettent de raisonner sur les performances du système et Anton [Anton01] sur la confidentialité. De même dans [Leveson95] et [Amoroso94] les buts non fonctionnels sont utilisés pour raisonner sur les exigences de sécurité. Le livre récent de Chung et al [Chung00] est une bible sur les buts non fonctionnels et les raisonnements qu'ils supportent.

On comprend que les buts dirigent la découverte des besoins qui permettent de les satisfaire. De nombreuses approches ont montré que les buts et les scénarios sont aujourd'hui les deux instruments clés du processus d'extraction des besoins [Potts94, Rubin92, Dardenne93, Anton94, Rolland98a, Dubois98, Kaindl00, Lamsweerde00].

Outre leur rôle essentiel pour la découverte des besoins fonctionnels et non fonctionnels, les buts ont montré leur utilité et leur importance dans nombre d'activités de l'IB :

- *Vérifier la complétude de la spécification des besoins* : l'ensemble des buts fournit un critère de mesure de la complétude d'une spécification de besoins; celle-ci est complète si l'on montre que l'ensemble des buts peut être satisfait par l'ensemble des besoins de la spécification [Yue87].
- *Expliquer les besoins aux parties prenantes du processus d'IB* : les buts notamment de haut niveau fournissent les raisons du développement du système. Un besoin n'existe que s'il y a un but qui justifie sa présence dans la spécification [Ross77, Mostov85, Dardenne93, Bubenko94, Sommerville97, Yu94].
- *Explorer des alternatives de conception* : le système à développer peut fonctionner et interagir avec son environnement de multiples façons ; le mécanisme de réduction des buts et le graphe ET/OU qui en résulte aide les ingénieurs d 'IB à expliciter de nombreuses alternatives et à raisonner pour déterminer celle qui semble la mieux appropriée à la situation du projet [Yu94, Rolland99, Lamsweerde00].
- *Structurer la collection des besoins* : le mécanisme de réduction des buts y contribue.
- *Etablir les liens de traçabilité* : le graphe de réduction des buts est un moyen d'établir les liens qualifiés de pré-traçabilité [Gotel94, Pohl96, Ramesh95]. On sait combien ces liens qui assurent la relation entre les objectifs organisationnels et les besoins à l'égard du système sont utiles pour propager un changement des premiers dans les seconds.
- *Identifier et gérer les conflits* : les différents acteurs du processus d'IB apportent des points de vue intéressants sur le système à développer ; on sait que ces points de vue peuvent être divergents et générer des conflits[Nuseibeh94]. Il semble que les buts aident l'explicitation des conflits et à leur résolution[Robinson89, Easterbrook94, Lamsweerde00]

On peut se référer à [Green93] pour une étude comparative des approches *'dirigées par les buts'*.

Approches à base de scénarios

Les *approches à base de scénarios* font l'hypothèse qu'il est plus facile de décrire, le plus souvent dans un récit en langue naturelle, ce qui se passera dans le système futur que d'exprimer directement les objectifs du futur système. Les acteurs de l'organisation ont parfois des difficultés à énoncer à priori les objectifs attendus du système ; ceux-ci ne sont facilement explicités qu'après avoir bien compris ce que doit faire le système. La

difficulté à manipuler des notions abstraites comme les buts a été reconnue par des études en sciences cognitives [Benner93]. Un *scénario* est une séquence d'interactions entre le système envisagé et son environnement énoncée dans le contexte restreint d'un propos particulier. Il décrit une 'histoire', c'est un écrit concret de la façon dont un acteur imagine d'interagir avec le système.

Une étude récente [Weidenhaupt98] montre que les scénarios sont des artefacts importants utilisés pour satisfaire de nombreux *propos*, avec différents *contenus* exprimés à différents *niveaux d'abstraction* et utilisant différentes *notations* [Rolland98b]. Chacun de ces aspects est développé dans ce qui suit.

Un scénario a un *propos* qui est descriptif, explicatif ou exploratoire. Les scénarios descriptifs [Potts94] décrivent des séquences d'événements envisagés alors que les scénarios explicatifs [Wright92] se focalisent sur les raisons d'un événement. Les scénarios exploratoires [Holbrook90] permettent de raisonner sur les alternatives de conception.

Les scénarios ont différents *contenus*. Ils peuvent décrire :

- des activités, actions ou événements actuels ou futurs,
- des objets mis en jeu par le système,
- le cadre de travail actuel ou imaginé des acteurs,
- des contraintes de qualité attendues du système,
- des informations sur l'organisation tels que sa structure, ses départements, ses groupes, ses agents etc.,
- les intervenants dans le processus de l'IB, leurs caractéristiques, leurs vues et leurs aspirations [Nardi92].

Cependant, la majeure partie des approches actuelles se focalise sur la description des fonctionnalités du futur système [Firesmith94, Glinz95, Rolland97, Potts94, Rubin92, Rawsthorme96, Some96] et vise donc à la découverte des besoins fonctionnels.

L'étude des approches du marché [Rolland98b] montre que les scénarios sont exprimés à trois *niveaux d'abstraction* différents : niveau instance, niveau type ou mixte. Au niveau instance [Caroll95, Potts94, Young87], un scénario utilise des noms réels (le client « Dupont ») ou des descriptions d'événements avec des valeurs de paramètres réels (arrivée de la commande 812, le 20/09/99 à 17h33). Les scénarios exprimés au niveau type [Hsia94, Jacobson92] n'utilisent pas d'entités du monde réel mais des types d'entités. Ils ne font pas référence à Dupont mais à la notion de client. L'exécution d'un scénario exprimé au niveau type est un scénario du niveau instance. Un scénario mixte comporte des parties au niveau type et d'autres au niveau instance.

Les scénarios sont exprimés dans différentes *notations* : informelles, semi-formelles ou formelles. Les scénarios informels sont décrits à l'aide du langage naturel [Rolland97, Erickson95, Holbrook90], de vidéos [Wood94, Haumer98], etc. Ils sont utilisés lorsque les utilisateurs rejettent les notations formelles ou semi formelles [Weidenhaupt98]. Les scénarios semi-formels utilisent des notations structurées comme des tableaux [Potts94] ou des scripts [Rubin92]. Pour finir, les scénarios formels sont décrits à l'aide de langages basés sur des grammaires régulières [Glinz95] ou des diagrammes d'états [Harel87]. Ils peuvent être utilisés pour simuler le fonctionnement futur du système et juger des réactions des utilisateurs.

On trouvera dans [Anton98] et [BenAchour99] des comparaisons de différentes *approches à base de scénarios.* Notons pour terminer que de nombreux auteurs suggèrent aujourd'hui de combiner *buts* et *scénarios* dans le processus d'ingénierie des besoins [Potts95, Cockburn95, Leite97, Kaindl00, Sutcliffe98, Haumer98, Anton98, Lamsweerde98]. Potts dit par exemple qu'il est « unwise to apply goal based requirements methods in isolation » [Potts95]. C'est aussi la position prise dans l'approche L'Ecritoire présentée dans la suite de cet article.

2. Survol de l'Approche L'Ecritoire

Cette section présente l'approche L'Ecritoire pour découvrir (on dit aussi extraire ou élucider) les besoins à l'égard d'un système souhaité et les documenter. L'accent est mis sur l'introduction aux quatre caractéristiques clés de l'approche et sur les justifications du choix de ces caractéristiques. Cette introduction permet de comprendre le processus de découverte des besoins et le produit qui en résulte.

2.1 Motivations et justifications des choix conduisant à l'approche L'Ecritoire

Comme l'état de l'art l'a mis en évidence, la plupart des approches récentes et émergentes d'IB font appel à l'une , l'autre ou les deux notions de *but* et de *scénario*. L'approche L'Ecritoire retient les deux notions et se centre sur le couple <*but, scénario*> appelé *fragment de besoin*. Les caractéristiques de l'approche sont justifiées par l'analyse critique des approches d'IB utilisant un seul des deux concepts que nous présentons ci-dessous.

Les approches dirigées par les buts ont fait la preuve de leur utilité en ingénierie des besoins en offrant à la fois

(a) un concept (le but) qui permet de formuler les besoins à différents niveaux d'abstraction et
(b) un mécanisme de décomposition permettant de réduire un but stratégique de haut niveau en une contrainte sur le système à développer.

L'expérimentation de ces approches sur le terrain a cependant montré leurs limites :

(a) le concept de but reste flou et s'il est admis que l'on démarre le développement d'un système avec un certain nombre d'objectifs en tête [Davis93], la question 'd'où viennent les buts ?' se pose [Anton96],
(b) les buts stratégiques qui initient le processus d'IB sont souvent idéalisés et irréalistes [Anton96], [Anton97, Potts97] peuvent conduire à des besoins inexacts,
(c) le processus de réduction des buts n'est pas simple à pratiquer et l'application des techniques de réduction telle que celle de KAOS [Dardenne93] n'est pas aussi directe que la littérature le laisse penser [Anton96]. Notre propre expérience dans les deux projets F3 [Bubenko94] et ELEKTRA [ELEKTRA97] nous a permis de le vérifier.

Notre position est qu'une aide doit être apportée dans trois directions : 1) pour lever les difficultés liées à la nature 'floue' des buts et faciliter le travail des experts dans le formulation des buts, 2) pour aider à la découverte des buts et 3) pour faciliter la tâche de réduction de but. L'Ecritoire vise à apporter cette aide.

Les approches à base de scénarios ont montré leur utilité pour

(a) découvrir des besoins correspondant à des situations du futur [Potts94],
(b) élucider les cas exceptionnels [Jacobson95],
(c) dériver les modèles conceptuels à partir de scénarios [Rumbaught91, Dano97] et
(d) raisonner sur les choix de conception [Caroll95].

Les approches à base de scénarios séduisent par leur (apparente ?) facilité d'application, leur caractère pragmatique et concret et leur recours à l'expression en langue naturelle. Les scénarios sont perçus comme étant plus faciles à utiliser que les buts réputés abstraits. L'enquête européenne que nous avons menée [Weindenhaup98] dans le cadre du projet CREWS [CREWS99] montre que les scénarios sont utiles en particulier là où la modélisation abstraite n'aboutit pas à des résultats satisfaisants : 90% des entreprises interrogées ayant choisi d'appliquer des approches à base de scénarios l'ont fait pour échapper aux difficultés rencontrées dans la pratique des approches conventionnelles et au rejet par les utilisateurs des techniques formelles ou semi-formelles de modélisation conceptuelle. Les scénarios aident à raisonner sur des systèmes complexes à partir d'exemples et d'illustrations. Mais leurs avantages impliquent des inconvénients :

(a) la fragmentation des besoins saisis dans de multiples scénarios rend difficile l'assurance de complétude de la spécification des besoins [Lamsweerde00],
(b) elle rend également difficile la recherche des différents aspects d'une même fonctionnalité du système au travers de multiples scénarios. Cockburn [Cockburn95] parle du "tedious problem of tracking system features accross use cases",
(c) le même auteur [Cocburn95] mentionne le caractère 'ad hoc' du processus d'identification des variations du scénario de base d'un cas d'utilisation
(d) enfin, si la pratique montre que le processus d'IB à base de scénarios est conduit de manière top-down, elle met en évidence les difficultés rencontrées dans la maîtrise des niveaux d'abstraction, un même scénario pouvant comporter des actions de différents niveaux d'abstraction et donc faisant référence à des besoins de niveaux différents.

Afin de pallier les difficultés recensées ci-dessus, nous proposons dans l'Ecritoire 1) des directives méthodologiques pour apporter des conseils d'écriture des scénarios textuels (écrits en langue naturelle) et des outils logiciels pour vérifier leur correction, 2) des règles d'analyse des scénarios aidant à la découverte des variantes, cas d'exceptions et complémentaires d'un scénario donné et 3) une formalisation du processus pour le systématiser tout en guidant son déroulement.

La proposition phare de L'Ecritoire est de *coupler buts et scénarios* pour faciliter la tâche de découverte et de documentation des besoins. Notre objectif est donc de découvrir les besoins du système futur en couplant chaque but découvert à un scénario qui illustre le comportement du système permettant d'atteindre le but. Les quatre caractéristiques suivantes de l'approche contribuent à la satisfaction de cet objectif:

1- la notion de *fragment de besoin* définie comme le couple *<but, scénario>*,
2- l'organisation hiérarchique des besoins basée sur les relations *ET*, *OU* et *Affiné par*, entre fragments de besoins,
3- un processus de découverte fondé sur un mouvement bidirectionnel entre but et scénario. Pour un but donné, un scénario est écrit pour illustrer sa réalisation. Un fois écrit, le scénario est analysé pour découvrir de nouveaux buts,
4- une aide méthodologique sous forme de *règles* semi-automatiques mises en œuvre par le *logiciel L'Ecritoire*.

On décrit ces quatre caractéristiques successivement dans la section suivante. On montre comment elles prennent en compte les améliorations suggérées pour palier (a) aux difficultés de la pratique des approches dirigées par les buts et (b) aux limites des approches à base de scénarios.

2.2 Caractéristiques de l'approche L'Ecritoire

2.2.1 La notion de fragment de besoin

La notion de *fragment de besoin* (FB) est au cœur de l'approche. Un FB est un couple <but, scénario> dans lequel le but explicite ce que souhaite l'usager tandis que le scénario décrit un comportement possible du système pour atteindre le but. Par nature, un but est 'intentionnel' tandis qu'un scénario est 'opérationnel'. Le couplage but-scénario permet donc de concrétiser le but au moyen d'un scénario.

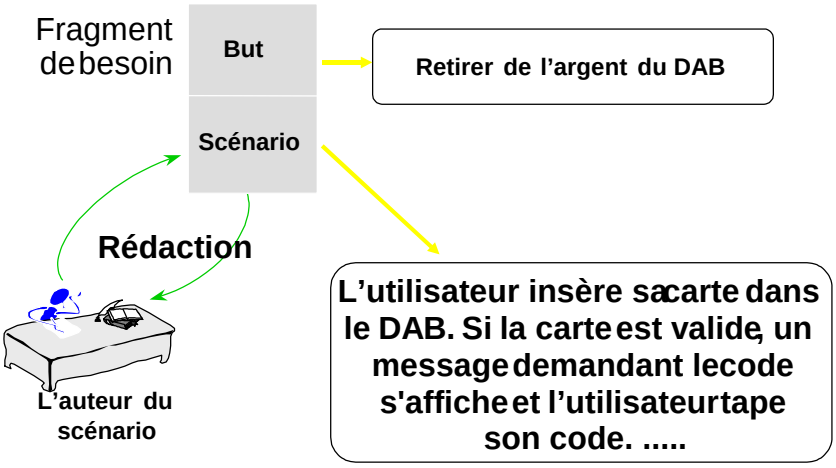


Figure 2.1 : La notion de fragment de besoin

Comme le montre la Figure 2.1, un *but* est une expression littérale qui exprime 'ce que l'usager souhaite obtenir avec l'aide du système [Lamsweerde01]. Pour lever une partie du caractère 'flou' d'un but, l'approche s'appuie sur une formalisation du but fondée sur une adaptation de l'approche linguistique de la grammaire des cas de Fillmore [Fillmore68] étendue par [Dik89]. Un but s'exprime au moyen d'un verbe à l'infinitif et d'un certain nombre de paramètres qui jouent des rôles (des cas) prédéfinis par rapport au verbe. Par exemple, l'expression de but

'$(Retirer)_{verbe}$ *de* $(l'argent)_{Obj}$ *du* $(DAB)_{So}$'

comporte le verbe '*Retirer*' et deux paramètres : '*l'argent*' qui est l'objet du retrait et le '*DAB*' qui est le moyen pour permettre le retrait d'argent. L'Ecritoire aide l'utilisateur à re-formuler l'expression intuitive d'un but en une expression conforme à la structure prédéfinie de tout but.

Un *scénario* est 'un comportement possible du système limité à un ensemble d'interactions entre un agent et le système' [CREWS98]. Comme le montre la Figure 2.1, les scénarios de l'Ecritoire sont textuels ; ils sont écrits en langage naturel. Schématiquement un scénario comporte une séquence d'actions intervenant entre l'usager et le système pour atteindre le but. On verra au chapitre suivant que l'approche autorise différents types d'actions : atomiques ou composées, conditionnelles, concurrentes ou itératives. *'L'utilisateur insère sa carte dans le DAB'*

est une action atomique ; *'si la carte est valide, un message demandant le code s'affiche'* est un flux composée de deux actions dont la première est conditionnelle. L'Ecritoire met en œuvre (a) des directives d'écriture des scénarios et (b) des outils linguistiques pour analyser, interpréter et transformer (on dit conceptualiser) un scénario pour assurer sa cohérence, sa complétude et sa conformité au modèle de scénario.

Dans un fragment de besoin <but, scénario>, le couplage d'un but à un scénario est un moyen de concrétiser un but. L'expérience [Rolland99] acquise dans des projets d'application de l'approche montre que ce couplage permet de déceler des buts irréalistes. Ceci survient lorsqu'il apparaît impossible de rédiger un scénario permettant d'atteindre le but.

2.2.2 La hiérarchie des fragments de besoin

Les fragments de besoin sont liés les uns aux autres par des liens *de composition*, *d'alternative* et *d'affinement*. Les deux premières conduisent à des relations structurelles *ET* et *OU* entre FBs tandis que la troisième aboutit a une organisation hiérarchique des FBs à travers des relations '*Affiné par'*.

Les relations ET permettent de lier des FBs qui sont complémentaires dans le sens où l'un est indispensable au fonctionnement de l'autre. C'est à travers l'ensemble des FBs associés par des ET que l'on peut s'assurer de la complétude du fonctionnement attendu du système. Par exemple à la Figure 2.2, les FBs correspondant aux deux buts '*Retirer de l'argent au moyen d'un DAB à carte'* et '*Remplir le DAB de billets de banque'* sont complémentaires. En effet, pour que les clients puissent retirer de l'argent 24h sur 24, il faut alimenter le réservoir du DAB de billets régulièrement : il faut donc satisfaire le but '*Remplir le DAB de billets de banque'* pour que le but '*Retirer de l'argent au moyen d'un DAB à carte'* soit lui-même satisfait.

Les fragments liés par des OU représentent différentes façons alternatives d'atteindre le même but. Par exemple à la Figure 2.2, les FBs correspondant aux buts '*Retirer de l'argent au moyen d'un DAB à carte'* et '*Retirer de l'argent avec émission de reçu'* sont deux façons alternatives de retirer de l'argent d'un DAB.

En faisant l'analogie avec les cas d'utilisation de Jacobson [Jacobson95] que l'on retrouve dans UML, on peut dire que l'ensemble des FBs liés par des ETs correspondent à l'ensemble des cas d'utilisation pour un système donné tandis que l'ensemble des FBs associés par des OU pour un but donné correspondent à l'ensemble des scénarios du cas d'utilisation pour ce but.

Les FBs liés par des relations '*Affiné par'* sont à différent niveaux d'abstraction. Comme le montre la Figure 2.2, le but '*Retirer de l'argent au moyen d'un DAB à carte'* et le but '*Vérifier la validité de la carte'* sont à différents niveaux d'abstraction, le second étant un besoin nécessaire à la satisfaction du premier.

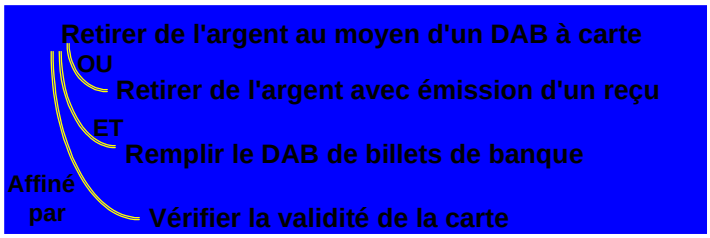


Figure 2.2 : Les relations entre fragments de besoin

L'Ecritoire reconnaît trois niveaux d'abstraction dans l'expression des FBs et de leurs buts, donc des besoins : *comportemental, fonctionnel et physique*.
Un FB du niveau *comportemental* identifie les services qui doivent être mis en œuvre par le système pour satisfaire un but de gestion. '*Améliorer les services à nos clients en mettant à leur disposition un service de retrait d'argent'* est, par exemple, un but du niveau comportemental.
Au niveau *fonctionnel* on se focalise sur les interactions de l'usager avec le système qui permettent de mettre en œuvre les services identifiés au niveau comportemental. Les buts de ce niveau sont fonctionnels et les scénarios associés organisent les séquences d'interactions permettant d'atteindre ces buts. '*Retirer de l'argent au moyen d'un DAB à carte'* est un but du niveau fonctionnel. Ce niveau établit donc un pont entre les objectifs de gestion ('*Améliorer les services à nos clients'*) et les besoins fonctionnels du système (permettre de '*Retirer de l'argent au moyen d'un DAB à carte').*

Le niveau physique se centre sur la découverte de ce que le système doit faire pour permettre les interactions sélectionnées au niveau fonctionnel. Le 'QUOI' est exprimé en termes d'actions internes au système requérant la manipulation d'objets internes mais aussi des objets externes tels que d'autres systèmes. C'est à ce niveau que l'on définit les contraintes logicielles qui permettent de satisfaire les besoins fonctionnels. *'Vérifier la validité de la carte'* est un besoin du niveau physique.
La hiérarchie de FBs est illustré à la Figure 2.4. On observe (a) la relation verticale entre FBs placés aux trois niveaux d'abstraction, comportemental, fonctionnel et physique et (b) les relations horizontales entre FBs placés à un niveau d'abstraction donné fondées sur les liens ET/OU.

2.1.3 Le processus de découverte et le mouvement bidirectionnel but-scénario

Le processus de découverte des besoins conduit par L'Ecritoire est organisé en deux principales activités :

- la *découverte de but* et
- *la rédaction de scénarios.*

Dans ce processus, la découverte de buts, c'est-à-dire de besoins et la rédaction de scénarios sont des activités complémentaires, la première suivant la seconde. Comme le suggère la Figure 2.3, ces activités sont répétées de façon à produire la hiérarchie de FBs de manière incrémentale.

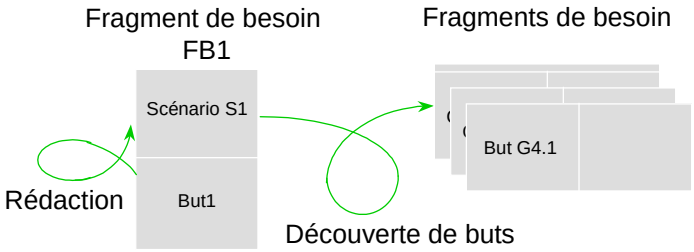


Figure 2.3 : Le mouvement bidirectionnel entre but et scénario

Le processus de découverte des besoins peut être vu comme un flux d'étapes, chacune exploitant la relation entre but et scénario dans les deux directions : *en avant* du but vers le scénario et *en arrière*, du scénario vers le but. Chaque étape débute avec un but et la relation but-scénario est exploitée *en avant* pour rédiger un scénario qui est une concrétisation possible du but. Ensuite, la relation est exploitée en sens inverse, *en arrière*, pour découvrir de nouveaux buts. Dans les étapes suivantes, partant des buts de ces nouveaux FBs, des scénarios sont rédigés et le processus de découverte continue. Ce processus itératif et incrémental est présenté à la Figure 2.4.

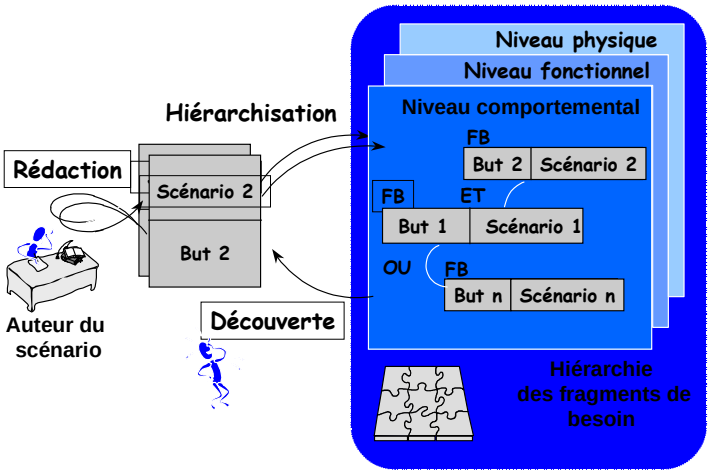


Figure 2.4 : Le processus itératif et incrémental de découverte des besoins

On voit donc que le processus de découverte des besoins correspond à un mouvement bidirectionnel du but vers le scénario et d'un scénario vers des buts.

Dans le sens direct, le scénario est un moyen de pallier au caractère abstrait d'un but concrétisant une possible des ses réalisations dans un scénario. Grâce à cette association les buts irréalistes peuvent être détectés et ainsi les deux limites exprimées par la pratique des approches d'IB dirigées par les objectifs (cf. introduction de cette section) sont réduites.

Dans le sens inverse, du scénario vers les buts, la relation permet de découvrir des buts réalistes. En effet cette découverte est basée sur l'analyse d'un scénario qui par définition, décrit un comportement concret et réaliste du système. On peut donc espérer que les buts résultants de cette analyse soient pertinents.

2.1.4 Les règles de guidage du processus

Comme le montre la Figure 2.5, le processus de découverte de besoins est sous le contrôle du logiciel *L'Ecritoire* qui guide l'utilisateur (que l'on appelle dans la suite l'auteur de fragment de besoin (AFB)) dans l'enchaînement des étapes et dans la mise en œuvre des activités de chacune des étapes.

Pour assurer le guidage du flux d'avancement dans le processus, le logiciel *L'Ecritoire* utilise un modèle du processus de découverte qui est présenté en détail au chapitre 4.

Pour aider et conseiller l'AFB dans la réalisation de chacune des deux activités du processus, découverte de buts et rédaction de scénario, le logiciel *L'Ecritoire* met en œuvre des *règles* déclenchées et exécutées de manière automatique mais avec l'intervention de l'AFB notamment pour prendre la décision en cas de choix multiples.

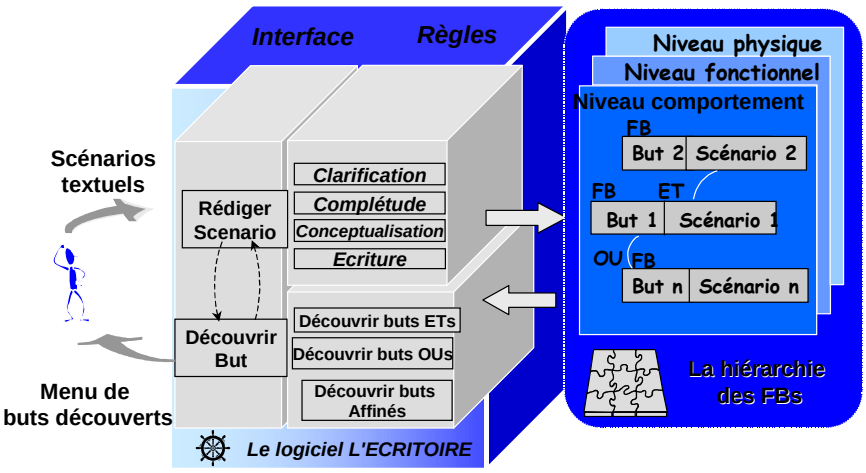


Figure 2.5: Le guidage du processus de découverte par le logiciel L'Ecritoire

Les règles sont de deux types :

(1) *de rédaction des scénarios* et
(2) *de découverte de buts.*

Le rôle des règles de rédaction des scénarios est d'assurer la production de scénarios de qualité telle qu'elle rende possible la découverte automatique de buts. Ces règles combinent des *directives de style et de contenu* à des outils linguistiques. Les premières conseillent l'AFB dans l'écriture des scénarios textuels tandis que les secondes aident à analyser le scénario, à corriger les expressions ambiguës et à assurer sa complétude. Ces règles font appel à une grammaire de cas et des patrons linguistiques détaillés au chapitre 4.

Les règles de découverte guident l'AFB dans la découverte de nouveaux buts et donc de nouveaux fragments de besoin. La découverte est basée sur l'analyse des scénarios et met en œuvre l'une des trois stratégies de découverte : d'affinement, de composition et d'alternative. Ces trois stratégies correspondent aux trois types de relations pouvant exister entre fragments de besoin. Etant donné un FB <G,Sc> :

- la stratégie de composition cherche des buts Gi liés à G par un lien ET,
- la stratégie d'alternative aide à découvrir des buts Gj liés à G par un lien OU,
- la stratégie d'affinement conduit à élucider des Gk à un niveau d' abstraction plus fin que G.

Ces règles sont détaillées au chapitre 4 de cet article.

3. Le modèle des besoins

Ce chapitre présente en détail le *modèle des besoins* et l'illustre avec le cas du GAB. Elle comporte une présentation générale du modèle à travers son méta-modèle puis détaille et illustre les différentes notions qui composent le modèle.

3.1 Survol du modèle

La Figure 3.1 présente le modèle par un schéma UML ou méta-schéma. Elle met en évidence les notions clés du modèle et leurs interrelations. La notion centrale est celle de *fragment de besoin (FB)* défini comme un couple <but, scénario>, le but étant l'objectif à atteindre et le scénario étant une description en langage naturel du comportement du système permettant de satisfaire le but.

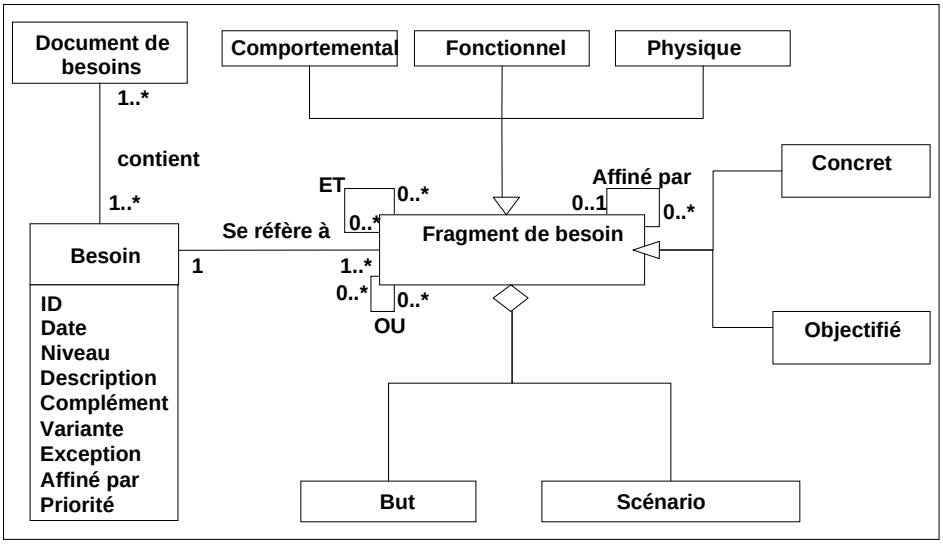


Figure 3.1 : Le méta modèle de l'approche L'Ecritoire

Une première spécialisation de la notion de FB permet de les classer en fonction du niveau d'abstraction auquel ils sont exprimés. Il y a trois niveaux *Comportemental*, *Fonctionnel*, et *Physique*. Un FB *comportemental* sert à identifier les services que le système doit fournir à l'entreprise. Le niveau *fonctionnel* détaille chaque service du niveau *comportemental* comme un ensemble d'interactions entre agents (le système lui-même et ses utilisateurs). Un FB fonctionnel décrit le flux d'actions permettant d'assurer un service du système. Le niveau *physique* détaille chaque interaction du niveau fonctionnel par un ensemble d'actions internes au système. Chaque action peut faire référence à des objets du système et à des objets externes comme d'autres systèmes. Il y a donc clairement une hiérarchie de niveaux : le niveau comportemental étant le plus haut, celui qui correspond à la vision le plus synthétique de ce qu'il est attendu du système ; le niveau fonctionnel étant le niveau intermédiaire d'expression des fonctions envisagées du système et le niveau physique celui des exigences sur la mise en œuvre des fonctions.

Une seconde spécialisation de la notion de FB introduit deux types : *Concret et Objectifié*. Un FB concret est attaché à un scénario tandis qu'un FB objectifié est réduit à un but. On verra que les FBs concrets sont la majorité et que les FBs objectifiés sont l'exception.

Les FBs ont entre eux des relations qui s'expriment au moyen de trois types de liens *'ET', 'OU' et 'Affiné par'*. Les deux premiers liens permettent de relier les FBs appartenant au même niveau d'abstraction dans une structure que l'on peut qualifier d'horizontale. Le troisième lien appelé 'A*ffiné par'* permet de relier les FBs de différents niveaux d'abstraction selon une structure verticale.

Les FBs sont au cœur du processus de L'Ecritoire qui a pour objectif principal d'aider à leur découverte. Ils constituent les énoncés des besoins que le modèle propose de regrouper et documenter dans une fiche de *Besoin*. L'ensemble de ces fiches constitue le *'Document de besoins'*. La Figure 3.1 montre qu'un certain nombre de paramètres permettent de décrire un besoin : *ID* est le numéro d'identification, *'Description'*, *'Priorité'* définit le

degré de priorité du besoin, les paramètres *'Complément', 'Variante', 'Exception' et 'Affiné par'* permettent d'établir la relation d'un besoin avec les autres besoins du document.

Les sections suivantes présentent les notions introduites ci-dessus. La prochaine section présente le modèle et la structure linguistique du but. La section 3.3 présente le modèle et la structure linguistique de scénario. A la section 3.4 nous détaillons la notion du fragment de besoin et leurs relations. La section 3.5 est relative à la documentation des besoins.

Au cours de ce chapitre, nous illustrons la présentation de chaque notion par des exemples tirés du processus d'extraction des besoins nécessaires à la conception d'un GAB.

3.2 La notion de but

Notre expérience pratique a montré qu'il n'est pas aisé de formuler les buts. Pour éviter des erreurs dans leur expression, lever les ambiguïtés de l'expression en langage naturel et guider l'ingénieur des besoins vers une formulation correcte des buts, nous proposons un formalisme de *but* dérivé d'une définition *linguistique* développée par [Prat97]. Celle-ci est fondée sur l'étude d'un large échantillon de buts décrits dans la littérature, et sur le formalisme de la grammaire de cas [Fillmore68, Dik89].

Les cas sont des types des relations sémantiques que des groupes des mots entretiennent avec le verbe dans toute clause d'une phrase. Les cas permettent de représenter la structure profonde d'une phrase par opposition à sa structure de surface. Fillmore [Fillmore68] a initialement identifié six cas : *agent, instrumental, datif, factuel, locatif et objet*. Par exemple, dans « La clé ouvre la porte », la clé joue le rôle d'instrumental et la porte celui d'objet.

Selon [Prat97], l'approche par cas appliquée à la formulation des buts conduit à représenter un but par un *verbe suivi de paramètres* Une expression de but est ainsi composée d'un verbe et d'un ou plusieurs paramètres, chaque paramètre jouant un rôle particulier à l'égard du verbe. N. Prat a identifié 12 paramètres : *Objet, Résultat, Source, Direction, Référence, Manière, Moyen, Qualité, Quantité, Bénéficiaire, Lieu, et Temps* parmi lesquels 7 nous ont paru pertinents dans le contexte de l'approche L'Ecritoire. Ce sont les paramètres suivants: *Objet, Résultat, Source, Destination, Bénéficiaire, Manière et Moyen*. Il en résulte un méta-schéma de la notion de but présenté à la Figure 3.2. A noter que les deux paramètres *objet* et *résultat* sont généralisés par le paramètre *Cible, source* et *destination* par le paramètre *Direction* et *manière, et moyen* par le paramètre *Voie (*Figure 3.2).

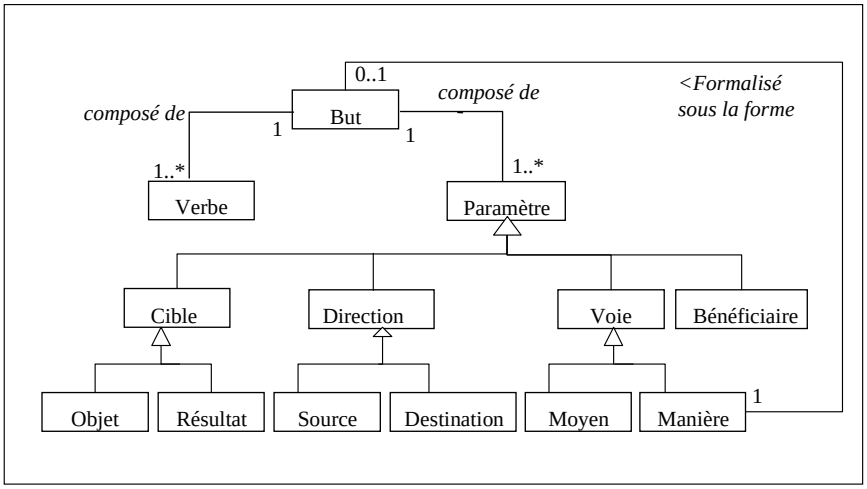


Figure 3.2 : La structure du but en notation UML

- **Cible :** La *cible* (Cib) concerne les entités affectées par le but. Il y a deux types de cibles : l'*Objet* et le *Résultat.*
- **Objet (Obj) :** L'entité/ (les entités) ou le(s) but(s) affecté (s) par le but désigné par le verbe. L'objet existe avant la réalisation du but et peut éventuellement être modifié ou supprimé par celui-ci. Par exemple dans le but,

  '$(Retirer)_{verbe}$ *de* $(l'argent)_{Obj}$ *du* $(GAB)_{So}$ *(dans le cas normal)*$_{Man,}$

  'L'argent' est un objet puisqu'il existe avant la réalisation du but.

- **Résultat (Rés) :** L'entité/ (les entités) ou le(s) but(s) qui résulte de la réalisation du but désigné par le verbe. Par opposition au cas objet, les entités/buts du cas résultat n'existent pas préalablement au but et sont concrétisés par la réalisation du but. Par exemple dans le but,
  '(Générer)$_{verbe}$ (un rapport de transaction)$_{Rés}$ pour (la banque)$_{Des}$',
  'un rapport de transaction' est un résultat puisqu'il n'existait pas préalablement en tant qu'entité physique avant la réalisation du but.

- **Direction** : Il y a deux types de *Direction* (Dir) appelées *Source* (So) et *Destination* (Des).
- **Source (So) :** Point de départ du but (source d'information ou lieu physique). Par exemple dans,
  '*(Retirer)$_{verbe}$ de (l'argent)$_{Obj}$ du (GAB)$_{So}$ (dans le cas normal)$_{Man'}$* ,
  '*GAB'* est la source puisqu'il représente l'endroit initial de '*l'argent'*.
- **Destination (Des) :** Point d'arrivée du but. C'est l'inverse de la source. Par exemple dans,
  '(Transférer)$_{verbe}$ (la somme déposée)$_{Obj}$ au (compte du client')$_{Des}$,
  'compte du client' est la destination puisqu'il est le point d'arrivée du transfert.

- **Voie :** Une voie est spécialisée en Manière (*Man*) et Moyen (*Moy*)
- **Moyen(Moy)** : C'est l'entité ou l'outil au moyen duquel le but est atteint. Par exemple dans,
  '(Fournir)$_{verbe}$ (des services de retrait bancaires)$_{Rés}$ à (nos clients)$_{Bén}$ au (moyen d'un GAB) $_{Moy}$,
  le *GAB* représente l'outil au moyen duquel le but est atteint.
- **Manière(Man):** Spécifie la façon d'atteindre le but. Par exemple dans
  '(Fidéliser)$_{verbe}$ (nos client)$_{Bén}$ (en fournissant des services de retrait d'argent au moyen d'un GAB)$_{Man}$'
  'en fournissant des services de retrait d'argent au moyen d'un GAB' représente une manière puisqu'elle décrit la façon d'atteindre le but.
  Une manière peut être un but. C'est le cas de l'exemple précédent où la manière *(en fournissant des services de retrait d'argent au moyen d'un GAB)$_{Man}$* associée au verbe *Fidéliser* est un but
  en ((fournissant)$_{verbe}$ (des services de retrait d'argent)$_{Rés}$ au (moyen d'un GAB)$_{Moy}$,
  Cette propriété permet de définir un but de manière récursive. Ceci est modélisé à la Figure 3.2 par l'association entre *Manière* et *But*.
- **Bénéficiaire (Bén) :** La personne ou le groupe en faveur de qui le but doit être atteint. Par exemple dans,
  '(Fournir)$_{verbe}$ (des services de retrait bancaires)$_{Rés}$ à (nos clients)$_{Bén}$ au (moyen d'un GAB)$_{Moy}$,
  '*nos clients'*, représente le bénéficiaire.

La structure de but identifiée proposée ci-dessus offre trois avantages : (a) elle permet d'identifier et d'expliquer la différence entre deux buts proches dans leur énoncé mais différents dans leur contenu, (b) elle est 'universelle' car elle permet d'énoncer n'importe quel but d'utilisation d'un système indépendamment de la méthode employée pour découvrir les buts, et (c) elle permet de raisonner sur le but comme nous le verrons plus loin.

3.3 La notion de scénario

L'approche L'Ecritoire préconise l'utilisation de scénarios textuels, c'est-à-dire écrits sous forme de textes en langage naturel. Ce choix à été motivé par une enquête industrielle [Wiedenhaup98] montrant qu'une large proportion des utilisateurs (80%) souhaitent écrire des scénarios en langage naturel. On peut donc voir un scénario comme un texte. Il correspond par ailleurs, à un concept fait d'autres concepts en interrelation. Un scénario a d'une part, une structure linguistique et d'autre part, une structure conceptuelle. Dans cette section, on présente successivement la *structure conceptuelle* puis la *structure linguistique* d'un scénario.

3.3.1 La structure conceptuelle de scénario

On introduit progressivement les différents concepts qui participent à la notion de scénario.

3.3.1.1 Notion d'action

Dans L'Ecritoire, un scénario définit *'un comportement possible du système limité à un ensemble d'interactions entre plusieurs agents*' [BenAchour99]. La définition est conforme à celle de Rational telle qu'elle s'exprime dans le RUP et à beaucoup de celles des approches par scénarios que nous avons analysées dans [Rolland98c]. Cette définition met l'accent sur les deux notions :

a) d'interaction,
b) et de flux que nous détaillons successivement.

Dans le modèle de scénario, les flux sont les actions complexes alors que les interactions sont des actions atomiques. La Figure 3.3 montre que ces deux notions Flux et Interaction sont généralisées par la notion d'Action. Cette généralisation est exclusive (noter la contrainte 'Exclusif' entre les deux liens de spécialisation).

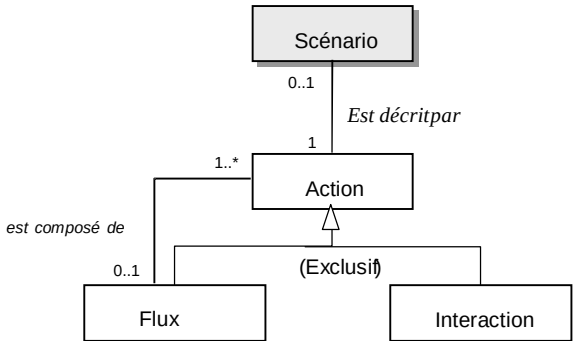


Figure 3.3 : La notion d'action dans un scénario

Le lien '*est composé de*' entre '*Flux*' et '*Action*' de la Figure 3.3 indique qu'un flux est composé d'au moins une action (cardinalité 1..*). De la même manière les actions peuvent être à leur tour composées de flux.
Le lien '*est décrit par*' entre '*Scénario*' et '*Action*', montre qu'un scénario est décrit par une seule action (cardinalité 1). Les actions décrivant un scénario sont souvent composées de plusieurs flux. Cependant, il n'est pas exclu qu'un scénario soit décrit par une seule interaction. D'un autre côté, une action fait partie d'une description de scénario ou d'un flux d'interactions. Par conséquent, les actions ne sont pas partagées entre plusieurs scénarios ni même entre plusieurs flux d'interactions au sein d'un même scénario (d'où la cardinalité 0..1 de '*Action*' vers '*Scénario*').

3.3.1.2 La notion d'interaction

Une interaction décrit le comportement d'un agent. Les interactions sont de différents types : demande de service, fourniture de service, demande d'information, fourniture d'information, communication d'une ressource physique, etc.

Par exemple,

'Le client introduit une carte bancaire dans le GAB' et
'le GAB affiche un message au client demandant de taper son code personnel',

sont deux interactions, la première est une communication de ressource physique et la deuxième est une demande d'information. La Figure 3.4 précise la notion d'interaction :

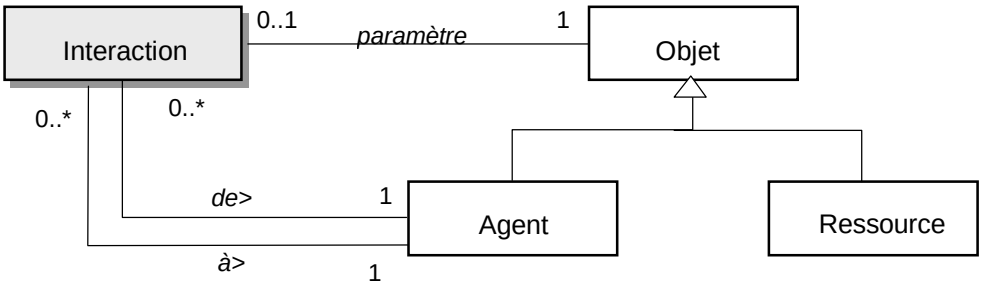


Figure 3.4 : Les paramètres d'une interaction

Toute interaction met en jeu deux '*Agents*', l'un étant source et l'autre cible de l'interaction. Par exemple, l'interaction, '*Le client introduit une carte bancaire dans le GAB*' est dirigée de l'agent '*client*' vers l'agent '*GAB*'.
Elle a un '*paramètre*' qui est un '*Objet*' échangé entre les agents du scénario. Dans l'interaction précédente, '*la carte bancaire*' est le paramètre de l'interaction.
L'Objet est un *Agent* ou une *Ressource*. Un agent est un objet source ou destination d'au moins une interaction. Les ressources sont les objets échangés dans un scénario et qui ne sont ni source ni destination d'interaction.

3.3.1.3 La notion de flux

Les flux permettent de définir l'ordonnancement des interactions d'un scénario. Comme le montre la Figure 3.5 les flux sont de quatre types : s*équence, concurrence, répétition et condition*.

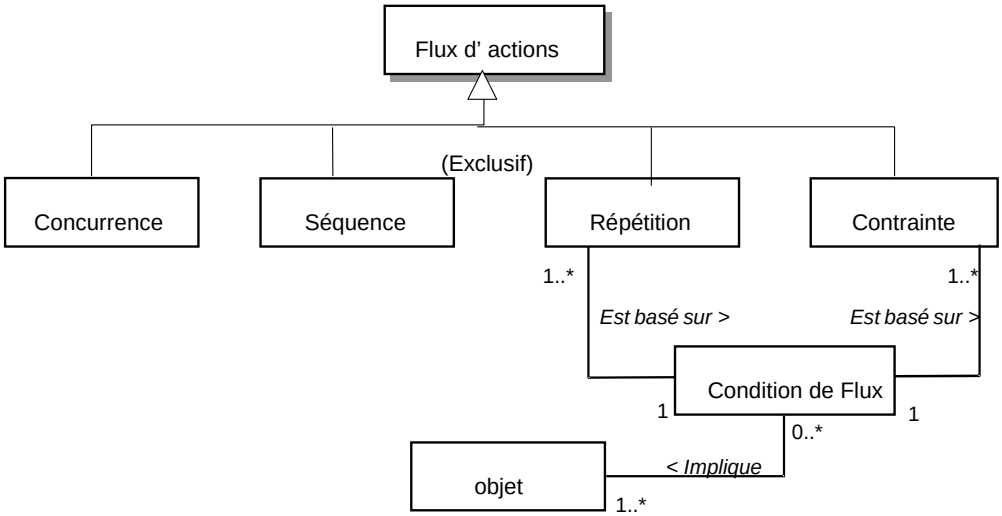


Figure 3.5 : La notion de flux d'interactions

Une *séquence* est composée de deux actions, la deuxième se déroulant comme conséquence de la première. Par exemple, la proposition,

'le client insère une carte dans le GAB, puis le GAB vérifie la validité de la carte','

est un exemple de deux interactions en séquence. Dans une séquence, l'une des actions peut être un flux. Par exemple, dans:

'*Le client insère une carte dans le GAB. Le GAB affiche un message d'accueil au client, puis le GAB vérifie la validité de la carte'*, la séquence est composée de l'interaction :

'Le client insère une carte dans le GAB',

et d'un flux composé de deux interactions :

'*Le GAB affiche un message d'accueil au client puis, le GAB vérifie la validité de la carte'*..

Contrairement à une séquence, il n'y a aucun ordre spécifique entre deux actions *concurrentes*. Elles sont perçues par l'utilisateur comme étant exécutées en parallèle.

'*Le GAB avale la carte tandis que le message 'code personnel invalide' est affiché au client'*

est un exemple de flux concurrent. Comme précédemment pour la séquence, la concurrence peut être entre deux flux d'actions complexes tels deux séquences d'interactions par exemple.

Lorsqu'un flux est *contraint*, l'exécution de l'ensemble des actions qui suivent la contrainte est soumis à la vérification d'une *condition de flux*.

'Si la carte est valide le GAB affiche un message demandant au client son code personnel. Le client tape son code personnel'…est un exemple de flux contraint par la condition '*Si la carte est valide'*. Rappelons qu'un scénario décrit '*un comportement possible'* et donc *un seul*. Celui-ci peut être contraint mais il ne peut y avoir dans le même scénario description de comportements alternatifs. La contrainte se rapproche du IF….THEN mais exclue le IF…. THEN……ELSE.

La *répétition* permet de décrire des flux d'actions qui se répètent plusieurs fois dans un scénario. La condition d'itération est spécifiée par une '*condition de flux*'. Le flux suivant est un exemple de flux itératif :

'Au plus trois fois et jusqu'à ce que le code soit valide, le GAB affiche un message demandant le code personnel au client. Le client saisit son code et le GAB vérifie la validité du code fourni' dont la condition d'itération est *'Au plus trois fois et jusqu'à ce que le code soit valide'*.

3.3.1.4 Etats initiaux et finaux

Comme le montre la Figure 3.6, un scénario est caractérisé par des *états initiaux* et des *états finaux*. Les premiers définissent la *pré-condition* pour que le scénario puisse être déclenché et les seconds à la post-condition qui doit être satisfaite après l'exécution du scénario. Les uns et les autres se définissent comme des états d'objets.

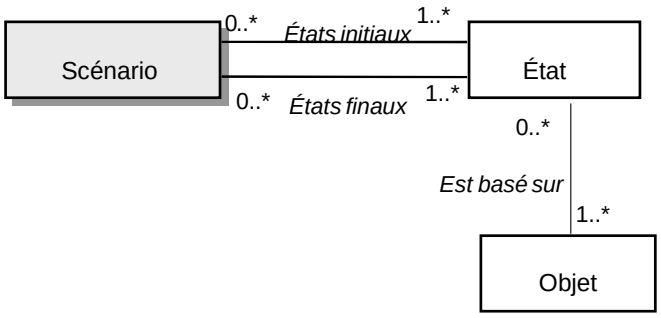


Figure 3.6 : La notion d'état d'un scénario

Par exemple,
'Le client a une carte' et '*Le GAB est prêt*' sont les deux états initiaux du scénario, '*Retirer de l'argent du GAB au moyen d'une carte dans le cas normal'* alors que *'Le client a l'argent'* et '*Le GAB est prêt'* sont les états finaux de ce même scénario.

3.3.1.5 Scénarios normaux et scénarios exceptionnels

Comme le montre la Figure 3.7, nous distinguons deux types de scénarios, les *scénarios normaux* et les *scénarios exceptionnels.*

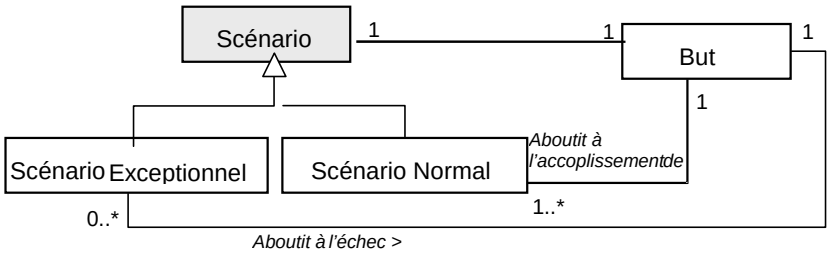


Figure 3.7 : scénarios normaux et scénarios exceptionnels

Un scénario normal permet d'atteindre le but auquel il est associé tandis qu'un scénario exceptionnel est ne le permet pas. Le scénario correspondant à une transaction de retrait d'argent qui aboutit à la délivrance de la somme demandée est un scénario normal ; celui qui décrit la saisie infructueuse du code personnel est un scénario dit exceptionnel.

3.3.2 La structure linguistique de scénario

La section 3.3.1 a montré quelle est la structure conceptuelle sous-jacente à tout scénario. Dans cette section nous montrons quelle est sa structure linguistique et la relation qui existe entre la vue textuelle et la vue conceptuelle. La 19 montre cette correspondance.

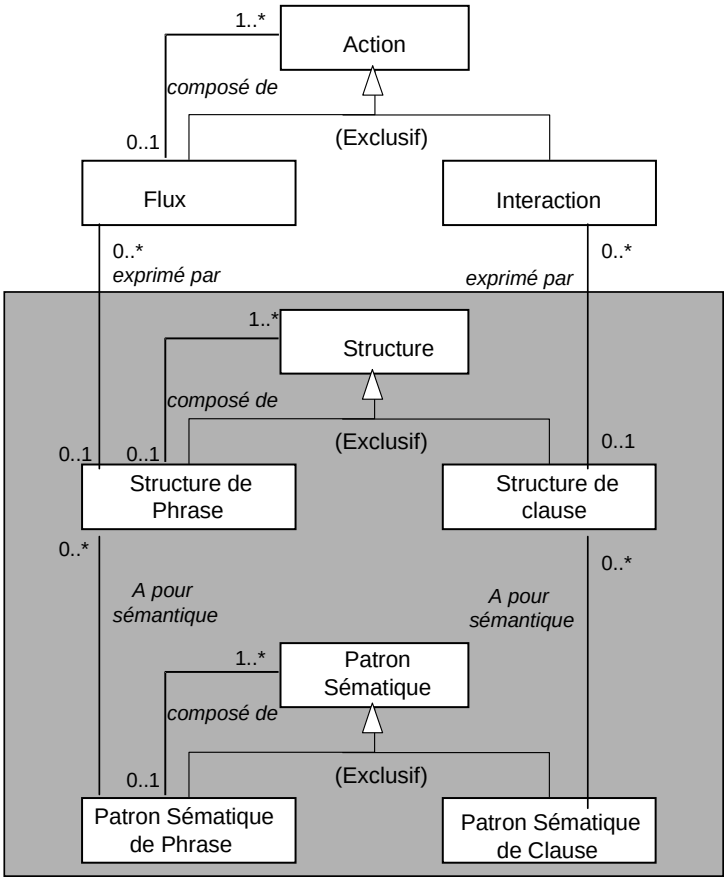


Figure 3.8 : Correspondance entre structure conceptuelle et structure linguistique

La partie grisée de la Figure 3.8 correspond à la structure linguistique d'un scénario tandis que la partie supérieure résume la structure conceptuelle développée à la section précédente. On voit que la structure linguistique est organisée en deux niveaux : celui des *structures linguistiques* de phrases et de clauses et celui des *patrons sémantiques.* Les structures linguistiques correspondent aux structures syntactiques et grammaticales des phrases et des clauses du scénario ; on dit que ce sont les *structures de surface.* Au contraire, les patrons sémantiques servent à définir le sens profond des phrases et clauses; on dit qu'ils sont les *structures profondes*.

La correspondance entre les deux structures montre que les structures linguistiques de clause correspondent à l'expression des interactions tandis que les structures linguistiques de phrase correspondent à la formulation textuelle des flux. De même les patrons sémantiques de clause donnent le sens profond, la sémantique des interactions alors que ce sont les patrons sémantiques de phrase qui permettent de comprendre la sémantique des flux.

3.3.2.1 Illustration de la correspondance entre concepts du modèle de scénario, patrons sémantiques et structures linguistiques

On trouvera dans [BenAchour99] une étude détaillée de l'approche linguistique de L'Ecritoire. On se limite dans cette section à un exemple illustratif de la correspondance entre patron sémantique, structure linguistique et concepts du modèle de scénario.

Prenons le cas du *patron sémantique de communication* qui s'énonce comme suit :

…. Communication (Verbe) [Agent ; Objet ; Source ; Destination].

Ce patron identifie la sémantique de toute communication qui est caractérisée par (a) *l'Agent* qui est l'entité active initiant ou contrôlant l'action de communication ; (b) *l'Objet* de la communication qui en est l'entité

passive ; (c) la *Source* et la *Destination* de l'objet communiqué. La communication fait référence à un *Verbe* tel que 'obtenir', 'donner', 'imprimer', 'transmettre'.

Ceci est une instance du patron sémantique de communication :

Communication Action ('obtenir') [Agent : 'le moteur du GAB' ; Objet : 'une validation de retrait d'argent' ; Source : 'le système central de la banque ; Destination : 'le moteur du GAB']

Qui définit dans les termes du patron, la sémantique de la clause suivante :

'Le moteur du GAB obtient la validation d'un retrait d'argent du système central de la banque'.

Pour un patron, il y a de nombreuses structures linguistiques possibles. Par exemple, les clauses suivantes :

'*Le moteur du GAB cherche à obtenir du système central de la banque une validation de retrait d'argent'*

'*Une validation de retrait d'argent du système central de la banque est en cours d'obtention par le DAB'*

'L'obtention par le DAB de la validation d'un retrait d'argent auprès du système de la banque'

sont différentes mais seront interprétées de la même manière. Toutes correspondent à la même instance du patron sémantique de communication telle qu'elle est présentée ci-dessus.

Les éléments du patron sont en relation avec les concepts du modèle de scénario. Ainsi dans le cas de l'exemple du patron de communication ci-dessus, il y a correspondance entre

- *l'Agent* du patron et *l'Agent* de l'interaction (Figure 3.4),
- *l'Objet* du patron et *l'Objet* de l'interaction (Figure 3.4),
- La *Source* et la *Destination* du patron et les liens '*de*' et '*à*' de l'interaction (Figure 3.4).

On comprend donc que la correspondance entre la vue structurelle d'un scénario et sa forme linguistique puisse être établie strictement. Ceci par l'intermédiaire de l'analyse linguistique du scénario permettant d'associer à chaque interaction et à chaque flux, la structure linguistique qui leur correspond, d'en déduire le patron sémantique adéquat et finalement, d'associer les éléments du patron à des instances de concepts du modèle de scénario.

3.4 La notion de fragment de besoin

Ayant étudié en détail la notion de scénario et celle de but, on est en mesure désormais de coupler les deux pour former un *fragment de besoin*. La notion de fragment de besoin est présentée à la Figure 3.9.

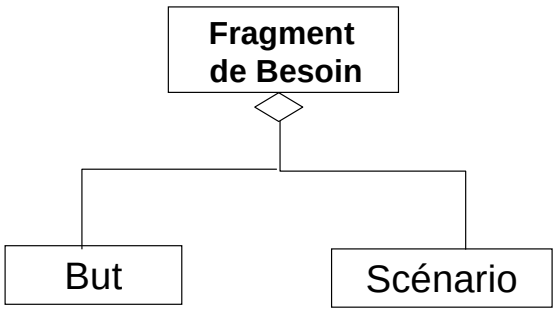


Figure 3.9 : La notion du fragment du besoin

La Figure 3.10 donne un exemple de fragment de besoin.

Fragment de besoin F1 : Retrait d'argent / normal
But B1 : *Retirer de l'argent du GAB au moyen d'une carte bancaire dans le cas normal*
Scénario SC1
Etats initiaux : '*le client a une carte bancaire', 'le GAB est prêt'*.
Le client insère une carte dans le GAB, le GAB vérifie la validité de la carte. Si la carte est valide alors, un message demandant le code est affiché par le GAB. Le client tape son code. Si le code est valide alors, un message demandant le montant est affiché par le GAB au client. Le client tape le montant désiré. Si le montant est disponible, le GAB éjecte la carte au client. Si un reçu est demandé par le client, le GAB imprime un reçu au client et il délivre les billets au client.
États finaux: *'le client a une carte bancaire', 'le GAB est prêt', 'le client a de l'argent', 'le client a un reçu'*.

Figure 3.10 : Le fragment de besoin FB1

Il est important de remarquer que le paramètre *manière* du but permet de caractériser la spécificité du comportement du système décrit par le scénario du même fragment. Ainsi par exemple, la manière dite *'dans le cas normal'* du fragment FB1 permet de comprendre que les interactions du scénario correspondent à la transaction normale de retrait d'argent.

Le cas d'une utilisation incorrecte de carte sera capturé dans une autre fragment de besoin dont le but sera *'Retirer de l'argent du GAB au moyen d'une carte bancaire invalide'* et dont le scénario exprimera la transaction d'échec due à la reconnaissance par le GAB de l'invalidité de la carte. Le fragment de besoin FB2 correspondant à ce cas est présenté à la
Figure 3.11 ci-dessous.

Fragment de besoin F2 : Retrait d'argent / carte invalide
But B2 : *Retirer de l'argent du GAB au moyen d'une carte bancaire invalide*
Scénario SC2
Etats initiaux : '*le client a une carte bancaire', 'le GAB est prêt'*.
Le client insère une carte dans le GAB, le GAB vérifie la validité de la carte. Si la carte est invalide alors, un message d'erreur est affiché par le GAB et la carte est rejetée.
États finaux : *'le client a une carte bancaire', 'le GAB est prêt'*.

Figure 3.11 : Fragment de besoin FB2

Chaque fragment de besoin correspond donc à un cas d'usage du système. Pour un même but 'racine' tel que *Retirer de l'argent du GAB au moyen d'une carte bancaire*, l'approche L'Ecritoire guide dans la recherche de toutes les variantes possibles d'atteinte du but et d'échec du but. Chacune est décrite par un fragment de besoin. Il y a donc une 'grappe' de fragment de besoins pour un but 'racine'.

3.5 Les relations entre fragment de besoin

Chaque fragment de besoin apporte une connaissance parcellaire sur le système à construire. Il est intuitivement évident qu'il faille organiser l'ensemble de ces connaissances parcellaires pour qu'elles fassent un tout complet et cohérent. Ce besoin a été ressenti dans la pratique des approches par scénarios et la découverte des liens entre parcelles reste un problème d'actualité. Dans L'Ecritoire nous avons identifié trois types de relations entre fragments de besoin nommées c*omposition, alternative* et *affinement* (Figure 3.12).

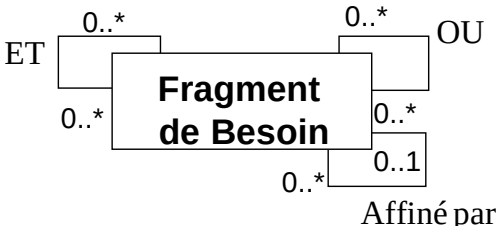


Figure 3.12 : Les trois types de liens entre les FBs

Les deux premiers liens sont similaires aux liens entre buts des approches d'ingénierie des besoins dirigées par les buts, KAOS [Lamsweerde98], ELEKTRA [ELEKTRA99], GBRAM [Anton96]. Le troisième type de lien permet de classer les FBs a différents niveaux d'abstraction de la connaissance du futur SI.

3.5.1 Les liens 'ET et 'OU'

Le lien 'ET' associe des FBs complémentaires. Ils capturent des comportements distincts mais qui sont complémentaires les uns des autres et nécessaires pour assurer une description complète de l'ensemble des fonctionnalités du système. Considérons par exemple les trois fragments de besoins FB1, FB2 et FB3 ayant comme buts respectifs :
B1 : $(Obtenir)_{Ver}$ *(une carte)*$_{Rés}$ *de (la banque)*$_{So}$ *(dans le cas normal)*$_{Man}$,
B2 : $(Retirer)_{Ver}$ *(de l'argent)*$_{Obj}$ *du (GAB)*$_{So}$ *au (moyen d'une carte bancaire)* $_{Moy}$ *(dans le cas normal)*$_{Man}$, et
B3 : $(Transmettre)_{Ver}$ *(un rapport de transaction)* $_{Rés}$ *par (le GAB)* $_{So}$ *au (système central de la banque)*$_{Des}$ *(dans le cas normal)*$_{Man}$.

Ces trois FBs sont complémentaires étant donné que chacun décrit un comportement distinct mais en même temps nécessaire aux deux autres. En effet, pour réaliser B2 (resp B3) il est nécessaire que B1 soit réalisé (resp B2). De la même manière réaliser B1 (resp B2) doit être suivi par la réalisation de B2 (Resp B3).
Les fragments FB1, FB2 et FB3 sont donc complémentaires.

Le lien 'OU' associe des FBs ayant le même but 'racine' (cf. section 3.4) mais des manières différentes de l'atteindre. Dans une 'grappe' telle qu'introduite en section 3.4, les FBs sont liés les uns aux autres par des liens 'OU'. Les FBs liés par des 'OU' sont donc alternatifs, ils correspondent à des buts composés des mêmes paramètres à l'exception du paramètre *'manière'* qui lui, caractérise le flux d'interactions exprimant les différentes manières de réaliser le but. Par exemple, les trois FBs FB1, FB2 et FB3 ayant comme buts respectifs:
'Retirer de l'argent du GAB au moyen d'une carte bancaire dans le cas normal',
'Retirer de l'argent du GAB au moyen d'une carte bancaire avec une carte non valide' et
'Retirer de l'argent du GAB au moyen d'une carte bancaire avec une seule correction du code personnel'
sont liés par des liens 'OU'. Ils décrivent comment *'Retirer de l'argent du GAB au moyen d'une carte bancaire'* de trois manières alternatives,

'dans le cas normal' ou
'avec une carte non valide' ou
'avec une seule correction du code personnel'.

3.5.2 Le lien 'Affiné par'

Les besoins n'échappent pas au principe de décomposition ou principe de Descartes. Un besoin vu comme un tout à un instant temps doit pouvoir être décomposé en plusieurs besoins plus fins à l'instant $t+\Delta$. Par exemple, le besoin B : *'le système DAB doit être capable de vérifier la validité de la carte'* peut être considéré comme une expression suffisamment précise à un instant t mais requérir une description plus fine à un instant ultérieur. Ceci est légitime car pour construire le système DAB il faut comprendre les détails de la vérification d'une carte et donc déterminer les besoins qui découlent du besoin initial B.

Un besoin de type B correspond souvent à une action d'un scénario (ou peut être impliqué par une action/flux de scénario). C'est le cas pour B qui est une action du scénario SC1 du fragment de besoin FB1 présenté à la Figure 3.10. Il n'est pas possible de détailler l'action dans le scénario SC1 lui-même car cela conduirait à un scénario décrivant *plusieurs* comportements distincts ce qui est contradictoire avec la définition d'un scénario. La décomposition de l'action B est donc faite dans un autre scénario d'un FB, FBb lié au fragment de besoin FB1 par une relation *'Affiné par'*.

Les deux fragments de besoin, FB1 et FBb correspondant aux deux buts

B1 : *Retirer de l'argent du GAB au moyen d'une carte bancaire dans le cas normal' et*
Bb *: Vérifier la validité de la carte*

sont liés par un lien de type *'Affiné par'*.

3.6 La hiérarchie des fragments de besoin

Les liens *'ET'* et *'OU'* établissent des relations horizontales entre les FBs : de complémentarité grâce aux liens *'ET'* et d'alternatives au moyen des liens *'OU'*. Les liens *'Affiné par'* établissent des relations verticales entre

FBs. Les trois liens conduisent donc à une organisation hiérarchique des fragments de besoins. La pratique de l'Ecritoire nous a conduit a prédéfinir trois niveaux d'affinement correspondant à trois niveaux d'abstraction dans l'expression des besoins. Ces trois niveaux sont appelés *comportemental, fonctionnel* et *physique.* Par voie de conséquence la hiérarchie de FBs est organisée en trois couches de Fbs que l'on qualifie du nom du niveau : FB comportemental, FB fonctionnel et FB physique (Figure 3.13).

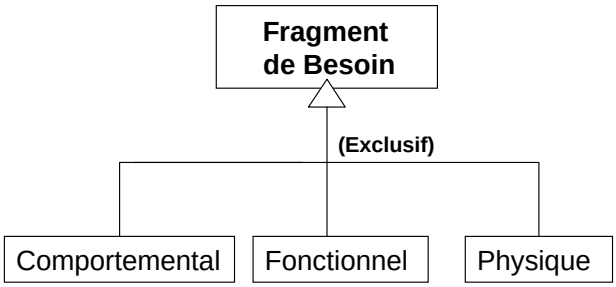


Figure 3.13 : Les trois types de fragments de besoin

Le niveau comportemental

L'objectif du niveau *comportemental* est d'identifier les besoins en services que le système doit fournir à l'entreprise. Un *FB* comportemental associe un *but de gestion* à un *scénario de service*. Le but de gestion correspond à un objectif de l'organisation à l'égard de ses clients. Le scénario de service décrit le flux de services entre les agents (l'un d'entre eux étant le système lui-même), qui est nécessaire pour satisfaire le but de gestion. Une action atomique dans un scénario de service est un service. Par exemple les trois actions du scénario SC1 de la Figure 3.13[1],

1. *Le client obtient une carte bancaire de la banque.*
2. *Le client retire de l'argent à partir du GAB au moyen de la carte bancaire.*
3. *Le GAB donne un rapport de transaction à la banque,*

expriment trois services à fournir respectivement par la banque et par le GAB pour réaliser le but : '*Fournir de services de retrait d'argent à nos clients à partir d'un GAB au moyen d'une carte'*.

B1 :	**SC1 :**
Fournir des services de retrait d'argent aux clients de notre banque à partir d'un GAB au moyen d'une carte bancaire d'une manière normale	1. Le client obtient une carte bancaire de la banque. 2. Le client retire de l'argent à partir du GAB. 3. Le GAB donne un rapport de transaction à la banque.

Figure 3.14 : Exemple de FB comportemental

Les FBs alternatifs de ce niveau décrivent différentes options de conception relatives au futur système. Le niveau comportemental est celui qui cherche à mettre le choix du SI en phase avec les objectifs stratégiques de l'organisation. Il s'adresse aux décideurs à qui l'on soumet plusieurs options parmi lesquelles il leur incombe de choisir. Par exemple, c'est à ce niveau qu'il est naturel de soumettre aux décideurs dans le cas du DAB différentes options de DAB : à carte, à code, à empreintes digitales etc ; différentes options de services : prêts, retrait d'argent, paiement de factures etc.

Le niveau fonctionnel

Le niveau *fonctionnel* détaille chaque service d'un scénario du niveau *comportemental* par un ensemble d'interactions entre agents (le système lui-même et ses utilisateurs) assurant la réalisation de ce service. Ce

[1] Comme nous pouvons le constater sur la Figure 3.14, un scénario dans un FB est décrit par un texte structuré et numéroté. Cette description est utilisée pour faciliter la lisibilité d'un part et l'exploitation des scénarios pour découvrir de nouveaux buts d'autre part. Le mécanisme permettant de passer du scénario en langage naturel en texte structuré est illustré dans le chapitre suivant.

niveau sert à identifier les fonctions du système garantissant les services retenus au niveau comportemental. Il correspond au niveau des ‘use cases’ d’UML et s’adresse aux concepteurs.

Un FB fonctionnel définit le flux d’actions nécessaires pour assurer un service du système. Il couple un *but de service* et un *scénario d’interaction*. Le but de service correspond à une manière d’assurer le service. Le scénario d’interaction associé décrit un flux d’interactions entre le système et ses utilisateurs pour atteindre le but de service. Par exemple, le but B2 du FB fonctionnel de la Figure 3.15 intitulé ‘*Retirer de l’argent à partir du GAB au moyen d’une carte bancaire dans le cas normal’* est issu du service ‘*Le client retire de l’argent à partir du GAB au moyen d’une carte bancaire*’ du scénario de service de la Figure 3.14. Le scénario SC2 associé détaille les interactions entre le client et le DAB qui permettent l’obtention du service.

B2 : Retirer l’argent du GAB au moyen d’une carte bancaire d’une manière normale	**SC2 :** **États initiaux** :L’utilisateur a une carte, le GAB est prêt. **Actions:** 1. l’utilisateur insère une carte dans le GAB 2. le GAB vérifie la validité de la carte 3. si la carte est valide, alors 4. un message demandant le code est affiché par le GAB à l’utilisateur 5. l’utilisateur tape son code sur le clavier du GAB 6. *si le code est valide* , alors 7. un message demandant le montant est affiché par le GAB à l’utilisateur 8. l'utilisateur introduit le montant dans le GAB 9. si le montant est disponible, alors 10. le GAB éjecte la carte à l'utilisateur 11. si un reçu est demandé par l'utilisateur, alors…. 12. le GAB imprime un reçu à l'utilisateur 13. le GAB délivre les billets à l'utilisateur **États finaux** : L’utilisateur a une carte, l’utilisateur a l’argent, l’utilisateur a un reçu, et le GAB est prêt.

Figure 3.15 : Exemple de FB fonctionnel

Notons que l’on a utilisé ici la forme indentée de formulation d’un scénario telle qu’elle est produite par l’outil logiciel L’Ecritoire après la conceptualisation du scénario textuel initial.

Les FBs complémentaires liés par des ‘ET’ de ce niveau capturent l’ensemble des fonctionnalités qui permettront un fonctionnement cohérent du système. garantissant les services identifiés au niveau comportemental. Les FBs alternatifs liés par des ‘OU’ capturent les différentes manières possibles menant à la réalisation du même but de service.

Le niveau physique

Le niveau *physique* s’adresse aux développeurs du futur système qui cherchent à identifier les besoins physiques ou internes du système nécessaires pour accomplir les interactions identifiées au niveau fonctionnel. Le niveau physique détaille ainsi chaque interaction du niveau fonctionnel par un ensemble d’actions internes au système. Chaque action peut faire référence à des objets du système et à des objets externes comme d’autres systèmes.

Un *fragment de besoin physique* est un couple comprenant un *but de système* et un *scénario interne*. Un but de système exprime une manière possible pour exécuter une interaction d’un scénario d’interaction. Un scénario physique décrit le flux d’actions internes au système permettant de satisfaire le but. La Figure 3.16 montre le fragment de besoin physique correspondant au but B3 ‘*Vérifier la validité de la carte dans le cas normal’*. Ce fragment de besoin détaille l’interaction ‘*le GAB vérifie la validité de la carte*’ du fragment de besoin fonctionnel de la Figure 3.15.

B3 Vérifier la validité de la carte d'une manière normale	**SC3** **États initiaux**: 'la carte est dans le lecteur de la carte ', ' l'état de la carte n'est pas validé '. **Actions:** 1. le moteur du GAB demande au lecteur de carte de lire la date de validité et le le numéro de la carte 2. le lecteur de carte envoi le numéro de la carte et la date de validité au moteur du GAB 3. le moteur du GAB demande la date courante à l'horloge du GAB l'horloge du GAB retourne la date courante au moteur du GAB 4. si le numéro de la carte correspond au numéro composé et la date de validité de la carte n 'est pas expirée alors 5. le moteur du GAB envoie le numéro de la carte au système informatique de la banque 6. le système informatique de la banque retourne l'état actuel du compte utilisateur au moteur du GAB 7. si le compte du client n 'est pas sur la liste rouge 8. le moteur du GAB enregistre la validité de la carte. **États finaux**: 'la carte est dans le lecteur de la carte ', ' l'état de la carte est validé '.

Figure 3.16 : Exemple de FB physique

Les FBs complémentaires de ce niveau capturent l'ensemble des besoins relatifs aux traitements internes au système imposés par les choix du niveau fonctionnel précédent. Les FBs alternatifs capturent les différents cas de traitement que le système doit considérer.

NB : Dans certains cas on peut considérer que l'expression du but est une expression suffisamment précise du besoin pour qu'il n'y ait pas lieu de décrire un scénario. On dit dans ce cas que le FB est objectifié (cf. Figure 3.1). C'est le cas du FB physique associé au but

'$(Ejecter)_{Ver}$ $(la\ carte)_{Obj}$ $à\ (le\ client)_{Des}$ $à\ partir\ (du\ lecteur\ de\ la\ carte)_{So}$ $(dans\ le\ cas\ normal)_{Man}$'

Ce but affine l'interaction d'un scénario fonctionnel

'*le lecteur de la carte éjecte la carte au client*',

Il en dit assez en termes de besoin si l'on admet que tout lecteur standard de carte sait mettre en œuvre la fonction d'éjection de la carte.. Il n'y a donc pas lieu d'aller plus loin dans l'affinement du besoin.

La hiérarchie de fragments de besoin

Pour conclure cette section on illustre à la Figure 3.17 une partie de la hiérarchie des fragments de besoin du cas DAB.

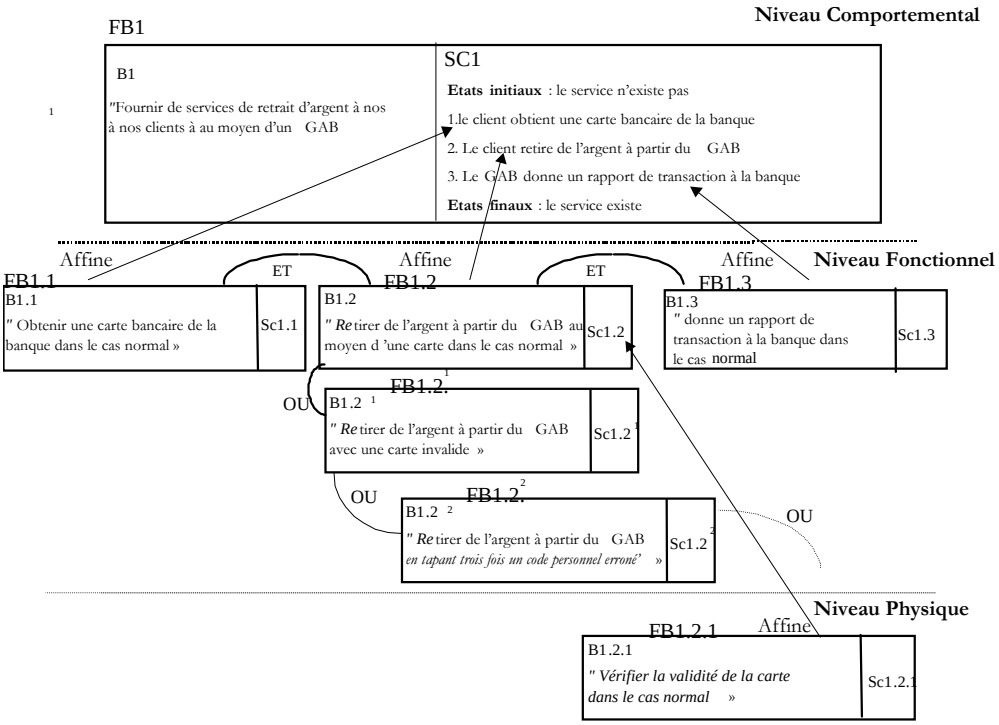


Figure 3.17 : Extrait de la hiérarchie des FBs du cas DAB

3.7 La documentation des besoins

Comme on l'a introduit au début de ce chapitre en section 3.1 le logiciel L'Ecritoire génère un document hypertexte de documentation des besoins. Ce document comprend un ensemble de fiches descriptives des besoins, une par besoin c'est-à-dire une par but de fragment de besoin. En effet, chaque but est une expression de besoin. Comme le montre la Figure 3.1 la fiche de besoin comporte un certains nombre de paramètres descriptifs : *Identité, But, Scénario, Variantes, Exceptions, Composition, Affiné Par et Priorité.* La fiche est illustrée à la Figure 3.18.

.

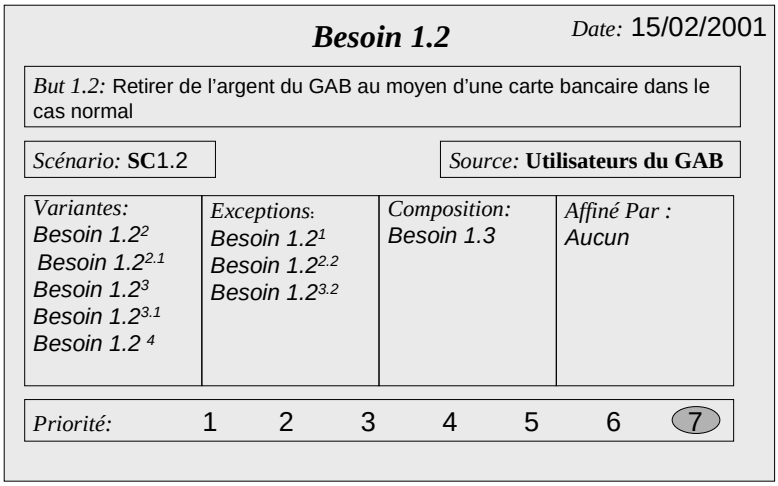


Figure 3.18 : Exemple de documentation de besoin

- ***Identité (ID) :*** Numéro d'identification du besoin ; il fait référence au fragment du besoin auquel il correspond. Par exemple, le besoin de la Figure 3.18 porte le numéro (1.2) faisant référence au fragment de FB1.2 de la hiérarchie des fragments de besoin.

- ***Date :*** Date de génération du besoin. Il permet aussi de gérer les différentes versions d'un besoin. Par exemple, ce paramètre indique que la version du besoin1.2 présentée à la Figure 3.18 a été extrait le 15/02/2001.
- ***But :*** But du fragment de besoin : *'Retirer de l'argent du GAB au moyen d'une carte dans le cas normal'.*
- ***Scénario :*** Ce paramètre fait référence au scénario du FB. Par exemple, le scénario SC1.2 dans l'exemple présenté à la Figure 3.18.
- ***Source :*** Ce paramètre indique les participants au processus d'ingénierie qui ont permis la découverte du besoin. L'exemple indique que le besoin a été extrait par des *futurs utilisateurs du GAB.*
- ***Priorité :*** Indique le degré d'importance que les décideurs accordent de la satisfaction de ce besoin par le futur système. Cette priorité est évaluée dans une échelle de 1 à 7. Dans l' exemple, le besoin1.2 est considéré comme un des objectifs principaux du GAB et donc son degré de priorité est évalué à 7.
- ***Variantes :*** Ce paramètre fait référence aux besoins alternatifs du besoin décrit c'est-à-dire qui lui sont liés par un lien 'OU' et dont les scénarios sont du type 'normal'. Les valeurs indiquées par ce paramètre sont générées à partir de la hiérarchie des FBS. Ainsi par exemple, le besoin1.2 a pour variantes les besoins 1.2^{2},$1.2^{2.1}$,1.2^{3},$1.2^{3.1}$et 1.2^{4} ayant pour buts respectifs,

  But1.2^{2} : Retirer de l'argent du GAB au moyen d'une carte bancaire avec une seule correction du code.
  But1.$2^{2.1}$: Retirer de l'argent du GAB au moyen d'une carte bancaire avec deux corrections du code.
  But1.2^{3} : Retirer de l'argent du GAB au moyen d'une carte bancaire avec une seule correction du montant.
  But1.$2^{3.1}$: Retirer de l'argent du GAB au moyen d'une carte bancaire avec deux corrections du montant.
  But1.2 ^{4} : Retirer de l'argent du GAB au moyen d'une carte bancaire sans demander un reçu.
- ***Exceptions :*** Ce paramètre fait référence aux besoins liés par un lien 'OU' au besoin décrit et dont les scénarios sont du type 'exceptionnel'. Les valeurs indiquées par ce paramètre sont générées à partir de la hiérarchie des FBs. Ainsi par exemple, le besoin1.2 a comme exceptions les besoins 1.2^{1},$1.2^{2.2}$ et $1.2^{3.2}$ ayant pour buts respectifs,

  But1.2^{1} : Retirer de l'argent du GAB au moyen d'une carte bancaire avec carte non valide.
  But1.$2^{2.2.}$: Retirer de l'argent du GAB au moyen d'une carte bancaire avec trois saisies d'un code erroné.
  But1.$2^{3.2}$: Retirer de l'argent du GAB au moyen d'une carte bancaire avec trois saisies d'un montant erroné.
- ***Composition :*** Ce paramètre fait référence aux besoins liés par un lien 'ET' au besoin de la fiche. Les valeurs indiquées par ce paramètre sont générées à partir de la hiérarchie des FBs. Ainsi par exemple, ce paramètre indique que le 15/02/2001, le besoin1.2 a été lié par un lien 'ET' au besoin 1.3 ayant pour but,

  B1.3: Faire un rapport de transaction à partir du GAB au système central de la banque dans le cas normal.
- ***Affiné par :*** Ce paramètre fait référence aux besoins liés par un lien 'Affiné par' au besoin décrit. Les valeurs indiquées par ce paramètre sont générées à partir de la hiérarchie des FBs. Ainsi par exemple, ce paramètre indique que le 15/02/2001, le besoin1.2 n'est affiné par aucun besoin.

En l'absence du logiciel, la documentation proposée peut être gérée par des moyens spécifiques ou bien par un outil logiciel du marché tel que RequisitPro ou Doors.

4. Le processus d'ingénierie des besoins

Ce chapitre présente et illustre le processus d'ingénierie des besoins préconisé par l'approche L'Ecritoire. Comme on l'a montré dans le survol de l'approche en section 2, ce processus se déroule selon un mouvement bidirectionnel entre les deux concepts du fragment des besoin (les buts et les scénarios) de la façon suivante :

> Etant donné un but, un scénario est écrit pour illustrer sa réalisation. Une fois écrit et couplé au but dans un fragment de besoin, le scénario est analysé pour découvrir des nouveaux buts. Ce mouvement bidirectionnel est répété jusqu'à l'extraction de tous les besoins du nouveau système.

Le processus est donc incrémental et itératif : les fragments de besoin sont découverts au fur et à mesure des itérations du processus. Chaque itération comporte deux activités principales :

- Ecrire et conceptualiser un scénario associé à un but,
- Découvrir de nouveaux buts par analyse du scénario.

Le processus est conduit par une équipe comportant un ingénieur des besoins qui joue le rôle de facilitateur et des experts du domaine. On utilisera dans la suite le terme d'Auteur de Fragment de Besoin (AFB) pour désigner cette équipe.

Nous avons formalisé le processus de notre approche par un modèle appelé *carte* qui utilise le formalisme de représentation des processus MAP [Benjamen99, Rolland99]. La section 4.1 introduit la *carte L'Ecritoire* qui est détaillée et illustrée par le cas du DAB en section 4.2.

4.1 La carte L'Ecritoire

La Figure 4.1 présente le modèle du processus de l'approche L'Ecritoire par une *carte* qui est un graphe ordonné *d'intentions* et de *stratégies.* Les intentions, nœuds du graphe, sont les buts à atteindre dans le déroulement du processus ; les stratégies, arcs du graphe, sont les manières pour atteindre les buts. Par exemple, *Découvrir un but* est une intention de la carte L'Ecritoire qui met en évidence le fait que tout ingénieur des besoins doit avant tout découvrir ce que sont les buts/besoins du système à construire. La stratégie *d'affinement* est une manière de découvrir un but qui se base sur l'affinement d'une action d'un scénario en un but du niveau inférieur (cf. l'exemple de la vérification de la validité de la carte à la section 3). Toute carte comporte une intention '*Démarrer'* pour débuter le processus et une intention '*Arrêter'* pour terminer le processus.

L'ordonnancement des intentions dans le graphe exprime l'ordre de précédence nécessaire à la satisfaction des intentions de la carte. Par exemple, il est indispensable d'avoir découvert un but avant d'écrire le scénario associé: l'intention '*Découvrir un but'* précède l'intention '*Ecrire un scénario'*.

La carte se place à un niveau d'expression des tâches du processus sous la forme d'intentions à atteindre et de manière de les atteindre. Elle évite donc d'entrer dans les détails du 'comment exécuter la tâche'. Exactement comme l'IB se focalise sur le POURQUOI (les buts), le modèle MAP (modèle de la carte) se centre sur les intentions du processus et donc sur les raisons, le POURQUOI du processus.

Comment atteindre un but par une stratégie donnée est capturé dans ce qu'on appelle une *section* de carte. Une section est un triplet *<intention source, intention cible, stratégie >*. Le triplet *<Conceptualiser un scénario, Découvrir un but, Stratégie d'affinement>* est une section. Chaque section est associée à des directives méthodologiques qui guident l'ingénieur des besoins dans l'atteinte de l'intention cible de la section. Par exemple, la directive méthodologique attachée à la section précédente explique comment analyser chacune des actions d'un scénario pour décider s'il faut l'affiner sous la forme d'un but du niveau inférieur.

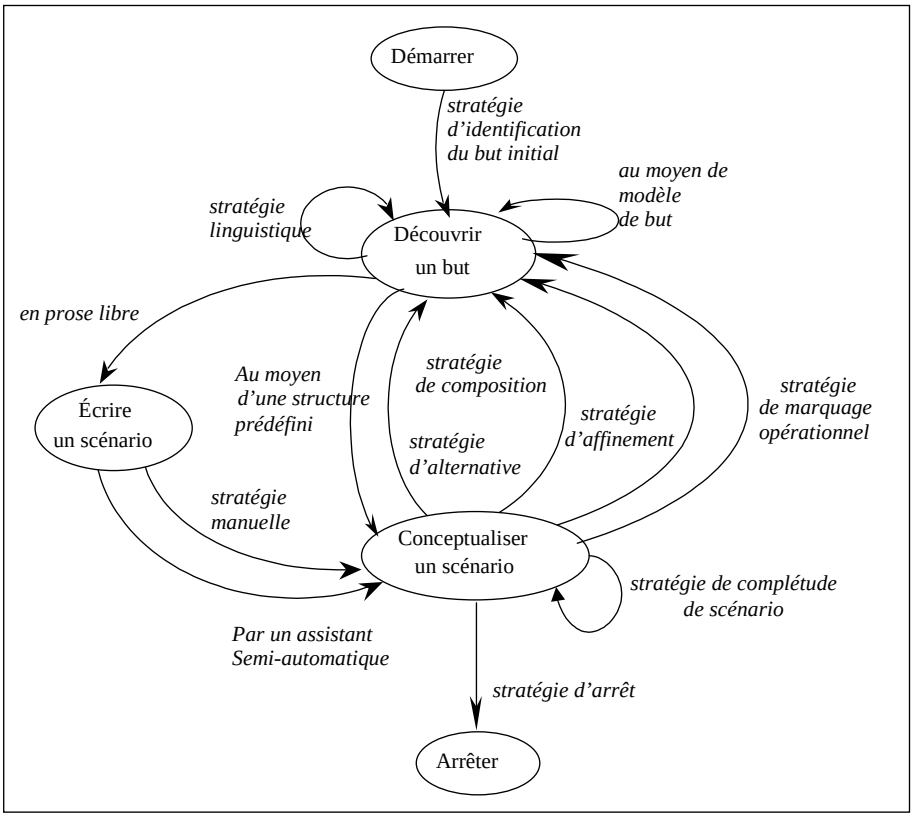


Figure 4.1 : La carte L'Ecritoire

Comme le montre la Figure 4.1 la carte L'Ecritoire comporte trois intentions en plus de *Démarrer* et *Arrêter* : *Découvrir un but, Ecrire un scénario et Conceptualiser un scénario*. Il y a au moins une et en général plusieurs stratégies préconisées par l'approche pour satisfaire les buts du processus. L'ensemble détermine 12 sections qui représentent les 12 tâches clés du processus d'ingénierie telles que l'approche L'Ecritoire les préconise. Chacune des sections a des directives méthodologiques attachées qui conseillent, guident sur la façon d'atteindre le but. On décrit les plus importantes dans la section suivante en les illustrant par l'exemple du DAB.

4.2 Les directives méthodologiques de L'Ecritoire

4.2.1 Section <Démarrer, stratégie d'identification du but initial, Découvrir un but>

Cette section de la carte démarre le processus. La directive associée suggère une façon de découvrir le premier but.

Directive méthodologique :

- Il est recommandé de démarrer le processus au niveau d'abstraction le plus haut, c'est-à-dire le niveau comportemental,
- Le but à écrire devrait définir l'objectif de l'organisation motivant la conception du système,
- Formuler le but en respectant le modèle de but (Verbe + paramètres),
- Lorsque le paramètre manière n'a pas de valeur spécifique, utiliser par défaut, l'expression 'dans le cas normal'.

Exemple :
Reprenons l'exemple de l'ingénierie des besoins d'un système GAB et admettons que la banque considère qu'installer un système DAB pour fournir à ses clients des services de retrait d'argent est un but organisationnel à atteindre impérativement.
En suivant la directive, l'AFB écrit le but suivant:

B1 :'Fournir des services de retrait d'argent à partir d'un GAB dans le cas normal'.

Ce but est de type comportemental. En formulant ce but, l'AFB atteint l'intention cible *'Découvrir but'*. Le but découvert est enregistré ensuite dans un nouveau fragment de besoin FB1.

4.2.2 Section <Découvrir un but, Découvrir un but, Stratégie linguistique>

Cette section peut être activée dès qu'un but a été découvert. Elle vise à s'assurer de la bonne formulation du but, à la corriger si nécessaire et éventuellement de la compléter.

Directive méthodologique :

- Identifier les valeurs des paramètres du but en utilisant le modèle de but,
- Détecter les paramètres manquants,
- Compléter la formulation du but en donnant des valeurs aux paramètres manquants utiles.

Exemple :
Considérons le but *B1 :*

'Fournir des services de retrait d'argent à partir d'un GAB dans le cas normal'.

L'AFB identifie les valeurs suivantes des paramètres:

Résultat : *des services de retrait d'argent*
Source : *un GAB*
Manière : *dans le cas normal.*

Les paramètres *Destination, Bénéficiaire, Moyen*, sont détectés manquants.
L'AFB décide d'ajouter au paramètre **Bénéficiaire** la valeur *'les clients de notre banque'* et dans la mesure où l'utilisateur doit être muni d'une carte, de donner au paramètre **Moyen** la valeur *'une carte bancaire'*.

Il en résulte la nouvelle formulation du but suivante :

B1: (Fournir)$_{Ver}$ *(des services de retrait d'argent)*$_{Rés}$ *aux (clients de notre banque)*$_{Bén}$ *à partir d'(un GAB)*$_{So}$ *au (moyen d'une carte bancaire)*$_{Moy}$ *d'(une manière normale)*$_{Man.}$

4.2.3 Section <Découvrir un but, Découvrir un but, par le modèle de but>

Cette section comme la précédente, peut être activée dès qu'un but a été découvert. Cependant la stratégie de cette section correspond à une autre tout autre préoccupation : chercher des expressions alternatives qui font sens du but.

Directive méthodologique :

- Identifier les valeurs des paramètres du but en utilisant le modèle de but,
- Identifier les valeurs alternatives possibles pour chacun des paramètres du but,
- Construire les buts par combinaison des différentes valeurs non exclusives,
- Sélectionner ceux qui semblent réalistes.

Exemple :
Reprenons le but B1 dans sa dernière formulation ci-dessus faisant apparaître les différentes valeurs des paramètres.
Supposons qu'un travail collectif conduise (étape2) au tableau ci-dessous donnant des valeurs alternatives considérées comme faisant sens .

Paramètre	Valeurs alternatives
Objet: des services de retrait d'argent	des prêts des chéquiers relevés historiques de solde des services de virement d'argent
Source: GAB	
Bénéficiaire: clients de notre banque	clients d'autres banques clients handicapés

Moyen: une carte bancaire	identification à base d ‘un code personnalisé identification à base d’une empreinte digitale
Manière: dans le cas normal	

Les combinaisons de ces valeurs conduisent à la génération des 45 buts alternatifs de B1 (1 verbe * 5 objets * 1 source * 3 bénéficiaires * 3 moyens* 1 manière) parmi lesquels les 9 qui ont retenu l’attention de l’équipe d’IB sont présentés dans l’encadré ci-dessous.

1.Fournir des services de retrait d’argent à partir d’un GAB aux clients d’autres banques avec une carte bancaire. 2. Fournir des services de retrait d’argent à partir d’un GAB aux clients handicapés avec une carte bancaire. 3. Fournir des services de retrait d’argent à partir d’un GAB aux clients d’autres banques avec une identification par un code personnalisé. 4. Fournir des services de prêts à partir d’un GAB aux clients de notre banque avec une identification par un code personnalisé. 5. Fournir des services de prêts à partir d’un GAB aux clients de notre banque avec identification à base d une empreinte digitale. 6. Fournir des services de prêts à partir d’un GAB aux clients d’autres banques avec une carte bancaire. 7. Fournir des chéquiers à partir d’un GAB aux clients de notre banque avec une carte bancaire, 8. Fournir des relevés historiques de solde à partir d’un GAB aux clients de notre banque avec une bancaire. 9. Fournir des services de virement d’argent à partir d’un GAB aux clients de notre banque avec une bancaire

L’examen de ces buts conduit l’équipe à faire des choix différents de l’option de départ : (a) on décide de passer de la solution DAB imaginée à l’origine à la solution GAB (Guichet Automatique Bancaire) ; (b) le service de retrait d’argent est étendu aux clients d’autres banques.

Les 4 buts finalement sélectionnés sont :

B1 : Fournir des services de retrait d’argent à partir d’un GAB aux clients d’autres banques avec une carte bancaire et
B7 : Fournir des chéquiers à partir d’un GAB aux clients de notre banque avec une carte bancaire,
B8 : Fournir des relevés historiques de solde à partir d’un GAB aux clients de notre banque avec une carte bancaire,
B 9 : Fournir des services de virement d’argent à partir d’un GAB aux clients de notre banque avec une carte bancaire.

L’application de la règle met l’AFB face à une sorte d’explosion combinatoire. Celle-ci a été compensée dans le logiciel L’Ecritoire par l’introduction (a) d’une priorité associée à chaque valeur de paramètre (forte, faible, moyenne) et (b) de critères d’exclusion mutuelle des valeurs de paramètres(par exemple, le service de fourniture de chéquier exclu les clients d’une autre banque). Par ailleurs, notre expérience sur le terrain nous a convaincu qu’il est essentiel d’aider les acteurs de l’ingénierie des besoins à générer des alternatives qu’ils n’imaginent pas facilement seuls. Les études empiriques menées pour évaluer l’approche L’Ecritoire confirment cette constatation : 92% de sujets doublent le nombre de buts découverts grâce à l’application de la directive ci-dessus. Sur l’ensemble des études, le nombre de choix pertinents a été multiplié par 8 grâce à la directive.

4.2.4 Section <Découvrir un but, Ecrire un scénario, en prose libre>

Cette section aide l’AFB a écrire un scénario textuel associé à un but découvert. Un scénario textuel est un texte en langage naturel (LN). Il n’y a pas de restrictions sur l’usage du LN mais des directives d’écriture qui sont de deux types :

- des directives de style,
- des directives de contenu.

Les premières sont des recommandations sur la forme de l'écriture tandis que les secondes sont relatives au contenu du texte, c'est à dire aux actions du scénario. On donne ci-dessous des exemples de directives de style (DS) et de contenu (DC).

Exemples de directives de style
DS1 : Eviter l'utilisation de références anaphoriques comme 'il', 'elle', 'lui' 'eux' ou 'leur', etc. Utiliser plutôt des termes explicites.

DS2 : Chaque action atomique doit être déclarée par une clause composée d'un verbe et des plusieurs paramètres. Ainsi, une action atomique doit être exprimée selon l'un des modèles de clause suivant :
<Agent> <Action> <Paramètre>
ex. : 'Le système vérifie la validité de la carte'
<Agent1> <Action> <Paramètre> à partir de <Agent2>
ex. : 'Le client choisit un service à partir de l'écran du GAB'
<Agent1> <Action> <Paramètre> à <Agent2>
ex. : 'le GAB affiche un message d'erreur au client'

DS3 : Afin d'éviter l'oubli des agents, utiliser la voie active.

DS4 : Les conditions de flux doivent être déclarées explicitement au moment où elles influencent le chemin d'actions. Une condition de flux s'exprime selon l'un des deux modèles suivants :
Si <condition> alors <Flux d'actions>,
Répéter <Flux d'actions> jusqu'à <condition>.

Exemples de directives de contenu
DC1: Un scénario doit décrire un seul chemin d'actions.

DC2 : Des scénarios alternatifs doivent être décrits séparément.

DC3 : L'enchaînement d'actions doit comprendre un ordonnancement séquentiel d'actions.

DC4 : Si le scénario contient des itérations, celles-ci doivent être restreintes par des conditions où le nombre d'itérations doit être bien précis. Pour cela, utiliser le modèle proposé par la directive de style DS4.

DC5 : Un scénario de service (niveau comportemental) doit décrire un flux de services exprimés par des interactions entre agents du système incluant le système lui-même. Eviter d'utiliser des flux de contrainte et de concurrence.

DC6 : Les actions de communication d'un scénario d'interaction (niveau fonctionnel) sont des demandes d'un agent externe au système ou des réponses du système à un agent externe.
ex: 'Le client insère une carte bancaire dans le GAB'.

DC7 : Les agents du niveau fonctionnel sont le système et ses utilisateurs.

DC8 : Les agents du niveau physique sont des objets du système et ou des objets d'interface.

DC9 : Les actions atomiques d'un scénario interne (niveau physique) sont des opérations effectuées par un objet du système sur un objet d'un autre système ou sur l'autre système lui-même.

DC10 : Les actions de communication d'un scénario interne (niveau physique) sont des demandes et des réponses échangées entre les objets internes et / ou avec des objets externes au système
ex: *le moteur du GAB envoie l'ordre d'éjecter la carte au lecteur de la carte.*

Les études empiriques conduites pour évaluer l'approche L'Ecritoire ont montré que les directives de style et de contenu améliorent sensiblement la qualité des scénarios écrits. La proportion des sujets décrivant correctement plus de la moitié d'un scénario est multipliée par 2 en utilisant les directives de contenu et de style.

Exemple:
Retenons un fragment du niveau fonctionnel. Il s'agit du fragment FB1.2 comprenant le scénario d'interaction SC1.2 couplé au but de service B1.2 pour former le fragment de besoin fonctionnel FB1.2 (la notation 1.2

signifie que le fragment résulte de l'affinement du fragment comportemental FB1 de but B1 identifié à l'étape précédente – voir 4.2.3) présenté ci-dessous :

B1.2 : *Retirer de l'argent du GAB au moyen d'une carte bancaire dans le cas normal*
SC1.2
Etats initiaux : *'le client a une carte', le GAB est en service'*
Actions : Le client insère une carte dans le GAB. Le GAB vérifie la validité de la carte. Si la carte est valide alors un message demandant le code est affiché par le GAB. Le client tape son code. *Si le code est valide* alors un message demandant le montant est affiché par le GAB au client. Le client tape le montant désiré. Si le montant est disponible, le GAB éjecte la carte au client. Si un reçu est demandé par le client, le GAB imprime un reçu au client et il délivre les billets au client.
États finaux : *'le client a une carte', 'le GAB est en service', 'le client a de l'argent', 'le client a un reçu''*

4.2.5 Section < Ecrire un scénario, Conceptualiser un scénario, par un assistant automatisé>

Cette section assure les tâches de conceptualisation d'un scénario textuel. Elle met en œuvre les algorithmes de l'approche linguistique qui permettent d'analyser le texte initial et de le transformer jusqu'à ce que son interprétation puisse se faire sans ambiguïté ; c'est à dire jusqu'à ce que toute phrase et clause du texte corresponde à des instances d'un ou plusieurs des patrons sémantiques de la bibliothèque de patrons de L'Ecritoire. On comprend qu'il soit difficilement envisageable d'exécuter ces algorithmes manuellement. Il faut au contraire se placer dans l'optique d'une interaction AFB-machine dans laquelle le logiciel exécute les algorithmes et demande l'intervention humaine pour corriger des erreurs qui l'empêchent de réaliser l'instanciation des patrons.

Etapes du processus sous-jacent à la section :
- Vérifier la terminologie du scénario par rapport au glossaire du projet,
- Clarifier et compléter le scénario à l'aide des patrons sémantiques,
- Conceptualiser le scénario.

Exemple:
Prenons le fragment de besoin FB1.2 introduit à la section précédente comme exemple d'illustration de la conceptualisation.

Etape 1 : L'Ecritoire gère un glossaire des termes du projet. Cette étape de la directive repose sur une fonctionnalité semi-automatique permettant de détecter les synonymes et de prendre des actions correctrices en conséquence. Si des synonymes sont détectés au sein du scénario, la directive suggère à l'AFB d'éliminer l'un ou l'autre des termes synonymes pour conserver le terme le plus pertinent. En revanche, si un terme du scénario est identifié comme un synonyme de terme du glossaire, la directive demande à l'AFB de remplacer ce terme par celui du glossaire.
L'utilisation de cette fonctionnalité pour SC1.2 conduit à la version révisée suivante du scénario (les termes corrigés sont en gras) :

États initiaux : *'le client a une **carte bancaire'**, 'le GAB est en service'.*
Actions : Le client insère une **carte bancaire** dans le GAB, le GAB vérifie la validité de la **carte bancaire.** Si la **carte bancaire** est valide alors, un message demandant le code est affiché au client. Si le code est valide alors, un message demandant le montant est affiché**.** Le client introduit le montant désiré**.** Si le montant est disponible, le GAB éjecte **la carte bancaire**. Le GAB propose un reçu au client Si un reçu est demandé par le client, le GAB imprime un reçu au client et il lui délivre **l'argent**.
États finaux : 'le client a une **carte bancaire'**, 'le GAB est en service', 'le client a de l'argent', 'le client a un reçu'.

On observe que l'AFB a remplacé 'billets' par 'argent' et 'carte' par 'carte bancaire'.

Etape 2 : l'application des algorithmes linguistiques conduit à une interprétation du texte de SC1.2 comme un ensemble d'instances de patrons telle que la Figure 4.2 l'indique.

États initiaux : **Possession** (avoir) [‘Agent : ‘le client’; Objet : ‘une carte bancaire’],
Etat (être)[Agent: ‘le GAB’ : Etat : ‘en service’].

Actions:
Séquence [Avant : **Communication** (‘insère’) [Agent : ‘le client‘; Objet : ‘une carte bancaire‘; Source : ‘le client’ ; Destination : ‘le GAB ‘] ; Après :
[**Séquence** : Avant : **Action interne** (‘vérifier’) [Agent : ‘le GAB’; Objet : ‘la validité de la carte bancaire’] ; Après :
Contrainte [**Condition:** ‘la carte est valide’,
Conditionné : **[Séquence** [Avant Communication (‘taper’) [Agent : ‘le client‘; Objet : ‘**son** code ‘; Source : ‘le client’ ; Destination : ‘**?**’] ; Après :
[Séquence[Avant : **Action interne** (‘vérifier’) [Agent : ‘le GAB’; Objet : ‘la validité de la carte bancaire’] ; Après
[**Séquence** [Avant : Communication (‘taper’) [Agent : ‘le client ‘; Objet : ‘montant désiré ‘; Source : ‘le client’ ; Destination : ‘**?**’] Après :
Contrainte [**Condition** : ‘le montant est disponible’,
Conditionné : [Séquence [Avant **Communication** (‘proposer’) [Agent : ‘le GAB’; Objet : ‘un reçu’; Source : ‘le GAB’ ; Destination : client’]: Après :
Contrainte [**Condition**: ‘un reçu est demandé par le client’,
Conditionné :[**Séquence [**Avant’ : **Communication** (‘imprimer’) [Agent : ‘le GAB’; Objet : ‘un reçu’; Source : ‘le GAB’ ; Destination : ‘**?**’] ; Après:
[Séquence[Avant : **Action interne** (‘éjecter’) [Agent : ‘le GAB’; Objet : ‘la carte bancaire’ Source : ‘le GAB’ ; Destination : client’] ; Après

Communication (‘délivrer’) [Agent : ‘**il**’; Objet : ‘l’argent; Source : ‘le GAB’ ; Destination : ‘**lui**’]]]]]]]]]]]]]]]]]]

États Finaux : **Possession** (avoir) [Agent : ‘le client’; Objet : ‘une carte bancaire’],
Etat (être)[Agent: ‘le GAB’ : Etat : ‘en service’],
Possession (avoir) [‘Agent : ‘le client’; Objet : ‘l’argent’],
Possession (avoir) [‘Agent : ‘le client’; Objet : ‘un reçu’]

Figure 4.2 : Instances de patrons du scénario SC1.2

La Figure 4.2 montre (a) que certains patrons n’ont pu être complètement instanciés : ceci est la marque d’une écriture incomplète du texte du scénario ; (b) le repère de références anaphoriques. L’AFB doit remplacer les référence anaphoriques par des termes du glossaire et compléter l’écriture des phrases incomplètes.
Le texte corrigé est le suivant :

États finaux : *‘le client a une carte bancaire’, ‘le GAB est en service’.*
Actions: Le client insère une carte bancaire dans le GAB, le GAB vérifie la validité de la carte. Si la carte est valide alors, un message demandant le code est **affiché par le GAB au client**. Le client introduit **le code personnel du client dans le GAB**. *Si le code est valide* alors un message demandant le montant est affiché **par le GAB au client**. Le client introduit le montant désiré **dans le GAB**. Si le montant est disponible, le GAB éjecte carte bancaire **au client**. Le GAB propose un reçu au client. Si un reçu est demandé par le client, le GAB imprime un reçu au client et **le GAB** délivre l’argent **au client**.
États finaux : *‘le client a une carte bancaire’, ‘le GAB est en service’, ‘le client a de l’argent’, ‘le client a un reçu’.*

Etape3 : Lorsque toutes les clauses et phrases du texte du scénario sont instanciées, les algorithmes linguistiques établissent la correspondance entre les éléments du texte et les concepts du modèle de scénario. On dit que le scénario est *conceptualisé*. Il est alors présenté à l'AFB dans une forme indentée qu'illustre la Figure 4.3 dans le cas de SC1.2.

États initiaux : 'le client a une carte bancaire', 'le GAB est en service',
Actions:
1. Le client insère une carte bancaire dans le GAB.
2. Le GAB vérifie la validité de la carte bancaire.
3. Si la carte bancaire est valide, alors
   4. Un message demandant le code est affiché par le GAB au client.
   5. Le client introduit le code personnel du client dans le GAB
   6. Si le code est valide, alors
      7.Un message demandant le montant est affiché par le GAB au client.
      8. Le client introduit le montant désiré dans le GAB.
      9. Si le montant est disponible, alors
         10. Le GAB propose un reçu au client
         11. Le GAB éjecte la carte bancaire au client
         12. Si un reçu est demandé par le client, alors
            13.Le GAB imprime un reçu au client
            14. Le GAB délivre l'argent au client

États finaux : 'le client a une carte bancaire', 'le GAB est en service',
'le client a de l'argent', 'le client a un reçu'.

Figure 4.3 : Scénario SC1.2 après conceptualisation

4.2.6 Section < Conceptualiser un scénario, Découvrir un but, par la stratégie d'affinement>

Les cinq sections considérées précédemment adressaient le problème de la *constitution d'un fragment de besoin*. Les trois suivantes mettent en œuvre des stratégies de raisonnement appliquées aux fragments constitués pour *découvrir de nouveaux besoins*. Plus précisément les 3 stratégies s'appuient sur l'analyse des scénarios conceptualisés pour découvrir de nouveaux buts.

La stratégie d'affinement qui correspond à cette section permet de découvrir des buts du niveau d'abstraction inférieur à celui du FB analysé. Par exemple si le FB est comportemental la stratégie d'affinement aboutira à la découverte de buts de service constitutifs de FBs fonctionnels. Cette stratégie repose sur l'idée qu'une action dans un scénario peut être considérée comme un but à un niveau d'abstraction inférieur. Cette idée est présente dans l'extension de l'approche des cas d'utilisation de Jacobson qu'en a faite Cockburn dans [Cockburn95]. Pour Cockburn un cas d'utilisation (use case) doit être attaché à un but (goal) déployé sous la forme d'une collection de scénarios. Chaque action d'un scénario peut à son tour, être vue comme un but associé à un nouveau cas d'utilisation de la collection des cas d'utilisation en cours de construction. Il y a donc une forme de récursivité illustrée par les liens croisés de la Figure 4.4.

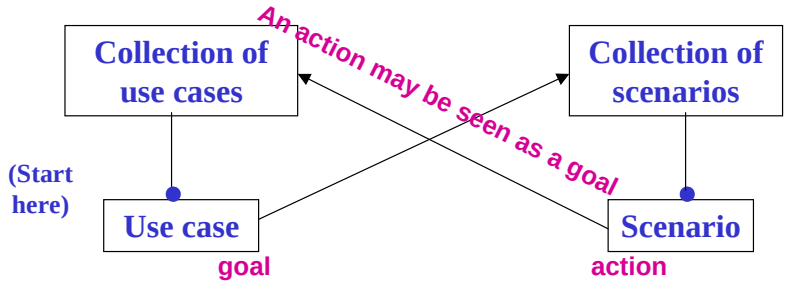


Figure 4.4 : Le lien récursif entre use case et scénario de Cockburn

Directive méthodologique :

- Considérer chaque action atomique d'un scénario Sc d'un fragment FBi et lui associer éventuellement un but,
- Lier chacun des buts découverts à Fbi par un lien '*Affiné par*'.

Exemple :
L'application de l'étape 1 de la directive conduit à découvrir 17 nouveaux buts qui sont affichés dans le tableau ci-dessous au regard de l'action du scénario SC1.2 qui a permis leur découverte.

Action Atomique	But découvert
1. Le client insère une carte bancaire dans le GAB.	B1.2.1 : (Insérer)$_{Ver}$ (une carte bancaire)$_{Obj}$ (dans le GAB)$_{Des}$ (dans le cas normal)$_{Man}$
2. Le GAB vérifie la validité de la carte bancaire.	B1.2.2 : (Vérifier)$_{Ver}$ (la validité de la carte bancaire.)$_{Rés}$ (dans le cas normal)$_{Man}$
4. Un message demandant le code est affiché par le GAB au client.	B1.2.3 : (Afficher)$_{Ver}$ (un message demandant le code)$_{Rés}$ au (client)$_{Des}$ (dans le cas normal)$_{Man}$
5. Le client introduit le code personnel du client dans le GAB.	B1.2.4 : (Introduire)$_{Ver}$ (le code personnel du client)$_{Rés}$ dans (le GAB)$_{Des}$ (dans le cas normal)$_{Man}$
6. Le GAB vérifie la validité du code.	B1.2.5 : (Vérifier)$_{Ver}$ (la validité du code)$_{Rés}$ (dans le cas normal)$_{Man}$
8. Un message demandant le montant est affiché par le GAB au client.	B1.2.6 : (Afficher)$_{Ver}$ (un message demandant le montant)$_{Rés}$ au (client)$_{Des}$ (dans le cas normal)$_{Man}$
9. Le client introduit le montant désiré dans le GAB.	B1.2.7 : (Introduire)$_{Ver}$ (le montant désiré)$_{Rés}$ dans (le GAB)$_{Des}$ (dans le cas normal)$_{Man}$
10. Le GAB vérifie la disponibilité du montant demandé.	B1.2.8 : (Vérifier)$_{Ver}$ (la validité du montant demandé)$_{Rés}$ (dans le cas normal)$_{Man}$
12. Le GAB propose un reçu au client.	B1.2.9 : (Proposer)$_{Ver}$ (un reçu)$_{Rés}$ au (client)$_{Des}$ (dans le cas normal)$_{Man}$
13. Le client introduit un choix de reçu dans le GAB.	B1.2.10 : (Introduire)v_{er} (un choix de reçu)$_{Rés}$ dans (le GAB)$_{Des}$ (dans le cas normal)$_{Man}$
14. Le GAB éjecte la carte au client.	B1.2.11: (Ejecter)v_{er} (la carte bancaire)$_{Rés}$ au (client)$_{Des}$ (dans le cas normal)$_{Man}$
16. Le GAB imprime un reçu au client.	B1.2.12 : (Imprimer)v_{er} (un reçu)$_{Rés}$ au (client)$_{Des}$ (dans le cas normal)$_{Man}$
17. Le GAB délivre l'argent au client.	B1.2.13 : (Délivrer)v_{er} (l'argent)$_{Rés}$ au (client)$_{Des}$ (dans le cas normal)$_{Man}$

Tous les buts découverts participent à des fragments de besoin liés à FB1.2 par des liens '*Affiné Par*'.

4.2.7 Section <Conceptualiser un scénario, Découvrir un but, par la stratégie d'alternative>

Cette stratégie guide l'AFB dans la découverte de buts alternatifs du but B du scénario Sc en cours d'analyse. Ils sont alternatifs dans le sens où ils identifient des comportements du système qui sont des variantes du comportement décrit dans le scénario Sc. En conséquence, les expressions de ces buts contiennent le même verbe et les mêmes paramètres que B mais s'en distinguent par des manières différentes. Les buts découverts participent à des fragments de besoin liés au fragment FBi (formé du couple <B,Sc>) par des liens '*OU*'.
La stratégie exploite l'idée que les négations des conditions de flux d'un scénario sont le point d'appui de la découverte des cas alternatifs.
Directive méthodologique :

- Extraire les imbrications de conditions de flux du scénario,
- Associer un cas à chacune des imbrications formées à partir des négations des conditions,
- Former un but correspondant à chacune des imbrications précédentes .

Exemple :
Appliquons la directive au scénario SC1.2. Il y a quatre conditions de flux suivants imbriquées comme suit :

1. *Si la carte est valide*
2. *Si le code est valide*
3. *Si le montant demandé est disponible*

4. *Si un reçu a été demandé par le client*

On en déduit 4 cas en considérant les imbrications formées à partir des négations des conditions.

1. *Si la carte est non valide*
2. *Si la carte est valide mais le code ne l'est pas*
3. *Si la carte est valide, si le codes est valide mais le montant demandé ne l'est pas*
4. *Si la carte est valide, si le codes est valide si le montant demandé est valide mais le client n'a pas demandé un reçu*

Chacun des quatre cas identifiés correspond à une situation que le système doit gérer. Il y a donc bien un besoin à exprimer sous la forme d'un but pour chacun des cas. Il s'en suit les quatre nouveaux buts suivants :

B1.2 [1] : (Retirer)$_{Ver}$ de (l'argent)$_{Obj}$ du (GAB)$_{So}$ au (moyen d'une carte bancaire)$_{Moy}$ avec (une carte non valide)$_{Man}$
B1.2.[2] : (Retirer)$_{Ver}$ de (l'argent)$_{Obj}$ du (GAB)$_{So}$ au (moyen d'un carte bancaire)$_{Moy}$ avec (une correction du code)$_{Man}$
B1.2.[3] : (Retirer)$_{Ver}$ de (l'argent)$_{Obj}$ du (GAB)$_{So}$ au (moyen d'un carte bancaire)$_{Moy}$ avec (une correction du montant)$_{Man}$
B1.2.[4]: (Retirer)$_{Ver}$ de (l'argent)$_{Obj}$ du (GAB)$_{So}$ au (moyen d'une carte bancaire)$_{Moy}$ (sans reçu)$_{Man}$

On peut observer que le comportement associé au but B1.2[1] est un comportement d'exception ; le scénario attaché sera 'exceptionnel' car son déroulement conduira à l'échec du but *racine 'Retirer de l'argent du GAB au moyen d'une carte bancaire'*. Au contraire les 3 buts suivants identifient un comportement normal qui est sont des variantes du comportement décrit dans le scénario SC1.2 et qui aboutissent toutes à la satisfaction du but *racine* .

4.2.8 Section <Conceptualiser un scénario, Découvrir un but, par la stratégie de composition>

Cette stratégie guide l'AFB dans la découverte de buts complémentaires du but B couplé au scénario Sc en cours d'analyse. Ces buts sont complémentaires de B dans le sens où B ne pourra être atteint de manière persistante que si ces buts le sont aussi. Par exemple, le but B1.2 ne peut être atteint que si le client a une carte de crédit, il est donc complémentaire au but '*Obtenir une carte de crédit de la banque'*. Les buts découverts participent à des fragments de besoin qui sont en relation '*ET*' avec le fragment Fbi (<B,Sc>). L'Ecritoire propose deux façons d'appliquer la stratégie de composition ; l'une et l'autre étant capturée dans une directive.

4.2.8.1 Directive de découverte basée sur l'exclusion des états initiaux et finaux

Cette directive exploite l'idée qu'un scénario ne peut se reproduire systématiquement que si les états initiaux sont vérifiés à la fin de son exécution ; c'est à dire si les états finaux incluent les états initiaux.

Directive méthodologique :

- Vérifier s'il y a inclusion des états initiaux dans les états finaux,
- Si l'inclusion n'est pas vérifiée, identifier l'obstacle à l'inclusion,
- Associer un ou plusieurs buts permettant de lever l'obstacle,
- Coupler le but (ou les buts) au fragment FBi par un lien 'ET'.

Exemple :
B :*'Retirer de l'argent du GAB au moyen d'une carte bancaire avec trois saisies incorrectes du code'*
Etats intiaux : *'le client a une carte bancaire', 'le GAB est en service'*

Actions :
1- le client insère une carte dans le GAB
2- le GAB vérifie la validité de la carte bancaire
3- si la carte est valide alors
  4- un message demandant le code est affiché au client par le GAB
  5- le client introduit le code personnel du client dans le GAB
  6- le GAB vérifie la validité du code
  7- si le code n'est pas valide alors
    8- Répéter
      9- Un message demandant le code personnel est affiché par le GAB au client
      10- Le client introduit un code personnel erroné dans le GAB

Deux fois
11 Le GAB garde la carte bancaire du client
12 Un message demandant au client de consulter l'agence bancaire est affiché au client par le GAB

Etats finaux : *'le GAB a la carte du client', 'le GAB est prêt à servir'.*

Dans le scénario d'échec précédent la condition d'inclusion des états initiaux dans les états finaux n'est pas satisfaite.
L'obstacle d'inclusion est que 'le client n'a plus sa carte bancaire'. Cet obstacle suggère un but de réparation tel que 'Restituer au client la carte bancaire absorbée par le GAB'. Le fragment associé à ce but est couplé au fragment FB1.2 par un lien *'ET'*.

4.2.8.2 Directive de découverte basée sur les actions de production/consommation de ressources

Cette directive aide à la découverte de buts par un raisonnement centré sur les ressources physiques. Plus précisément cette directive fait l'hypothèse que les actions de production de ressources et les actions de consommation de ressources doivent s'équilibrer dans l'ensemble des fonctionnalités du système.

Directive méthodologique :
- Identifier les ressources physiques du scénario Sc en cours d'analyse,
- Identifier les actions de production et les actions de consommation de ressources,
- Etablir les couples d'actions de production et de consommation pour chaque ressource et identifier les actions isolées,
- Proposer des buts assurant l'équilibre Production/Consommation pour chacune des actions isolées,
- Coupler les fragments découverts au fragment Fbi (<B,Sc>).

Exemple :
Considérons à nouveau le scénario SC1.2.
Il y a trois ressources physiques produites ou consommées par les actions du scénario : *la carte bancaire, l'argent et le reçu.*

Le scénario comporte une action de consommation (le point de vue pris est celui du système)
1. Le client insère une carte bancaire dans le GAB
et trois actions de production de ressources
11.Le GAB éjecte la carte bancaire au client
13.Le GAB imprime un reçu au client
14. Le GAB délivre l'argent au client.

L'interaction 1 se couple avec l'interaction 11 mais les deux autres interactions restantes sont isolées.
(1. Le client insère une carte bancaire dans le GAB, 11 .Le GAB éjecte la carte bancaire au client).
*(13.Le GAB imprime un reçu au client, **?**)*
*(14. Le GAB délivre l'argent au client, **?**)*

Il faut que le système établisse une contre partie à chacune des actions de production (13 et 14). Il va de soi que le GAB ne peut pas délivrer des billets sans que son magasin soit réapprovisionné. De même il faut que le GAB soit alimenté en papier pour qu'il puisse y avoir une production de reçu chaque fois que le client du scénario SC1.2 le souhaite. Ce raisonnement permet à l'AFB d'identifier deux nouveaux buts :
B1.5 :Emmagasiner de l'argent dans le GAB dans le cas normal.
B1.6: Alimenter le GAB en papiers dans le cas normal.

Les deux nouveaux fragments associés aux buts découverts sont couplés au fragment FB1.2 par un lien '*ET*'.

4.2.9 Section <Conceptualiser un scénario, Découvrir un but, par la stratégie de marquage opérationnel>

Cette dernière stratégie de découverte de buts par analyse de scénario vise à identifier des buts *opérationnels* (voir chapitre 2), c'est à dire des buts représentant des besoins qui peuvent être traduits sans ambiguïté en contraintes sur les objets physiques ou informationnels du système en développement. Il n'y a donc pas lieu d'écrire des scénarios pour ces buts. Ils correspondent aux feuilles de la hiérarchie des besoins. Ces buts font partie de fragments de besoin de type '*objectifié*' caractérisé par l'absence de scénario.

Marquer un but de type *'opérationnel'* dépend du contexte dans lequel se déroule le processus d'IB. Ainsi par exemple, le but *B1.2.1: (Insérer)$_{Ver}$ (une carte bancaire)$_{Obj}$ (dans le GAB)$_{Des}$ (dans le cas normal)$_{Man}$*
peut être marqué *opérationnel* dans un cas et *concret* dans un autre projet. Dans le premier cas, il se peut que la recherche des besoins soit centrée sur les besoins fonctionnels. Ce qui importe alors est d'identifier que *le client doit insérer sa carte bancaire dans le GAB.* On ne cherche pas à savoir commet s'effectue le traitement interne permettant de détecter que *la carte est insérée*. Dans le deuxième cas au contraire on doit aussi comprendre cet aspect des choses, identifier quels sont les objets internes du système concernés par l'insertion de la carte et surtout comprendre *comment* ces objets doivent interagir pour que le but soit réalisé. Ici, l'écriture du scénario associé au but aidera à faire le raisonnement et à identifier des besoins physiques de fine granularité.

Directive méthodologique :

- Scruter chacune des actions atomiques du scénario Sc et décider s'il faut lui associer un but opérationnel,
- Introduire un fragment de type 'objectifié' pour chaque but découvert et les lier au fragment Fbi (<B, Sc>) par un lien '*Affiné par*'.

Exemple :
Considérons le fragment FB1.2.5 suivant :

B1.2.5: (Vérifier)$_{Ver}$ (la validité du code)$_{Rés}$ (dans le cas normal)$_{Man}$	**SC1.2.2** **États initiaux**: 'la carte est dans le lecteur de la carte'. **Actions:** 1. le moteur du GAB demande au lecteur de carte de lire la date de validité et le numéro de la carte 2. le lecteur de carte envoie le numéro de la carte et la date de validité au moteur du GAB 3. le moteur du GAB demande la date courante à l'horloge du GAB 4..l'horloge du GAB retourne la date courante au moteur du GAB 5. si le numéro de la carte correspond au numéro composé et la date de validité de la carte n'est pas expirée alors 6. le moteur du GAB envoie le numéro de la carte au système informatique de la banque 7. le système informatique de la banque retourne l'état actuel du compte client au moteur du GAB 8. si le compte du client n 'est pas sur la liste rouge 9.. le moteur du GAB enregistre la validité de la carte. **États finaux**: 'la carte est dans le lecteur de la carte', 'la carte est valide'.

Le tableau ci-dessous résume les décisions prises par l'AFB à partir de l'analyse de chacune des actions atomiques du scénario physique ci-dessus.

Action Atomique	But opérationnel découvert
1. le moteur du GAB demande au lecteur de carte de lire la date de validité et le numéro de la carte	Bop1: (Demander)$_{Ver}$ au (lecteur de carte)$_{Des}$ de (lire la date de validité et le numéro de la carte)$_{Res}$ par (le moteur du GAB)$_{So}$ (dans le cas normal)$_{Man}$
2. le lecteur de carte envoie le numéro de la carte et la date de validité au moteur du GAB	Bop2 : (Envoyer)$_{Ver}$ (le numéro de la carte et la date de validité)$_{Obj}$ par (le lecteur de carte)$_{So}$ au (moteur du GAB)$_{Des}$ (dans le cas normal)$_{Man}$
3. le moteur du GAB demande la date courante à l'horloge du GAB	Bop3 : (Demander)$_{Ver}$ de (la date courante)$_{Res}$ par (le moteur du GAB)$_{So}$ à (l'horloge du GAB)$_{Des}$ (dans le cas normal)$_{Man}$
4. l'horloge du GAB retourne la date courante au moteur du GAB	Bop4 : (Retourner)$_{Ver}$ (la date courante)$_{Obj}$ par (l'horloge du GAB)$_{So}$ au (moteur du GAB)$_{Des}$ (dans le cas normal)$_{Man}$
6. le moteur du GAB envoie le numéro de la carte au système informatique de la banque	Bop5 : (Envoyer)$_{Ver}$ (le numéro de la carte)$_{Obj}$ par (moteur du GAB)$_{So}$ au (système informatique de la banque)$_{Des}$ (dans le cas normal)$_{Man}$
7. le système informatique de la banque retourne l'état actuel du compte utilisateur au moteur du GAB	Bop6 : (Retourner)$_{Ver}$ (le numéro de la carte)$_{Obj}$ par (système informatique de la banque)$_{So}$ au (moteur du GAB)$_{Des}$ (dans le cas normal)$_{Man}$
9. le moteur du GAB enregistre la validité de la carte.	Bop7 : (Enregistrer)$_{Ver}$ (la validité de la carte)$_{Res}$ par (moteur du GAB)$_{So}$ (dans le cas normal)$_{Man}$

On voit que l'AFB a identifié 7 buts opérationnels faisant partie de fragments de besoins de type 'objectifiés' liés au fragment FB1.2.2 par des liens '*Affiné par*'.

4.2.10 La hiérarchie résultante

On termine cette partie en montrant la hiérarchie qui a servi d'exemple au cours de cette section. La Figure 4.5 montre trois types de fragments : ceux dont l'état est '*(But 'Découvert', Scénario 'Conceptualisé')*' ceux qui ont l'état '*(But 'Découvert', 'Scénario Non Ecrit') et* les FBs objectifiés dont l'état est '*(But 'Découvert', Scénario 'Vide')*'.

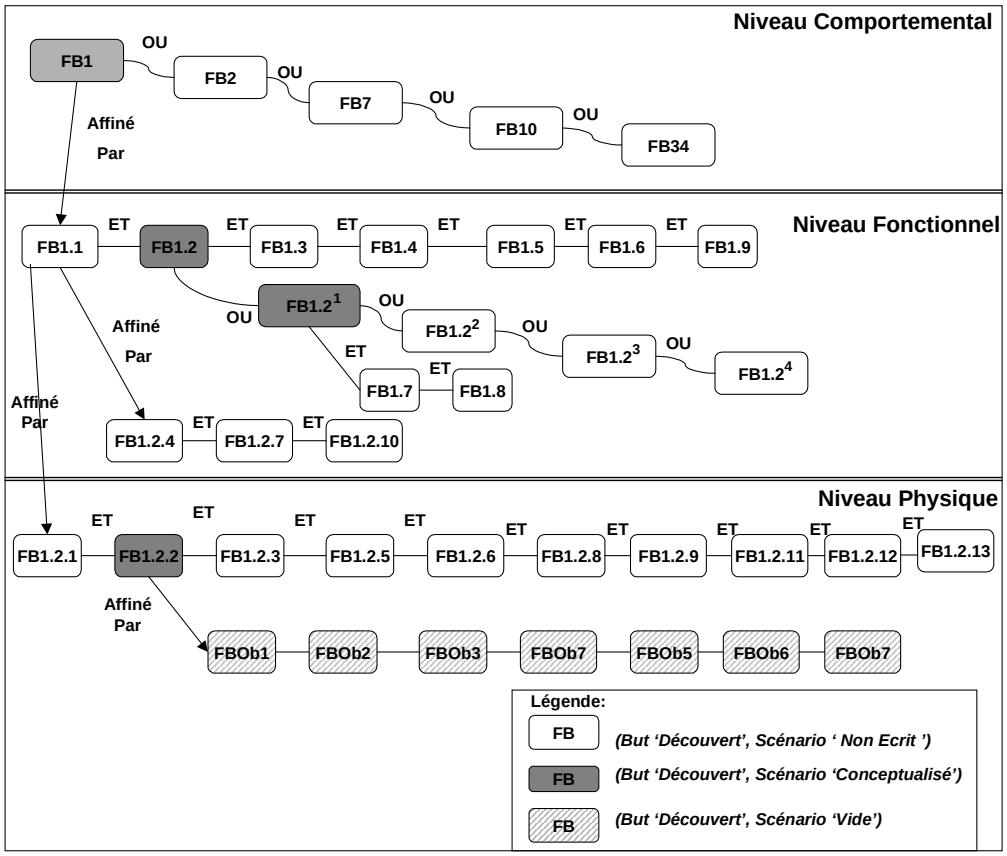


Figure 4.5 : La hiérarchie des FBs obtenue à partir des exemples présentés dans cette section

5. Conclusion

Dans cet article on a montré les enjeux du processus d'ingénierie des besoins et motivé des solutions qui se centrent sur le POURQUOI plutôt que sur le QUOI. Les approches dirigées par les buts et les approches par scénarios sont dans cette perspective et cherchent à déduire les besoins de l'étude du contexte organisationnel dans lequel le système va opérer.

L'approche *L'Ecritoire* présentée en détail dans cet article a la particularité de combiner une modélisation des buts à leur illustration par des scénarios. De cette façon, l'aspect abstrait des buts est compensé par la dimension concrète des scénarios. L'approche s'appuie sur (a) un modèle des besoins qui définit le produit de l'IB et (b) un modèle du processus de découverte et de spécification des besoins l'un et l'autre formellement définis. Le

processus est interactif et guidé par un ensemble des règles automatisées et exécutables dans l'environnement logiciel qui supporte l'approche.

Un article complémentaire de cette série illustrera sur un exemple, le déroulement de l'approche sous le contrôle et le guidage du logiciel *L'Ecritoire.*

6. Références

[Amoroso94] E.J. Amoroso, *Fundamentals of Computer Security*. Prentice-Hall, 1994.

[Anton94] A.I. Anton, W. M. Mc Cracken, C. Potts, *Goal decomposition and scenario analysis in business process reengineering.* Proceedings of the 6th International Conference CAiSE'94 on Advanced Information Systems Engineering, Utrecht, the Netherlands, Springer Verlag, pp. 94-104, 1994.

[Anton96] A.I. Anton, *Goal based requirements analysis*. Proceedings of the 2nd International Conference on Requirements Engineering ICRE'96, pp. 136-144, 1996.

[Anton97] A. Anton, *Goal Identification and Refinement in the Specification of Software-Based Information Systems.* PhD Thesis presented to acadamic faculty, Georgia Institute of Technology, June 1997. Available from http://www.csc.ncsu.edu/faculty/anton/pubs/thesis/thesis.html.

[Anton98] A. I Anton, C. Potts, *The use of goals to surface requirements for evolving systems.* International Conference on Software Engineering (ICSE '98) , Kyoto, Japan, pp. 157-166, 1998.

[Anton01] A. I. Anton, J.B. Earp, C. Potts, T.A. Alspaugh, *The role of policy and stakeholder privacy values in requirements engineering.* IEEE 5th International Symposium on Requirements Engineering (RE'01), Toronto, Canada, pp. 138-145, 27-31, 2001.

[Bell76] T.E. Bell, T.A. Thayer, *Software Requirements: Are They Really a Problem ?*. Proc. ICSE-2: 2nd Intrnational Conference on Software Enginering, San Francisco, 1976, 61-68.

[BenAchour99] C. Ben Achour, *Extraction des Besoins par Analyse des Scénarios Textuels.* Thèse du doctorat à l'Université de Paris 6, 1999.

[Benjamen99] A. Benjamen, *Une Approche Multi-démarches pour la modélisation des démarches méthodologiques*. Thèse du doctorat à l'Université de Paris 1, 1999.

[Benner93] K. M. Benner, M. S. Feather, W. L. Johnson, L. A. Zorman, *Utilizing Scenarios in the Software Development Process.* Information System Development Process, Elsevier Science Publisher B.V. (North-Holland), 117-134, 1993.

[Berzins91] V. Berzins, Luqi, *Software Engineering with Abstractions*. Addison-Wesley, 1991.

[Brooks87] F.P. Brooks, *No Silver Bullet: Essence and Accidents of Software Engineering*. IEEE Computer, Vol. 20 No. 4, pp. 10-19, 1987.

[Bubenko94] J. Bubenko, C. Rolland, P. Loucopoulos, V De Antonellis, *Facilitating 'fuzzy to formal' requirements modelling*. IEEE 1st Conference on Requirements Enginering, ICRE'94 pp. 154-158, 1994.

[Caroll95] J. M. Caroll, *The Scenario Perspective on System Development*. In Scenario-Based Design : Envisioning Work and Technology in System Development, Ed J.M. Carroll, 1995.

[Chung00] K. L. Chung, B. A. Nixon, E. Yu, J. Mylopoulos, *Non-Functional Requirements in Software Engineering*. Kluwer Academic Publishers, 2000.

[Cockburn95] A. Cockburn, *Structuring use cases with goals*. Technical report. Human and Technology, 7691 Dell Rd, Salt Lake City, UT 84121, HaT.TR.95.1, http: //members.aol.com/acocburn/papers/usecases.htm, 1995.

[CREWS98] CREWS Team, *The CREWS glossary.* CREWS report 98-1, http://sunsite.informatik.rwth-aachen.de/CREWS/reports.htm.

[CREWS99] CREWS (ESPRIT N°21.903), Cooperative Requirements Engineering With Scenarios, http://SunSITE.Informatik.RWTH-Aachen.DE/CREWS.

[Dano97] B. Dano, H. Briand, F. Barbier, *A use case driven requirements engineering process*. Third IEEE International Symposium On Requirements Engineering RE'97, Antapolis, Maryland, IEEE Computer Society Press, 1997.

[Dardenne93] A. Dardenne, A. van Lamsweerde, S. Fickas, *Goal directed requirements acquisition.* Science of Computer Programming, 20(1-2), pp.3-50, 1993.

[Davis93] A.M. Davis, *Software requirements :objects, functions and states*. Prentice Hall, 1993.

[Dik89] S.C. Dik, *The theory of functional grammar, part I : the structure of the clause*. Functional Grammar Series, Fories Publications, 1989.

[Dubois98] E. Dubois, E. Yu, M. Pettot, *From early to late formal requirements: a process-control case study*. Proc. IWSSD'98 – 9th International Workshop on software Specification and design. Isobe.IEEE CS Press, 34-42, 1998.

[Easterbrook94] S. Easterbrook, *Resolving Requirements Conflicts with Computer-Supported Negotiation*. In Requirements Engineering : Social and Technical Issues, M. Jirotka and J. Goguen (Eds.), Academic Press, 41-65, 1994.

[ELEKTRA97] ELEKTRA consortium, *Electrical Enterprise Knowledge for Transforming Applications - Athena deliverable : initial requirements for PPC*. ELEKTRA Project Internal Report, 1997.

[Erickson95] T. Erickson, *Notes on Design Practice: Stories and Prototypes as Catalysts for Communication.* In Scenario-Based Design: Envisioning Work and Technology in System Development, Ed J.M. Carroll, 1995.

[ESI96] European Software Institute, *European User survey analysis.*Report USV_EUR 2.1, ESPITI Project, 1996.

[Fillmore68] C. Fillmore, *The case for case*. In "Universals in linguistic theory", Holt, Rinehart and Winston (eds.),Bach & Harms Publishing Company, pp. 1-90, 1968.

[Finkelstein90] A. Finkelstein, J. Kramer, M. Goedicke, *ViewPoint Oriented Software Development.* Proc. Conf « Le Génie Logiciel et ses Applications », Toulouse, p 337-351, 1990.

[Firesmith94] D. G. Firesmith, *Modelling the dynamic Behaviour of Systems, Mechanisms, and Classes with Scenarios.* In Software DevCon '94, pages 73-82. SIGS Publications, NY, 1994.

[Glinz95] M. Glinz, *An integrated formal Model of Scenarios based on Statecharts.* Lecture Notes in Computer Science '95, pages 254-271, 1995.

[Gotel94] O. Gotel, A. Finkelstein, *Modelling the contribution structure underlying requirements.* In Proc. First Int. Workshop on Requirements Engineering : Foundation of Software Quality, Utrech, Netherlands, 1994.

[Harel87] D. Harel, *Statecharts: a Visual Formalism for Complex Systems.* Sci. Computer Program, 8, pp. 231-274, 1987.

[Haumer98] Peter Haumer, Klaus Pohl, Klaus Weidenhaupt, *Requirements Elicitation and Validation with Real World Scenes*. To appear in IEEE Transactions on Software Engineering, Vol. 24, No. 12, Special Issue on Scenario Management, 1998.

[Holbrook90] C. H. Holbrook, *A scenario - based methodology for conducting requirements elicitation*. ACM SIGSOFT, Software Engineering Notes Vol. 15, N° 1, pp. 95-104, 1990.

[Hsia94] P. Hsia, J. Samuel, J. Gao, D. Kung, Y. Toyoshima, C. Chen, *Formal Approach to Scenario Analysis.* IEEE Software, pp. 33-41, 1994.

[Jackson95] M. Jackson, *Software Requirements & Specifications – A Lexicon of Practice, Principles and Pejudices*. ACM Press, Addison-Wesley, 1995.

[Jacobson92] I. Jacobson, M. Christerson, P. Jonsson, G. Oevergaard, *Object Oriented Software Engineering: a Use Case Driven Approach.* Addison-Wesley, 1992.

[Jacobson95] I. Jacobson, *The use case construct in object-oriented software Engineering.* In ''Scenario-based design: envisioning work and technology in system development'', John M. Carroll (ed.), John Wiley and Sons, 309-336, 1995.

[Kaindl00] H. Kaindl, *A design process based on a model combining scenarios with goals and functions.* IEEE Trans. on Systems, Man and Cybernetic, Vol. 30 No. 5, 537-551, 2000.

[Lamsweerde98] A. V. Lamsweerde, L. Willemet, *Inferring declarative requirements specifications from operational scenarios*. In: IEEE Transactions on Software Engineering, Special Issue on Scenario Management. Vol. 24, No. 12, 1089-1114 , 1998.

[Lamsweerde00] A. V. Lamsweerde, *Requirements engineering in the year 00: A research perspective*. In Proceedings 22nd International Conference on Software Engineering, Invited Paper, ACM Press, June 2000.

[Leite97] J. C. .S. Leite, G. Rossi, F. Balaguer, A. Maiorana, G. Kaplan, G. Hadad, A. Oliveros, *Enhancing a requirements baseline with scenarios*. In Third IEEE International Symposium On Requirements Engineering RE'97, Antapolis, Maryland, IEEE Computer Society Press, pp. 44-53, 1997.

[Leveson95] N. Leveson, *Safeware - System Safety and Computers*. Addison-Wesley, 1995.

[Loucopoulos94] P. Loucopoulos, *The f^3 (from fuzzy to formal) view on requirements engineering*. Ingénierie des systèmes d'information, Vol. 2 N° 6, pp. 639-655, 1994.

[Loucopoulos97] P. Loucopoulos, V. Kavakli, N. Prakas, *Using the EKD approach, the modelling component.* ELEKTRA project internal report, 1997.

[Lubars93] M. Lubars, C. Potts, Charles Richter, *A Review of the State of Practice in Requirements Modeling*. Proceedings of the IEEE International Symposium on Requirements Engineering, RE'93, San Diego, USA, pp.2-14, 1993.

[Lutz93] R.R. Lutz, *Analyzing Software Requirements Errors in Safety-Critical, Embedded Systems*. Proceedings RE'93 – First International Symposium on Requirements Engineering, San Diego, IEEE, 1993, 126-133.

[McGraw97] K. Mc Graw, K. Harbison, *User Centered Requirements, The Scenario-Based Engineering Process*. Lawrence Erlbaum Associates Publishers, 1997.

[Mostov85] J. Mostow, *Towards better models of the design process*. AI Magazine, Vol. 6, pp. 44-57, 1985.

[Munford81] E. Munford, *Participative Systems Design: Structure and Method.* Systems, Objectives, Solutions, Vol. 1, North-Holland, 1981, 5-19.

[Mylopoulos92] J. Mylopoulos, K.L. Chung, B.A. Nixon, *Representing and using non- functional requirements: a process-oriented approach*. IEEE Transactions on Software Engineering, Special Issue on Knowledge Representation and Reasoning in Software Development, Vol. 18, N° 6, pp. 483-497, 1992.

[Mylopoulos99] J. Mylopoulos, K..L. Chung, E. Yu, *From object-oriented to goal6oriented requirements analysis*. Communications of the ACM. Vol 42 N° 1, 31-37, 1999.

[Nardi92] B. A. Nardi, *The Use of Scenarios in Design.* SIGCHI Bulletin, 24(4).

[Nilsson71] N.J. Nilsson, *Problem Solving Methods in Artificial Intelligence*. McGraw Hill, 1971.

[Nix93] B. A. Nixon, *Dealing with Performance Requirements During the Development of Information Systems.* Proc. RE'93 - 1st Intl. IEEE Symp. on Requirements Engineering, 42-49, 1993.

[Nuseibeh94] B. Nuseibeh, J. Kramer, A. Finkelstein, *A framework for expressing the relationships between multiple views in requirements specification*. In IEEE Transactions on Software Engineering, volume 20, pages 760-- 773. IEEE CS Press, 1994.

[Potts94] C. Potts, K. Takahashi, A.I. Anton, *Inquiry-based requirements analysis*. In IEEE Software 11(2), pp. 21-32, 1994.

[Potts95] C. Potts, *Using schematic scenarios to understand user needs.* Proc. DIS'95 - ACM Symposium on Designing interactive Systems : Processes, Practices and Techniques, University of Michigan, 1995.

[Potts97] C. Potts, *Fitness for use : the system quality that matters most.* Proceedings of the Third International Workshop on Requirements Engineering: Foundations of Software Quality REFSQ'9, Barcelona, pp. 15-28, 1997.

[Prat97] N. Prat, *Goal formalisation and classification for requirements engineering*. Proceedings of the Third International Workshop on Requirements Engineering: Foundations of Software Quality REFSQ'97, Barcelona, pp. 145-156, 1997.

[Ramesh95] B. Ramesh, T. Powers, C. Stubbs, M. Edwards, *Implementing requirements traceability : a case study*. In Proceedings of the 2nd Symposium on Requirements Engineering (RE'95), pp89-95, UK, 1995.

[Rawsthorne96] D. A. Rawsthorne, *Capturing functional Requirements through Object Interactions.* In Proceedings of ICRE '96, pages 60-67. IEEE, 1996.

[Robinson89] W. N. Robinson, *Integrating multiple specifications using domain goals.* Proc. IWSSD-5 - 5th Intl. Workshop on Software Specification and Design, IEEE, 219-225, 1989.

[Rolland97] C. Rolland, C. Ben Achour, *Guiding the construction of textual use case specifications*. Accepted to Data and Knowledge Engineering Journal, 1997.

[Rolland98a] C. Rolland, C. Souveyet, C. Ben Achour, *Guiding goal modelling using scenarios*. IEEE Transactions on Software Engineering, Special Issue on Scenario Management, Vol. 24, No. 12, Dec. 1998**.**

[Rolland98b] C. Rolland, C. Ben Achour, C. Cauvet, J. Ralyté, A. Sutcliffe, N.A.M. Maiden, M. Jarke, P. Haumer, K. Pohl, Dubois, P. Heymans, *A Proposal for a Scenario Classification Framework.* Requirements Engineering Journal, Vol. 3, No. 1, pp. 23-47, 1998.

[Rolland99] C. Rolland, N. Prakash, A. Benjamen, *A Multi-model View of Process Modelling*. The Requirements Engineering Journal, p. 169-187, 1999.

[Rolland99] C. Rolland, G. Grosz, R. Kla, *Experience With Goal-Scenario Coupling In Requirements Engineering*. Fourth IEEE International Symposium on Requirements Engineering (RE'99), University of Limerick, Ireland, 1999.

[Ross77] D. T. Ross, K. E. Schoman., *Structured analysis for requirements definition*. IEEE Transactions on Software Engineering , vol. 3, N° 1, 6-15, 1977.

[Rubin92] K.S. Rubin, A.Golberg, *Object Behavior Analysis*, Communications of the ACM, 35(9), Sept 1992, pp 48-62.

[Rumbaugh91] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, W. Lorensen, *Object-oriented modelling and design*. Prentice Hall, 1991.

[Some96] S. Some, R. Dssouli, J. Vaucher, *Toward an Automation of Requirements Engineering using Scenarios.* Journal of Computing and Information, Special issue: ICCI'96, 8th International Conference of Computing and Information, Waterloo, Canada, 2(1) pp 1110-1132, 1996.

[Sommerville97] I. Sommerville, P. Sawyer, *Requirements engineering*. Worldwide Series in Computer Science, Wiley, 1997.

[Standish95] The Standish Group, *Chaos*. Standish Group Internal Report, http://www.standishgroup.com/chaos.html, 1995.

[Sutcliffe98] A.G. Sutcliffe, N. A. Maiden, S. Minocha, D. Manuel, *Supporting scenario-based requirements engineering.* IEEE Trans. Software Eng. vol. 24, no. 12, 1072-1088, 1998.

[Weidenhaupt98] K. Weidenhaupt, K. Pohl, M. Jarke, P. Haumer, *Scenario usage in system development : a report on current practice*. IEEE Software, March 1998.

[Wood94] D.P. Wood, M. G. Christel, S. M. Stevens, *A Multimedia Approach to Requirements Capture and Modelling.* Proceedings. ICRE'94, Colorado Springs, 1994.

[Wright 92] P. Wright, *What's in a Scenario*. SIGCHI Bulletin, Volume 24, Number 4, 1992.

[Young87] M. R. Young, P. B. Barnard, *The Use of Scenarios in Human-Computer Interaction Research: Turbocharging the Tortoise of Cumulative Science.* CHI + GI 87 Human Factors in Computing Systems and Graphics Interface, Toronto, 1987.

[Yu94] E. Yu, *Modelling strategic relationships for process reengineering*. Ph.D. Thesis, Dept. Computer Science, University of Toronto, 1994.

[Yue87] K. Yue, *What does it mean to say that a specification is complete?*. Proc. IWSSD-4. Four International Workshop on Software Specification and Design, Monterrey, 1987.

[Zave97] P. Zave, M. Jackson., *Four dark corners of requirements engineering.* ACM Transactions on Software Engineering and Methodology, 1-30. 1997