

YVES CONSTANTINIDIS  
Préface de Michel Volle



# Expression des besoins pour le système d'information

Guide d'élaboration  
du cahier des charges

EYROLLES



# Préface

D'après le Standish Group, les projets informatiques connaissent un taux d'échec qui ne saurait être toléré dans aucun autre domaine de l'ingénierie : de ses enquêtes, on peut retenir qu'en gros 25 % des projets échouent, 50 % aboutissent avec un délai et un coût très supérieurs à la prévision, 25 % seulement sont convenablement réussis. En cas d'échec, on entend souvent dire « c'est la faute de l'informatique », mais en fait, il convient de dire que, par principe, *c'est toujours la faute du métier* que le produit informatique devait outiller (maîtrise d'ouvrage, MOA). Certes, il arrive que le réalisateur du produit (maîtrise d'œuvre, MOE) soit défaillant, mais alors la MOA aurait dû prendre en temps utile des mesures pour redresser la situation. Souvent d'ailleurs, le seul tort de la MOE sera d'avoir accepté un contrat impossible, car l'ingénierie des besoins a été défaillante et le projet était donc condamné dès le départ : la MOA s'est lancée dans le projet sans savoir ce qu'elle voulait *faire*, sans avoir exprimé ses priorités ni levé les ambiguïtés du vocabulaire, puis par la suite elle a été versatile, etc. Une expression de besoin bien faite garantit le succès ou du moins (car on ne peut jamais se prémunir contre toutes les surprises) une probabilité de succès de l'ordre de 90 % : on mesure l'enjeu si l'on compare ce taux aux données du Standish Group.

Jean-Pierre Meinadier est l'un des maîtres français les plus respectés en ingénierie des systèmes industriels<sup>1</sup>. Invité à participer à un projet de système d'information (SI), il fut immédiatement congédié pour avoir posé une question jugée incongrue : « qui est responsable ? ». Il a alors compris que contrairement à un projet d'automobile ou d'avion, dont les enjeux techniques sont mesurables et « froids », un SI est « chaud » : chaque projet comporte implicitement de tels enjeux « politiques » de légitimité, de découpage des pouvoirs et de responsabilité, que l'entreprise préfère souvent laisser implicites les données nécessaires à son succès. C'est cette « chaleur » du SI qui explique le taux d'échec extravagant des projets. Les acteurs ne manquent ni de bon sens ni d'intelligence, mais ils les mettent au service de priorités qui ne sont pas celles de l'efficacité de la production de l'entreprise ni de la qualité de ses produits – objectifs qui, en revanche, sont précisément ceux que sert un SI.

---

1. Jean-Pierre Meinadier, *Ingénierie et intégration des systèmes*, Hermes, 1998.

---

L'enjeu de l'ingénierie des besoins n'est donc pas purement technique : il s'agit de promouvoir des priorités (efficacité, qualité) qui sans doute devraient être celles de toute entreprise, mais à qui s'opposent d'autres formulations de sa mission (produire « de l'argent » ou « de la valeur pour l'actionnaire ») ainsi que les intérêts des corporations professionnelles.

Pour pouvoir agir dans les sables mouvants de la sociologie et de l'idéologie, il faut des idées claires et des principes fermes : c'est ce que nous propose ici Yves Constantinidis. On peut en effet désamorcer les pièges de la « politique » si l'on sait s'y prendre à temps, dans l'étape préliminaire et relativement « froide » de l'expression de besoins, si l'on se donne la peine d'éliminer les défauts du vocabulaire et les malentendus qu'ils provoquent, si l'on s'est enquis de la pertinence des besoins et priorités, si l'on a obtenu les validations qui, étant *authentiques*, scellent l'accord ultérieur des dirigeants et leur soutien au projet.

Le pivot de cette affaire, c'est la personne morale que Constantinidis nomme « assistance à maîtrise d'ouvrage » et que je préfère nommer « maîtrise d'ouvrage déléguée » (MOAD), car elle a reçu du directeur, stratège et responsable suprême de la MOA qu'il engage par sa signature, délégation du soin de l'expertise en SI. La MOAD a trois interlocuteurs : les stratèges (celui du métier et le DG, stratège de l'entreprise), les utilisateurs (qui se subdivisent en plusieurs catégories) et l'informatique (c'est-à-dire la MOE qui peut être, selon les cas, soit la DSI de l'entreprise, soit un fournisseur). Elle doit savoir parler les langages de ces divers interlocuteurs et assurer entre eux la fonction d'interprète. La MOAD est donc une vraie spécialité, et celui qui a exercé cette fonction pendant quelques années acquiert une connaissance approfondie de l'entreprise et du métier pour lesquels il travaille, y compris dans leurs dimensions « politiques » et symboliques qu'il sait gérer avec souplesse et discrétion. Malheureusement, les entreprises n'ont pas encore toutes compris la nécessité de cette fonction, ni perçu les compétences qu'elle exige : sa dénomination change d'une entreprise à l'autre, et cela montre bien la perplexité des DRH.

L'outil fondamental de la MOAD est un *modèle*, une mise en forme qui concrétise l'expression de besoin en schématisant le métier tel qu'il fonctionnera une fois outillé par le SI. Tout comme une base de données, ce modèle est un être que personne ne peut voir en entier, mais qui présente à chaque catégorie d'acteurs la *vue* qui lui convient : à l'informaticien, le diagramme de classe, étape initiale de la programmation dans un langage à objets (et quelques autres diagrammes) ; au stratège, le diagramme d'activité qui décrit le processus. Pour les utilisateurs, la vue pourra être audiovisuelle : en plaçant sur l'intranet un dessin animé complété par une documentation et un outil d'autoformation, on aide chacun à percevoir la finalité du système et l'étendue de sa responsabilité personnelle,



on élucide le processus de production dans lequel il intervient. Le cahier des charges est, sur ce modèle, la vue qui précise et récapitule toutes les exigences auxquelles le produit doit satisfaire, qu'elles soient propres au métier ou à la plate-forme technique de l'informatique. Ce document peut prendre une forme contractuelle, mais mieux vaut le concevoir comme une étape nécessaire de la coopération entre la MOE et la MOAD.

Pour prendre une métaphore dans la vie courante, supposons que vous fassiez construire une maison. Vous avez établi son plan avec l'aide d'un architecte et dites : « dans cette pièce, il faudra quatre prises de courant et une applique commandée par un interrupteur ». Ce sont là des *spécifications générales*, produites par la MOAD avec le conseil de la MOE et validées par le stratège. Puis la MOE demande à la MOA de dire où installer les prises, les interrupteurs et l'applique. Marquer ces emplacements sur le plan, c'est établir les *spécifications détaillées*, produites par la MOE et validées par la MOAD. Enfin, la MOE fera le plan de câblage qui précise le parcours des goulottes et saignées : ce sont des *spécifications techniques* qui n'intéressent pas la MOA, mais qui sont indispensables pour passer à la réalisation.

Il importe de respecter cette progression : comme le dit Donald Knuth, « l'optimisation prématurée est la racine de tous les maux » et certains projets s'enlisent parce que l'on se préoccupe, dès une phase initiale, de choix techniques qui devraient intervenir plus tard. Il faut, nous l'avons dit, que les validations soient *authentiques* : chacun des responsables doit pouvoir *comprendre* les documents qu'on lui soumet, et se savoir *engagé* par sa signature. Il ne convient pas, par exemple, de soumettre le diagramme de classe à un stratège, car cette vue sur le modèle n'est pas faite pour lui : comme il ne la comprendra pas, sa signature sera *passive* et il n'hésitera pas par la suite à remettre le modèle en question alors que les travaux sont déjà engagés. La MOAD doit donc lui présenter à l'appui du diagramme d'activité une synthèse en français, claire et en quelques pages, qui indique ce qu'il s'agit de *faire*, *comment* on envisage de le faire et *pourquoi* il ne convient pas de le faire autrement. Il sera également utile de lui présenter le processus sous la forme d'un dessin animé, même si certains stratèges prennent cela comme une atteinte à leur « sérieux ». Entre le stratège et la MOAD, le rapport est celui qui existe entre le décideur et l'expert. La décision revient au stratège qui, ayant la vue d'ensemble, peut tenir compte de contraintes et d'opportunités que l'expert ignore ; mais le stratège doit écouter l'expert qui, mieux que lui, se tient au courant de l'état de l'art des SI.

Il ne suffit pas de réussir des projets : il faut encore que l'informatique soit bien utilisée. La MOAD a donc avec les utilisateurs une relation continue : elle les forme, observe leur relation avec le poste de travail, promeut les bonnes pratiques et combat les mauvaises. Pour la MOE,

la MOAD doit être un *client compétent* qui sait ce qu'il veut et qui connaît assez l'informatique pour en comprendre les contraintes, mais qui se garde d'imposer des choix techniques. Enfin, et même si le cahier des charges précise les obligations de chacun, la relation entre la MOAD et la MOE doit être plus partenariale que contractuelle. Il convient qu'elles travaillent en tandem dès le début du projet, la responsabilité du travail passant de l'une à l'autre selon l'étape considérée.

Certains SI sont étonnamment bien réussis. Lorsqu'on interroge les utilisateurs, ils disent « on sait ce qu'on a à faire », « le travail est clair », « l'entreprise est bien dirigée », « on est bien outillés », etc. Et si l'on s'enquiert de la cause de cette réussite, on reçoit *toujours* la même réponse : « le DG (ou le directeur) s'est impliqué personnellement, il a mis le poids de son autorité dans la balance, il a réglé les problèmes politiques ». Un SI n'est pas en effet seulement une affaire d'informatique : sa conception inclut la définition du travail humain, l'organisation du processus de production et de son contrôle. C'est pourquoi il faut considérer que la responsabilité globale du SI appartient à la MOA, personne morale, et nommément à son directeur, personne physique qui engage la MOA par sa signature. Nos entreprises auront fait un grand pas lorsqu'elles sauront qu'un directeur ou un DG qui dit « c'est la faute de l'informatique » révèle une incompétence...

Michel VOLLE,  
*président d'honneur du Club des Maîtres d'Ouvrage  
des Systèmes d'Information*

# Remerciements

Je tiens à remercier les confrères qui m'ont apporté leurs remarques et leurs idées, en particulier Dominique Lepère, Benjamin Lemoine, Josselin Coupard et Jean-François Goglin. Merci à tous mes clients, maîtres d'œuvre, maîtres d'ouvrage et utilisateurs qui, en se prêtant à des interviews ou en participant à des groupes de travail, m'ont permis d'améliorer ma démarche. Merci à Suzanne Robertson et Karl Wieggers, qui m'ont encouragé lors de l'écriture de cet ouvrage. Merci à Michel Volle, qui en a écrit la préface. Merci à Nicolas, Mélina et Karin pour leurs idées fraîches et leur soutien moral.



# Table des matières

Guide de lecture .....	1
Quiz : êtes-vous prêt ? .....	5
Réponses au quiz .....	7

## Partie 1 – Méthodologie

Chapitre 1 – La méthode en action .....	11
Un exemple imaginaire, mais concret .....	11
Le point sur les concepts .....	18
Chapitre 2 – Les enjeux d’une bonne définition des besoins .....	21
L’utilité d’un cahier des charges .....	21
Un investissement très rentable .....	21
Les gains « cachés » .....	23
Les difficultés .....	23
Les risques .....	24
Améliorer les pratiques d’élaboration .....	26
Chapitre 3 – Compétences et savoir-faire .....	27
Le savoir .....	27
Le savoir-faire .....	29
Le savoir-être .....	30
Chapitre 4 – Exigences et cycle de vie du logiciel .....	35
Le cycle de vie du logiciel .....	35
Maîtres d’ouvrage et maîtres d’œuvre .....	36
Bâtisseurs et exploitants .....	38
La phase d’exigences .....	38
Les engagements réciproques .....	39

Chapitre 5 – La démarche .....	41
Décrire, documenter, communiquer .....	41
Les différents niveaux d'exigences .....	42
Les étapes de l'élaboration .....	43
Description formelle du processus global .....	44
Le processus en pratique .....	45
Check-list .....	47
Chapitre 6 – Définir le concept et les objectifs .....	49
Une activité préalable indispensable .....	49
Objectifs, périmètre et parties prenantes .....	50
Recueillir les objectifs .....	51
Déterminer le périmètre .....	52
Analyser les parties prenantes .....	54
Grille de questionnement .....	56
Tableau des parties prenantes .....	57
Le document de cadrage .....	58
Check-list .....	58
Chapitre 7 – Planifier le projet d'élaboration .....	59
Le plan projet .....	59
Cadrer la méthodologie .....	60
Élaborer le plan projet .....	61
Check-list .....	64

## Partie 2 – Développement des exigences

Chapitre 8 – L'étape de recueil .....	67
L'enjeu .....	67
Le processus de recueil .....	68
Le plan de recueil .....	70
Risques liés au recueil et atténuation .....	70
Détermination des profils utilisateurs .....	71
Recherche des sources d'exigences .....	72
Techniques de recueil .....	73
L'analyse de documents .....	74
La réunion d'un groupe de travail .....	75

L'interview structurée individuelle .....	76
Le brainstorming .....	83
Le diagramme des affinités .....	83
L'observation directe .....	84
Les questionnaires .....	85
La réutilisation d'exigences .....	86
L'analyse de produits existants .....	86
Rechercher l'information là où elle se trouve .....	86
Les bonnes pratiques .....	87
Check-list de fin d'étape de recueil .....	91
<b>Chapitre 9 – Les cas d'utilisation .....</b>	<b>93</b>
Qu'est-ce qu'un cas d'utilisation ? .....	93
Le contenu d'un cas d'utilisation .....	94
Avantages des cas d'utilisation .....	96
Élaboration des cas d'utilisation .....	96
Un exemple .....	98
Difficultés et risques des cas d'utilisation .....	99
Les diagrammes de cas d'utilisation .....	100
<b>Chapitre 10 – L'étape d'analyse .....</b>	<b>103</b>
Utilité de l'étape d'analyse .....	103
Le processus d'analyse .....	104
Structurer et organiser les exigences .....	105
Établir un dictionnaire de données .....	107
Analyser les règles métier .....	107
Définir les priorités d'un projet .....	109
Modéliser sous forme graphique .....	112
Maquettes et prototypes .....	122
Matrices CRUD et RACI .....	124
Évaluer la faisabilité et le coût .....	124
Check-list d'analyse .....	126
<b>Chapitre 11 – Les exigences non fonctionnelles .....</b>	<b>127</b>
Les caractéristiques de qualité .....	127
La norme ISO/CEI 25000 .....	128
Zoom sur l'utilisabilité .....	133

Chapitre 12 – Exigences projet et contraintes techniques . . . . .	137
Contraintes de projet . . . . .	137
Contraintes d'environnement . . . . .	138
Services d'accompagnement . . . . .	139
Chapitre 13 – L'étape de spécification . . . . .	143
Le processus de spécification . . . . .	143
La formulation . . . . .	144
La structuration . . . . .	149
Cas des exigences non fonctionnelles . . . . .	151
Check-list de spécification d'une exigence . . . . .	152
Check-list d'étape de spécification . . . . .	153
Chapitre 14 – Structure du cahier des charges . . . . .	155
Le modèle de cahier des charges . . . . .	155
Le modèle IEEE 830 . . . . .	156
Le modèle AFNOR X50-151 . . . . .	158
Le modèle de Wiegers . . . . .	160
Le modèle Volere (Robertson & Robertson) . . . . .	162
Construire son propre modèle . . . . .	164
Check-list : cahier des charges . . . . .	165
Chapitre 15 – L'étape de validation . . . . .	167
Intérêt de la validation . . . . .	167
Le processus de validation . . . . .	168
Les techniques . . . . .	168
Vérification par check-lists . . . . .	169
Relecture simple . . . . .	170
Relecture croisée . . . . .	170
Revue formelle et inspection . . . . .	171
Élaboration de cas de test . . . . .	173
Contrôle qualité des exigences . . . . .	174
Impliquer les personnes concernées . . . . .	174
Check-list : validation . . . . .	175
Chapitre 16 – Un modèle de processus . . . . .	177
Un processus-type . . . . .	177
Étape 1 : cadrer . . . . .	178



Étape 2 : planifier . . . . .	179
Étape 3 : analyser l'existant . . . . .	180
Étape 4 : recueillir et analyser les besoins . . . . .	182
Étape 5 : spécifier les exigences . . . . .	184
Étape 6 : valider les exigences . . . . .	184
Étape 7 : capitaliser . . . . .	185
Chapitre 17 – Améliorer le processus . . . . .	187
Pourquoi le processus peut être amélioré ? . . . . .	187
Comment le processus peut être amélioré . . . . .	189
La méthode du document navette . . . . .	191

### Partie 3 – Faire vivre les exigences

Chapitre 18 – La gestion des exigences . . . . .	197
La gestion des changements . . . . .	197
Une discipline nécessaire et rentable . . . . .	203
Chapitre 19 – Les outils . . . . .	205
Check-list : votre projet en a-t-il besoin ? . . . . .	205
Les bonnes raisons d'utiliser les outils . . . . .	206
Fonctions de base et fonctions avancées . . . . .	206
Attention aux mirages . . . . .	210
Quand et comment choisir un outil ? . . . . .	210
Chapitre 20 – Au-delà des exigences . . . . .	213
Étude de choix . . . . .	213
Étude comparative complexe . . . . .	214
Étude d'opportunité complexe . . . . .	216
Étude d'intégrabilité et design . . . . .	218
Chapitre 21 – Neuf conseils . . . . .	221
1. Ayez toujours l'objectif en tête . . . . .	221
2. Analysez et validez au plus tôt . . . . .	222
3. Lancez-vous un défi et donnez-vous les moyens de le réussir . . . . .	222
4. Conciliez concepts et action de terrain . . . . .	223
5. Concentrez-vous sur votre livrable . . . . .	223
6. Sachez réussir presque à coup sûr . . . . .	224
7. Mettez en avant votre client . . . . .	224

8. Perdez un peu de temps pour en gagner beaucoup . . . . .	225
9. Faites de la définition des besoins un projet en soi . . . . .	225
Annexe – Les questions à large spectre . . . . .	227
Comment utiliser ces questions ? . . . . .	227
Liste de questions . . . . .	227
Glossaire français . . . . .	231
Glossaire bilingue . . . . .	235
Bibliographie . . . . .	237
Index . . . . .	239

# Guide de lecture

Cet ouvrage présente une démarche structurée d'ingénierie des exigences pour les systèmes d'information, en suivant une logique qui va des concepts généraux jusqu'aux conseils pratiques. Il a l'ambition de guider le lecteur depuis les objectifs stratégiques jusqu'à la validation finale du cahier des charges, et au-delà. Il gagne donc à être lu d'un bout à l'autre, dans l'ordre des chapitres.

Parallèlement, cet ouvrage a pour vocation de servir de guide de terrain que tout consultant ou analyste, quelle que soit son expérience, peut utiliser dans sa pratique quotidienne. Il doit donc être consultable de manière non linéaire. La carte heuristique de la figure 1 page 3 tente de concilier ces deux approches, en permettant au lecteur de naviguer plus facilement entre les étapes et les techniques.

La première partie de ce livre décrit le métier, la démarche et les étapes préparatoires à la définition des besoins, nécessaires pour mener à bien une mission d'élaboration d'un cahier des charges.

Le premier chapitre donne un exemple concret (bien que fictif) qui permet de mettre en situation tout lecteur, qu'il soit débutant ou expérimenté. Il introduit les notions de base et le vocabulaire, qui n'est pas encore bien stabilisé dans notre métier.

Le chapitre 2 est le seul à s'occuper du pourquoi. Il présente les enjeux, les coûts, les gains et les risques.

Le chapitre 3 énumère et détaille les compétences nécessaires à l'élaboration d'un bon cahier des charges. Il faut la voir comme une check-list qui sert à s'améliorer, voire à recruter un consultant.

Le chapitre 4 situe l'ingénierie des exigences dans le cycle de vie du logiciel ; il permet de mieux comprendre les enjeux et met en lumière les rapports de force entre les différents acteurs.

On présente au chapitre 5 la démarche de développement des exigences, qui part de l'objectif et va jusqu'au cahier des charges. Cette démarche est générique : elle concerne la quasi-totalité des activités d'élaboration

d'un cahier des charges. L'enjeu est de l'adapter à son organisation pour ensuite pouvoir l'optimiser.

Le chapitre 6 détaille l'étape préalable de définition des objectifs et du concept, cruciale et souvent négligée, dont dépend grandement la réussite d'un projet.

Définir les besoins pour un logiciel est un projet à part entière. Le chapitre 7 parle de gestion de projet, spécifiquement pour la phase d'exigences.

La deuxième partie de cet ouvrage décrit les quatre activités de développement des exigences : recueil (chapitre 8), analyse (chapitre 10), spécification (chapitre 13) et validation (chapitre 15). Des chapitres intermédiaires permettent de faire un zoom sur des techniques très utiles, comme les cas d'utilisation (chapitre 9). Le chapitre 11 est entièrement consacré aux exigences non fonctionnelles, beaucoup trop souvent négligées.

Le chapitre 14 donne la description de plusieurs modèles de spécification d'exigences (cahier des charges) qu'une organisation pourra reprendre en y apportant les adaptations nécessaires.

Le lecteur aura alors pris connaissance de la démarche générale et des différentes techniques qui permettent d'aller du recueil des besoins jusqu'au cahier des charges. Pour être efficace, il devra articuler ces techniques entre elles, c'est-à-dire s'appuyer sur un processus formalisé. Le chapitre 16 donne un exemple de processus, que tout consultant ou assistant à la maîtrise d'ouvrage devra adapter à son contexte de travail. Ce processus peut toujours être amélioré. Le chapitre 17 offre quelques pistes.

Une fois spécifiés, les besoins ne sont pas gravés dans le marbre. En réalité, ils ne vont cesser d'évoluer. La gestion des exigences, activité transversale qui consiste à gérer les demandes de modification, pendant et après l'élaboration du cahier des charges, fait l'objet du chapitre 18.

Les outils de gestion des exigences progressent et changent rapidement. C'est pourquoi nous n'allons pas lister les outils présents sur le marché, mais exposer l'utilité qu'ils peuvent avoir au cours d'un projet et les critères pour les choisir (chapitre 19).

L'activité de l'analyste ou de l'assistant à la maîtrise d'ouvrage ne s'arrête pas à l'élaboration du cahier des charges. L'étude de l'opportunité, de la faisabilité ou de l'intégrabilité d'une solution dans le système d'information fait partie de son métier. Les différentes techniques de choix, de sélection d'un progiciel et de l'étude de son intégration dans le système d'information sont traitées au chapitre 20.

Enfin, nous donnons au chapitre 21 neuf conseils pratiques qui proviennent de l'expérience de l'auteur.

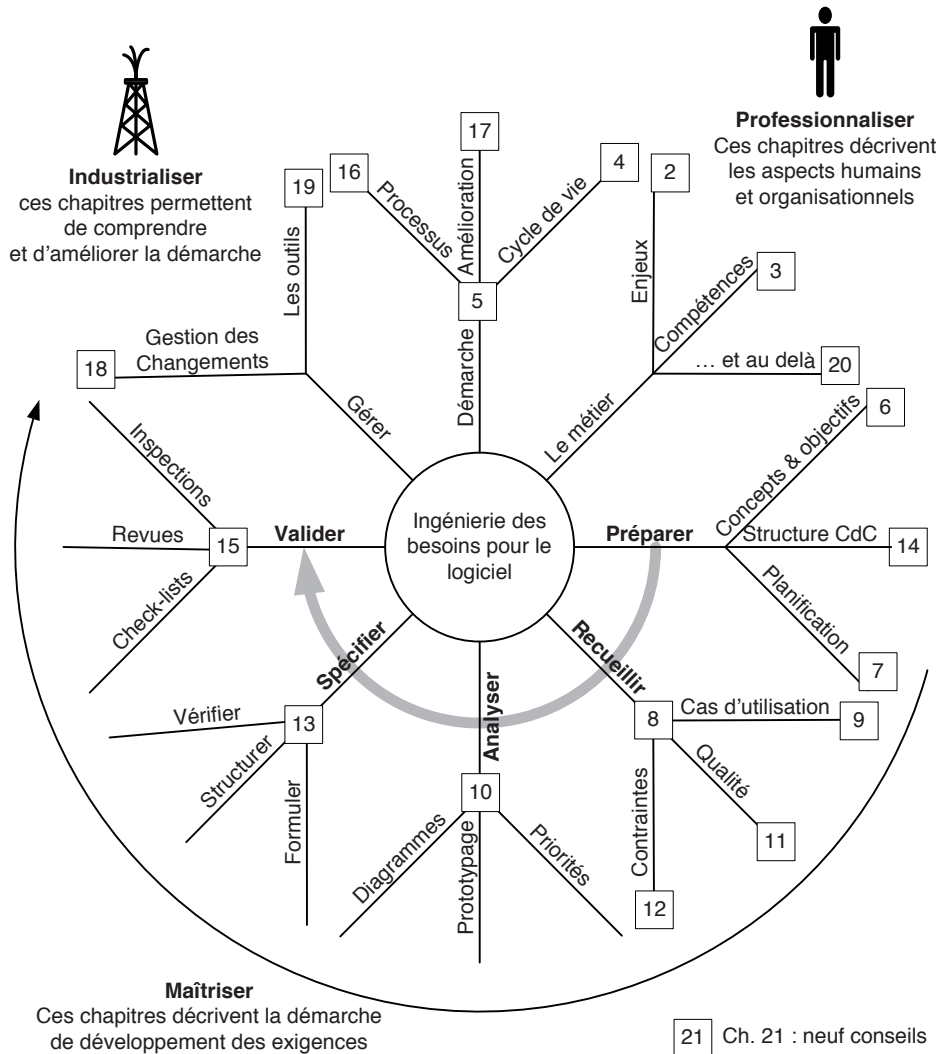


Figure 1 : La carte de lecture de ce livre



# Quiz : êtes-vous prêt ?

Ce test vous permettra de vous situer dans le métier du *business analysis*, quel que soit votre titre (consultant, analyste, AMOA, expert métier...). Pour chaque question, donnez la réponse qui vous paraît la plus juste.

Nous vous suggérons de répondre aux questions une première fois avant de lire ce livre, et de noter vos réponses. Puis une deuxième fois après l'avoir lu, de noter les réponses et de comparer les résultats.

1. Dans la constitution d'un cahier des charges, les difficultés proviennent surtout :
  - A. De la complexité technique.
  - B. Des aspects humains.
  - C. De la gestion des changements des exigences.
2. Pour élaborer un cahier des charges, les qualités requises sont :
  - A. Une bonne connaissance du domaine d'application.
  - B. Des bases solides en informatique.
  - C. La connaissance des techniques d'ingénierie des exigences.
3. Lorsque vous devez élaborer un cahier des charges, le mieux est :
  - A. D'utiliser tel quel un modèle de cahier des charges prédéfini.
  - B. D'adapter le modèle de cahier des charges à votre cas.
  - C. De créer un modèle spécifique de cahier des charges.
4. L'essence d'un cahier des charges est :
  - A. Un texte clair en français.
  - B. Des schémas rigoureux et lisibles.
  - C. Une structuration systématique et méthodique du document.

5. On vous demande d'élaborer un cahier des charges. Votre première tâche consiste à :
  - A. Faire préciser les objectifs.
  - B. Analyser le contexte.
  - C. Connaître les parties prenantes.
6. Dans le travail d'élaboration d'un cahier des charges, les deux qualités indispensables sont :
  - A. L'empathie et l'écoute.
  - B. La rigueur et l'organisation.
  - C. La créativité et l'imagination.
7. Dans le recueil des besoins, tout est dans :
  - A. La préparation.
  - B. L'écoute active.
  - C. Le feedback donné aux intéressés.
8. Dans l'analyse des exigences, il faut constamment surveiller :
  - A. La cohérence entre exigences.
  - B. La traçabilité des exigences.
  - C. La priorité entre exigences.
9. S'il ne fallait utiliser qu'un seul type de diagramme, ce serait :
  - A. Le diagramme de classes (ou entité-association).
  - B. Le diagramme d'états.
  - C. Le diagramme de flux.
10. Les exigences qui demandent le plus d'attention sont :
  - A. Les exigences fonctionnelles.
  - B. Les exigences non fonctionnelles.
  - C. Les contraintes.



# Réponses au quiz

Voici les réponses au quiz précédent.

1. Dans la constitution d'un cahier des charges, les difficultés proviennent surtout...

Si vos connaissances techniques sont faibles, il se peut que A soit la bonne réponse. Mais dans la majorité des cas, les problèmes humains (réponse B) sont les plus délicats à traiter, du moins dans un premier temps. À long terme, lorsqu'il s'agit de maintenir le cahier des charges ou de faire évoluer les exigences, le suivi des modifications des exigences pose de réelles difficultés.

2. Pour élaborer un cahier des charges, les qualités requises sont...

Une bonne connaissance du domaine d'application (réponse A) est indispensable, et des bases solides en informatique (réponse B) sont fort utiles. La connaissance des techniques d'ingénierie des exigences est aussi importante que les deux autres qualités, mais est très souvent négligée.

3. Lorsque vous devez élaborer un cahier des charges, le mieux est...

Si un modèle de cahier des charges prédéfini est bien adapté au contexte et aux objectifs, il n'y a pas de raison de s'en priver. Sinon, il faudra adapter le modèle de cahier des charges à votre cas (réponse B), en essayant de garder un maximum de cohérence. Il est rare de devoir créer un modèle spécifique de cahier des charges. Même dans ce dernier cas, on combinera des modèles existants.

Cela va sans dire, il vaut mieux investir du temps à adapter un modèle existant qu'à « réinventer la poudre ».

4. L'essence d'un cahier des charges est...

Tout dépend du domaine d'application, de la réponse recherchée (choix de progiciel ou développement spécifique). Mais on oublie trop souvent que la langue naturelle est une merveilleuse invention et un outil de spécification universel.

5. On vous demande d'élaborer un cahier des charges. Votre première tâche consiste à...

Sans objectif, vous n'irez pas loin. La réponse A est donc a priori la correcte. Mais qui va vous donner l'objectif ? Celui qui finance, celui qui décide ou celui qui réceptionne le produit ? Et comment allez-vous interpréter cet objectif en termes opérationnels ? D'où la nécessité d'analyser des parties prenantes et le contexte. Les trois tâches sont donc imbriquées.

6. Dans le travail d'élaboration d'un cahier des charges, les deux qualités indispensables sont...

... les vôtres ! Là aussi, il n'y a pas de meilleure réponse. Si chez vous certaines qualités sont nettement prédominantes, utilisez-les, bien sûr, et essayez aussi de cultiver les autres.

7. Dans le recueil des besoins, tout est dans...

Les trois réponses sont les bonnes, bien sûr. La préparation méthodique et systématique tient une place importante dans cet ouvrage. Elle vous fera gagner énormément de temps et vous évitera de disperser votre énergie et celle des participants. L'écoute est un don du ciel, le meilleur outil de l'analyste, qu'il faut toujours cultiver. Le feedback donné aux intéressés est une bonne pratique qui contribue à l'efficacité et à la fiabilité de l'activité de recueil que vous pouvez tout de suite mettre en application, à tous les niveaux.

8. Dans l'analyse des exigences, il faut constamment surveiller...

... à vous de deviner la réponse (vous vous en doutiez maintenant) ! Pour savoir ce qu'il faut surveiller, il suffit d'imaginer ce qui se passerait si vous ne le surveilliez pas !

9. S'il ne fallait utiliser qu'un seul type de diagramme, ce serait...

Un diagramme n'est pas une fin en soi, c'est un mode d'expression. Il s'agit d'être le plus clair et précis possible pour ceux qui vont vous lire. Ce que vous devez exprimer dépend de vos objectifs, du contexte, des parties prenantes, du domaine d'application et du niveau de détail exigé.

10. Les exigences les plus importantes à spécifier sont...

Toutes, bien sûr ! La seule chose que l'on peut dire est que les exigences non fonctionnelles sont difficiles à écrire et beaucoup trop souvent négligées. Pour les contraintes, cela est très dépendant du type de prestations que l'on attend du fournisseur.

# PARTIE

# 1

---

# Méthodologie

L'ingénierie des exigences consiste, dans une large mesure, à créer des modèles abstraits à partir d'un relevé de faits généralement beaucoup plus concrets, puis à les présenter de manière claire, structurée et intelligible par tous les acteurs, donc dans un langage souvent différent que celui de départ. La capacité à effectuer des voyages aller-retour dans l'abstraction, à élaborer des modèles, tout en restant proche du métier de ses interlocuteurs, constitue la principale difficulté, mais aussi tout l'attrait du métier d'analyste des exigences. Pour exercer ce métier et mener à bien les missions qui lui sont confiées, celui-ci va faire appel à ses compétences et à son savoir-faire acquis par l'expérience, ainsi qu'à une démarche rigoureuse.

Dans cette partie, nous allons introduire, puis définir précisément, les notions de besoin et d'exigence, les enjeux d'une bonne définition des besoins, les compétences nécessaires à ce travail de définition, et sa position dans le cycle de vie du logiciel.

L'efficacité de ces activités dépend dans une large mesure du soin apporté à leur préparation. Celle-ci comprend les activités de planification, inhérentes à tout projet, et que nous allons détailler pour le cas particulier du développement des exigences.

La préparation comprend également la détermination précise des objectifs, du champ de l'étude et des parties prenantes. Ces trois activités, auxquelles nous consacrons un chapitre, sont cruciales.



## Chapitre 1

# La méthode en action

Dans ce chapitre, nous allons tenter d'introduire à la fois le vocabulaire, la problématique et les concepts. La situation prise comme exemple est fictive, mais présente toutes les difficultés auxquelles est confronté un analyste ou assistant à maîtrise d'ouvrage au cours de sa mission. Ces difficultés rendent nécessaire une approche organisée et systématique, appelée *ingénierie des exigences*.

### Un exemple imaginaire, mais concret

Supposons que la direction du marketing d'un constructeur automobile envisage de mettre sur le marché un nouveau véhicule révolutionnaire, équipé d'un pilote automatique. Plus rien à faire pour le conducteur, ou alors si peu... Supposons que nous sommes à la tête d'une équipe chargée de spécifier<sup>1</sup> ce que le système de pilotage automatique doit faire, doit être ou doit avoir pour être opérationnel. Supposons que nous-mêmes ne connaissons rien, ou presque rien, à la mécanique automobile, ni même aux systèmes embarqués dans les véhicules, domaines réservés des techniciens et ingénieurs. Notre challenge est de définir les besoins et de les spécifier sous forme d'*exigences*.

---

1. Le challenge :  
décrire le besoin sans  
décrire la solution.

---

#### L'analyste

Le métier de la définition des besoins est désigné en anglais par *requirements analyst* ou *business analyst*. En français, on parle d'analyste, d'analyste métier, de consultant métier, parfois d'analyste des exigences, et très souvent de consultant assistant à la maîtrise d'ouvrage. Dans la suite de cet ouvrage, nous utiliserons indifféremment ces termes.

Pour nous, la direction du marketing est le maître d'ouvrage opérationnel. Les techniciens et ingénieurs sont le maître d'œuvre. La direction

générale, nécessairement impliquée dans un projet d'une telle envergure, est le maître d'ouvrage stratégique. Le document que nous allons produire, et qui s'adressera à l'ensemble de ces acteurs, appelés parties prenantes, s'appelle cahier des charges.

### L'analyse de l'existant

La première tâche à laquelle nous allons nous atteler consiste à examiner tout ce qui existe en matière d'automatismes embarqués dans les véhicules. En effet, bien qu'aucune voiture actuellement sur le marché ne dispose à proprement parler d'un pilote automatique, de nombreux mécanismes, qu'ils soient mécaniques, électromécaniques, ou informatisés, existent déjà. Cette première tâche, donc, appelée analyse de l'existant, est indispensable et va nous occuper pour un bon moment. Elle est indissociable de l'activité de recueil des besoins, car l'existant d'une automobile pourra servir à spécifier les besoins d'une automobile à venir.

Concrètement, l'analyse de l'existant va consister à :

- étudier la documentation des automobiles existantes ;
- examiner les spécifications des automobiles déjà conçues, mais qui n'ont jamais vu le jour ;
- visiter les ateliers, les usines, les chaînes de montage ;
- examiner les automobiles de la concurrence ;
- examiner ce qui se fait, en matière de pilotage automatique, dans d'autres domaines que l'automobile.

### Le cahier des charges

Après avoir étudié l'existant, nous allons passer à la partie la plus longue et la plus difficile de notre travail, l'élaboration du cahier des charges. Comme indiqué, le cahier des charges est un document que tous les acteurs en présence doivent être capables de comprendre facilement, et qui servira de contrat entre plusieurs parties prenantes. Pour ce faire, il devra donc être écrit dans un langage clair et concis. Pour l'essentiel, il sera constitué d'un ensemble de phrases telles que, par exemple :

En mode automatique, le véhicule doit se conformer au Code de la route.

Il s'agit là de ce que l'on appelle une exigence de haut niveau. Énoncée ainsi, elle paraît évidente, mais doit néanmoins être spécifiée de manière précise, non ambiguë, dans un langage que tous les acteurs comprennent.

## Les règles de gestion

Plus loin dans le cahier des charges, cette exigence de haut niveau sera exprimée en exigences de plus bas niveau, comme :

En mode automatique, le système fera en sorte que le véhicule respecte les limitations de vitesse imposées par la signalisation.

En mode automatique, le système fera en sorte que le véhicule n'accélère pas lorsqu'un véhicule placé derrière lui tente de le dépasser.

Dans ces phrases, qui constituent un échantillon parmi des centaines d'autres, le marketing (maître d'ouvrage) demande au bureau d'études (le maître d'œuvre) que le système fasse en sorte qu'à tout moment, lorsque le véhicule est sous contrôle du pilote automatique, le Code de la route soit respecté.

Ces phrases sont des exigences d'un type particulier. Dans cet ouvrage, et dans le vocabulaire consacré de l'ingénierie des exigences, on les appelle règles de gestion (en anglais, *business rules*).

### Où stocker les règles de gestion ?

Si les règles de gestion ne sont ni plus ni moins que des articles de loi, ou des extraits de textes réglementaires, ou des éléments de procédure, faut-il nécessairement les reprendre dans le cahier des charges, ou bien peut-on se contenter d'y faire référence ?

La question est loin d'être triviale, elle n'a pas de réponse définitive, elle dépend de plusieurs facteurs : qui lira le cahier des charges ? La réglementation est-elle suffisamment claire pour être comprise des ingénieurs ? N'y a-t-il pas des règlements qui se contredisent entre eux ?

## Les contraintes

Il ne suffit pas de recopier le Code de la route, ou de le paraphraser, ou d'y faire référence, pour élaborer le cahier des charges d'un pilote automatique pour automobile. Dans le cahier des charges, on trouvera des exigences telles que :

Lorsque le régime du moteur dépasse un seuil maximum noté  $R_s$ , le système devra faire en sorte que le véhicule change de régime en passant à la vitesse supérieure.

Lorsque le régime du moteur passe sous un seuil minimum noté  $R_m$ , le système devra faire en sorte que le véhicule change de régime et rétrograde.

Notons en passant que ces exigences permettent de percevoir une frontière entre deux mondes : le *système*, que nous sommes en train d'étudier

(parfois appelé SAE, système à l'étude) et le *contexte*, qui est l'ensemble des systèmes extérieurs au système à l'étude, et avec lesquelles il réagit. Le contexte peut être constitué de logiciels, de matériels, et d'humains (comme le conducteur du véhicule). La détermination du contexte est une activité indispensable que nous allons détailler dans un prochain chapitre.

### Les exigences fonctionnelles

Exigences réglementaires et contraintes techniques sont des exigences extérieures à l'organisation qui exprime les besoins. La majorité des besoins internes à l'organisation vont se traduire par des exigences fonctionnelles. Par exemple :

En l'absence d'exigences réglementaires et de contraintes mécaniques, la vitesse du véhicule sera celle déterminée par l'utilisateur.

Si la vitesse n'a pas été déterminée par l'utilisateur, et en l'absence d'exigences plus prioritaires, le véhicule accélérera jusqu'à atteindre la vitesse maximale autorisée par la signalisation ou la réglementation.

### Les exigences métier

Ni le Code de la route, ni une quelconque contrainte mécanique n'obligent un véhicule à circuler à la vitesse maximale autorisée. L'exigence que nous venons de citer est une exigence métier (en anglais, *business requirement*). Dans notre exemple, la « direction métier » est le marketing. Cette dernière exigence, déterminant la vitesse maximale, peut se rapporter à une exigence de plus haut niveau telle que :

En l'absence de contraintes mécaniques ou réglementaires, le véhicule circulera à la vitesse maximale possible.

Cette exigence peut elle-même se rapporter à une exigence de niveau supérieur, c'est-à-dire à un objectif :

Les performances du véhicule devront dépasser celles des véhicules de la concurrence de la même catégorie.

On remarquera en passant que les exigences peuvent former une cascade, entre les exigences de plus haut niveau, les objectifs, et les exigences du niveau le plus opérationnel. Une des difficultés de l'ingénierie des exigences consiste à maintenir la traçabilité entre les exigences de différents niveaux.



### La négociation des exigences

Tous les métiers, à l'instar du marketing, pourront exprimer des besoins. Il est évident que des exigences provenant de sources différentes pourront être antinomiques. Par exemple, la direction du développement durable voudra minimiser la consommation en énergie, alors qu'une autre voudra maximiser la sécurité des passagers. Une des activités de l'analyse des exigences consiste à négocier les exigences face à des besoins antagonistes. On commence à voir une première distinction entre besoin et exigence : les besoins sont ceux des personnes ou des organisations, et peuvent être contradictoires. Les exigences, formulées dans un cahier des charges, ne le sont pas.

### Les autres exigences

Les systèmes informatiques manipulent des données. Les exigences sur les données, comme les formats de données, les seuils de déclenchement, les relations entre données, constituent une catégorie à part.

Les exigences fonctionnelles décrivent ce que le système doit faire, en d'autres termes son comportement. Une autre catégorie d'exigences, aisées à comprendre, mais difficiles à décrire, concerne la qualité du système. On les appelle exigences non fonctionnelles.

### Classer les exigences par catégories

Il y a plusieurs manières de structurer les exigences. Dans ce chapitre, nous avons illustré la structuration donnée par Karl Wiegers. Elle classe les exigences client dans l'une des neuf catégories suivantes :

- Exigences métier (*business requirements*) : elles concernent les avantages, bénéfices, intérêts stratégiques ou tactiques (augmentation du chiffre d'affaires, économies, gains).
- Scénarios et cas d'utilisation : ils correspondent à des besoins opérationnels des utilisateurs (avoir la possibilité de saisir, d'imprimer, de gérer une information).
- Règles de gestion (règles métier) : le logiciel doit être conforme à une loi, un règlement, une directive ou une procédure.
- Exigences fonctionnelles, relatives au comportement du système.
- Attributs de qualité, également appelés exigences non fonctionnelles : ce sont des caractéristiques exigées du système (fiabilité, maintenabilité, portabilité, ergonomie) trop souvent négligées et passées sous silence.
- Exigence d'interfaces externes : interface utilisateur, interface avec le matériel ou un autre logiciel.
- Contraintes : elles apportent des restrictions au système ; il s'agit souvent de contraintes liées au matériel (volumes traités, plate-forme). Si trop de contraintes apparaissent lors du recueil des besoins, cela peut être un frein à la recherche de la solution adéquate ; il est utile de connaître les motivations de celui qui les formule.
- Définitions de données, relatives au format des données (par exemple, la forme d'un numéro de Sécurité sociale ou d'un numéro de série).

- Suggestions de solutions, formulées comme des exigences ; souvent, un utilisateur a une idée préconçue de la solution à mettre en œuvre. Lors du recueil et de l'analyse des exigences, il est important de les faire expliciter, puis de rechercher les motivations de celui qui les suggère.

D'autres exigences peuvent être formulées et concerner le projet, les coûts, les délais, mais elles sont distinctes des exigences produit.

Nous reviendrons sur ces différentes catégories dans la partie de l'ouvrage consacrée au développement des exigences.

### Le métier : gérer une combinaison d'exigences

Une combinaison d'exigences (règles métier, exigences métier, contraintes techniques) peut donner naissance à de nouvelles exigences, bien réelles, mais jamais formulées comme telles. C'est là une des grandes difficultés de l'analyse des exigences.

Reprenons notre exemple fictif du cahier des charges d'un pilote automatique pour automobile. Le Code de la route, ou du moins son sous-ensemble concernant la conduite automobile, constitue un ensemble de règles métier. En voici une (pas toujours respectée des automobilistes !) :

#### Article R412-31 du Code de la route

Tout conducteur doit marquer l'arrêt devant un feu de signalisation jaune fixe, sauf dans le cas où, lors de l'allumage dudit feu, le conducteur ne peut plus arrêter son véhicule dans des conditions de sécurité suffisantes.

Le « feu de signalisation jaune fixe » est ce que l'on appelle communément un feu orange. Pourquoi cette règle de gestion ne peut-elle pas être reprise telle quelle dans le cahier des charges ? Il y a à cela plusieurs raisons :

- Le vocabulaire du Code de la route n'est pas le même que celui du cahier des charges. Bien sûr, on pourrait s'aligner sur ce vocabulaire juridique, puisque c'est la loi. Mais pourquoi pas sur celui de l'industrie automobile, qui est plus précis sur le plan de la mécanique ?
- La règle n'est centrée ni sur le système à l'étude (le pilote automatique), ni sur le véhicule, ni sur l'utilisateur du système. Elle est centrée sur le couple conducteur-véhicule, représenté par « le conducteur », seul responsable, devant la loi, du comportement de son véhicule (en cas de défaillance du véhicule, il pourra par la suite se retourner contre le constructeur automobile, comme cela peut être le cas lors de toute défaillance d'un système automatisé).
- Surtout, l'article fait référence à une autre règle ou contrainte, qui est décrite ailleurs ou est censée être compréhensible par tous. Cette contrainte

concerne les conditions de sécurité « suffisantes ». Elle est sans doute facile à interpréter (quoique...) par les automobilistes, qui doivent respecter la loi, et par les autorités, qui doivent la faire respecter. Mais elle est beaucoup trop floue pour décrire le comportement attendu d'un système.

En réalité, l'article cité fait appel à d'autres articles, ainsi qu'à la jurisprudence, qui permet également de déterminer ce que sont des conditions de sécurité suffisante, dans quelles circonstances le véhicule pouvait ou non être arrêté sans danger, etc.

Tentons de traduire cette règle métier en exigence système. Cela donne :

```
Si
le feu de signalisation dont dépend le véhicule passe à
l'orange,
et si
le système dispose du temps nécessaire pour arrêter le véhicule
à l'aplomb du signal
et si
en s'arrêtant, le véhicule ne met pas en danger le conducteur,
les passagers et l'environnement du véhicule
et si
les conditions de sécurité telles que définies par [un ensemble de
règles de gestion préalablement définies] sont respectées
alors
le système fait en sorte que le véhicule ralentisse
progressivement jusqu'à s'arrêter à l'aplomb du signal.
```

Il s'agit là d'une traduction encore très approximative de la règle, qu'il faudra certainement retravailler. Le but de l'exercice est de montrer à quel point une règle de gestion qui nous paraît simple, par ce que nous vivons tous les jours avec, se traduit en une exigence système qui peut être extrêmement complexe.

En particulier, une analyse rapide de la règle de gestion, en interaction avec les contraintes mécaniques, fait émerger de nouvelles conditions qui étaient cachées ou sous-entendues jusque-là. Dans notre exemple, c'est le facteur *temps* qui émerge.

D'autres contraintes peuvent émerger, comme la notion de risque. La règle de gestion peut très bien s'écrire :

```
Si, en arrêtant le véhicule à l'aplomb du signal, le système
cause moins de dégâts qu'il n'en aurait causés en ne s'arrêtant
pas, alors le système doit arrêter le véhicule à l'aplomb du
signal.
```

Cette dernière formulation de l'exigence est sans doute moins « politiquement correcte », mais beaucoup plus proche de la réalité que l'article de loi. En effet, un conducteur automobile fait en permanence un calcul de risques. Il ne se contente pas de s'assurer que les conditions de sécurité sont vérifiées (elles ne le sont jamais à 100 %), il estime les risques à passer à l'orange et à ne pas passer, puis compare les deux en temps réel (souvent, il compare aussi le risque de rater son rendez-vous avec celui de se faire pincer par un agent de la circulation, mais cela nous éloigne du strict respect de la loi).

Selon le niveau de cahier des charges à élaborer, l'article R412-31 du Code de la route se traduira par quelques dizaines, centaines, voire milliers d'exigences logicielles, qui tiendront compte d'exigences aussi différentes que le Code de la route, la psychologie du comportement des piétons, ou les contraintes mécaniques.

### La notion de partie prenante

Cet exemple fait ressortir la notion de partie prenante, extrêmement importante et trop souvent négligée. Un ingénieur mécanicien, un juriste et le représentant d'une association d'automobilistes sont tous trois des parties prenantes au projet. Considérons chacun de ces acteurs comme une personne d'expérience, de bonne foi, douée de bon sens. Cela ne les empêchera pas d'exprimer des besoins contradictoires, car le bon sens est une question de point de vue.

L'analyste devra trouver et faire approuver un consensus entre des expressions de besoins conflictuels, en gérant des priorités entre exigences. Sa tâche sera grandement facilitée si, au lieu de devoir gérer des contradictions, voire des conflits tout au long du recueil des besoins, il anticipe sur ces éventuels conflits en analysant les intérêts et les rapports de force des différents acteurs. Cette tâche s'appelle *analyse des parties prenantes*.

## Le point sur les concepts

---

### Besoin, demande, exigence et spécification

On a vu, grâce à l'exemple précédent, les différences entre besoin, demande et exigence.

Le *besoin* est interne à une organisation ou une personne. Dans le cas d'une personne physique, il peut être ressenti, intellectuellement ou physiquement. Il peut être exprimé plus ou moins clairement, sous-entendu, passé sous silence, ignoré ou nié par son auteur.

La *demande* est l'expression d'un besoin. Elle peut être supérieure au besoin, en termes de coût, de qualité, de fonctions, ou lui être inférieure.

Par rapport à la demande, une *exigence* tient compte des contraintes, et introduit une notion de consensus, de cible à atteindre. Elle est demandée par le client et doit être acceptée par le fournisseur. Elle doit être mesurable, ou du moins vérifiable.

Enfin, la *spécification* est la traduction des exigences dans un langage convenu entre le client et le fournisseur. Elle permet d'éviter les effets pervers qui perturbent leurs relations, y compris dans le cas où les deux parties, client et fournisseur, sont de bonne foi. La spécification des exigences est une démarche industrielle.

### Ingénierie des exigences

L'ingénierie des besoins, ou ingénierie des exigences, est donc la discipline, vue comme un ensemble de techniques, consistant à recueillir les besoins, à les transformer en exigences, et à les spécifier dans un cahier des charges.

Quand on parle d'ingénierie, on pense souvent à des techniques complexes. Cependant, sur le plan conceptuel, les différentes techniques décrites dans cet ouvrage sont simples. Comme nous le verrons, les vraies difficultés proviennent essentiellement :

- du choix entre plusieurs techniques ;
- de la nécessité d'un suivi rigoureux des exigences ;
- des aspects humains.

### Cahier des charges

Un cahier des charges peut donc être défini de trois façons :

- C'est l'expression des besoins, après qu'ils aient été recueillis, analysés, négociés, priorisés, spécifiés, classés.
- C'est un ensemble de demandes, après traitement et arbitrage.
- C'est un ensemble structuré d'exigences, qu'un client présente à un fournisseur.



---

## Chapitre 2

# Les enjeux d'une bonne définition des besoins

Le cahier des charges est un document constitué d'un ensemble structuré d'exigences, transmis d'un client à un fournisseur, et qui a un caractère contractuel. Comme nous le verrons tout au long de cet ouvrage, son élaboration est un travail collectif nécessitant la collaboration de nombreux acteurs. Le processus de cette élaboration est aussi important que le document qui en résulte.

### L'utilité d'un cahier des charges

---

Un bon cahier des charges est un document clair, facile à lire et à comprendre par les différents acteurs, qui a été établi sur la base d'un consensus. Étant donnée la complexité de certaines situations que ce document représente, il peut être très complexe, mais il n'est jamais « compliqué » : c'est un document relativement facile à maintenir et à modifier.

Ce document, élaboré de manière participative, définissant clairement les exigences d'un client vis-à-vis d'un fournisseur, sera utile au choix, à la conception, à la réalisation, au paramétrage d'une application ou d'un système informatique, ainsi qu'à sa mise en œuvre opérationnelle, sa maintenance et son exploitation.

### Un investissement très rentable

---

L'ensemble des activités d'élaboration qui part du recueil des besoins et aboutit à un cahier des charges est appelé *développement des exigences*. Les enjeux liés au développement des exigences sont très importants. Les personnes qui élaborent un cahier des charges pour la première

1. Ces chiffres émanent de plusieurs études, en particulier celles de Boehm et de Grady.

fois, ou demandent de l'assistance pour son élaboration, sont rarement conscientes de l'importance de cette phase, et en sous-estiment les conséquences. Plutôt qu'un long discours, voici quelques faits<sup>1</sup> :

Environ les trois quarts des erreurs dans le choix, la mise en œuvre et la construction d'un logiciel sont dues à des exigences mal formulées ou à un cahier des charges mal construit (nous verrons plus loin ce que cela signifie).

Le coût de la correction d'une erreur produite lors de la phase d'exigences est beaucoup plus important (jusqu'à cent fois) lorsque cette erreur est découverte en phase d'exploitation que pendant la phase d'exigences elle-même (voir figure 2-1).

Ce constat, statistiquement démontré, est apparemment contre-intuitif, puisque nombre de donneurs d'ordres n'investissent pas suffisamment de temps et d'argent dans cette phase, pensant sans doute que c'est du temps et de l'argent perdus, et qu'on pourra toujours corriger le tir et rattraper les petites erreurs par la suite. Or, corriger le tir coûte largement plus cher que de viser juste du premier coup.

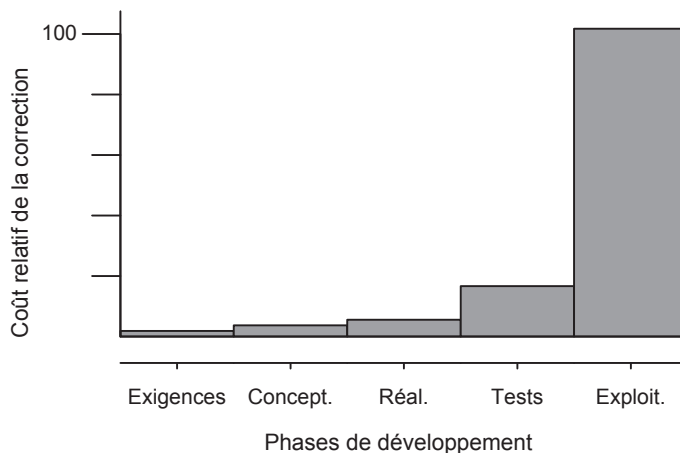


Figure 2-1 : Augmentation exponentielle des erreurs

Si le cahier des charges est utilisé pour le choix d'un logiciel sur étagère (progiciel), le coût d'un mauvais choix est nettement supérieur au coût d'un bon cahier des charges.

Enfin, la réécriture de spécifications ou de programmes (le *rework*) suite à des spécifications erronées ou insuffisamment précises peut coûter jusqu'à la moitié du coût total du développement.

Spécifier les besoins de manière scrupuleuse est donc un investissement extrêmement rentable. Il peut rapporter plus de cent fois ce qu'il aura coûté.



### **Retour sur investissement**

Sans être ambitieux, on peut tabler sur un retour sur investissement de 1000 % : la différence entre un cahier des charges bien construit et un cahier des charges médiocre va rapporter au moins dix fois ce qu’elle aura coûté. Ce coût inclut l’effort d’élaboration du cahier des charges lui-même, le coût du « retravail » (*rework*) des exigences en phase de réalisation, et bien sûr la rentabilité du logiciel produit.

## Les gains « cachés »

---

Les gains que nous venons de mentionner sont les plus visibles.

Mais obtenir de bonnes spécifications dès le début apporte de nombreux bénéfices secondaires, et pas seulement sur le plan financier. Réussir du premier coup est beaucoup plus motivant et encourageant pour les utilisateurs que de revenir sur des spécifications.

De plus, le processus même d’élaboration d’un cahier d’exigences permet aux différentes parties prenantes, à commencer par les utilisateurs, de participer à une démarche collaborative de recueil, de spécification, de validation des besoins, et de réfléchir à l’impact de la mise en œuvre d’un nouveau système sur les processus actuels.

D’autres gains sont induits par le processus même de l’élaboration. Élaborer un cahier des charges est un travail d’équipe qui, bien mené, va souder cette dernière autour d’un projet commun. L’effort ultérieur de mise en œuvre du produit, de formation, de conduite du changement en sera grandement facilité.

Enfin, *last but not least*, un cahier des charges bien construit coûte beaucoup moins cher qu’un cahier des charges élaboré de manière empirique, avec des allers-retours incessants entre les différentes parties prenantes. Élaborer un cahier des charges n’est jamais chose aisée, suivre une méthode rigoureuse et des techniques éprouvées va grandement faciliter les choses.

## Les difficultés

---

Inutile de se voiler la face. Le travail d’élaboration est long et difficile. Même avec une bonne méthode et des têtes bien faites, c’est une épreuve, et on passe toujours par des moments de découragement. Il est important de connaître par avance certaines difficultés pour mieux pouvoir s’en prémunir.

La première difficulté vient du donneur d’ordres. Ce peut être la direction générale d’une entreprise, ou une administration centrale dans le secteur

public. Le donneur d'ordres doit s'impliquer, définir les objectifs, donner une lettre de mission claire à l'équipe qui va définir les besoins. Ce n'est pas toujours le cas. Si la direction ne s'implique pas, il faudra tout faire pour qu'elle s'engage dans ce projet, car sa réussite en dépend.

La deuxième difficulté provient du rythme de travail, qu'il convient de cadencer pour conserver la motivation des acteurs jusqu'au bout et éviter l'essoufflement. En effet, l'enthousiasme de départ ne dure pas éternellement, surtout lorsque la définition des besoins devient un exercice routinier, avec ses lectures croisées et ses revues formelles. L'inspiration peut aussi manquer au démarrage du projet, pendant les phases créatives de recueil des besoins, lorsque les futurs utilisateurs manquent d'idées. Dans les deux cas, c'est le talent de l'analyste des besoins qui est requis.

Une autre difficulté concerne les pressions extérieures aux groupes de travail. Le donneur d'ordres peut être pressé d'avoir des résultats rapides, et préférer un travail vite fait, mal fait, à un document conclusif. À d'autres moments, le donneur d'ordres peut au contraire lâcher la tension, pris par d'autres priorités. Or, la définition des besoins est un exercice qui doit être rythmé.

Enfin, une difficulté importante vient du fait que les gains engendrés par une bonne définition des besoins ne seront visibles qu'à très long terme, c'est-à-dire lorsque le logiciel sera en production. Pourquoi un dirigeant investirait-il dans une activité qui n'est visible que lorsqu'elle est défaillante ?

## Les risques

---

Quels que soient la rigueur méthodologique et le talent de l'analyste, les risques existent et sont nombreux. En voici quelques-uns, parmi les plus importants.

### Les spécifications rampantes

Il est normal que les exigences évoluent. Dans un tel cas, les spécifications du produit changent en conséquence, et le produit lui-même évoluera par la suite. Les spécifications rampantes adviennent lorsque l'évolution des exigences n'est pas contrôlée. Le produit ainsi couvert de « rustines » risque de perdre en robustesse et en cohérence.

Souvent, les phases d'exigences et de conception sont bien séparées. C'est en particulier le cas des appels d'offres publics, où le cahier des clauses techniques particulières (le CCTP) doit être finalisé et approuvé et où toute modification doit en principe faire l'objet d'un avenant au

contrat initial. Même dans ce cas, les spécifications peuvent évoluer de manière incontrôlée. Le cahier des charges devient alors un patchwork, avec tous les risques d’incohérences que cela peut entraîner.

Pour éviter cette dérive, il faut d’une part que les objectifs du logiciel aient été clairement définis, et d’autre part que les modifications du cahier des charges fassent partie d’un processus contrôlé.

### **Le perfectionnisme (surspécification)**

Donner trop de détails ne sert à rien, et peut être nuisible. C’est en particulier le cas lorsque les utilisateurs ou leurs représentants donnent des détails sur la manière dont une fonction sera mise en œuvre, en particulier en spécifiant l’interface utilisateur (comme « cliquer avec le bouton droit de la souris »). De tels détails n’ont rien à faire dans un cahier des charges. S’il faut absolument spécifier des contraintes techniques, ce sera dans une annexe, et non dans la description fonctionnelle des exigences.

Une autre forme de surspécification consiste à exiger des fonctions superflues. Cela augmentera inutilement le coût du logiciel et les délais de livraison. Généralement, un utilisateur ne connaît pas le coût de ses exigences. Pour éviter cette dérive, il est indispensable d’avoir cadré le projet, en temps, en coût et en qualité. Le travail de l’assistant à la maîtrise d’ouvrage consiste alors à rappeler à son client les objectifs que ce client a lui-même fixés.

### **Les spécifications insuffisamment détaillées**

Le risque est ici de décrire les exigences de manière très globale (sous-spécification), en espérant que les concepteurs et réalisateurs vont avoir suffisamment de « bon sens » pour faire un produit conforme à ces besoins sous-entendus. C’est une utopie. Si un besoin est insuffisamment explicité, un développeur fera appel à sa créativité, essaiera de deviner ce que veut le client, avec de fortes chances de se tromper.

### **Négliger certains profils utilisateurs**

Un même logiciel s’adresse généralement à des types d’utilisateurs différents<sup>2</sup>, avec des besoins parfois très distincts. Bien que pouvant accéder aux mêmes informations, un comptable et un conseiller de clientèle ne travaillent pas du tout de la même façon. Pour construire un logiciel qui soit au service de plusieurs catégories d’utilisateurs, il est indispensable de tenir compte de leurs besoins particuliers, et pour cela de les faire participer à l’expression des exigences.

---

2. Voir le Chapitre 6, « Analyser des parties prenantes ».

---

## Améliorer les pratiques d'élaboration

---

Des bonnes pratiques d'expression et de gestion des exigences amélioreront considérablement la qualité du cahier des charges, et partant de l'application finale. Cela est vrai dans tous les cas, qu'il s'agisse de développer une application sur mesure ou de choisir, paramétrer et installer un progiciel.

Cet ouvrage n'a d'autre ambition que d'être un manuel de bonnes pratiques qui permettront :

- de donner aux personnes en charge de l'élaboration des spécifications d'exigences (cahier des charges) une approche et des outils qui faciliteront leur travail ;
- d'améliorer la communication entre client (maîtrise d'ouvrage), fournisseur (maîtrise d'œuvre) et analyste (assistant à la maîtrise d'ouvrage) ;
- d'éviter le phénomène des spécifications rampantes ;
- d'améliorer l'efficacité en choisissant la technique adéquate de recueil, d'analyse, de spécification des exigences ;
- de minimiser l'effort en évitant d'effectuer certaines tâches trop tôt, ou de descendre trop vite dans les détails, ce qui obligera de revenir dessus une deuxième fois ;
- de sécuriser ce processus, grâce à des check-lists de fin d'étape ;
- de conserver la motivation des participants tout au long de l'élaboration ;
- de préparer le terrain à l'élaboration de cas de test.

## Chapitre 3

# Compétences et savoir-faire

Élaborer un cahier des charges exige de solides connaissances techniques et méthodologiques, ainsi que des qualités humaines. Pour cette raison, il n'y a pas un parcours type pour devenir analyste des exigences. On trouve dans ce métier des anciens utilisateurs de produit, des ex-développeurs, des experts du domaine d'application. L'analyste doit apprendre, puis maîtriser, les techniques de l'ingénierie des besoins. C'est un animateur, un négociateur, un interprète qui va écouter les besoins des futurs utilisateurs et les reformuler dans un langage clair (figure 3-1).

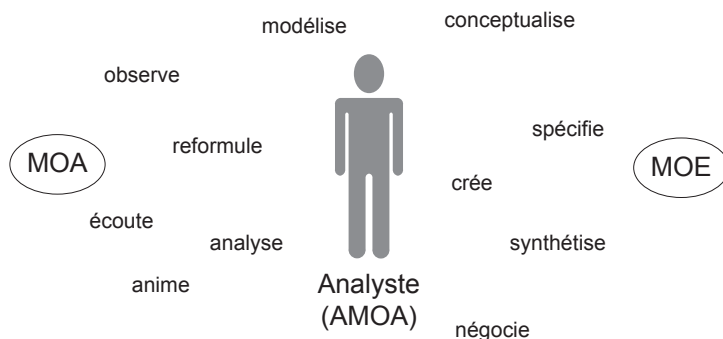


Figure 3-1 : Les compétences de l'analyste

## Le savoir

### La connaissance du métier du client

Sans être un expert, l'analyste doit être familiarisé avec le métier de son client. En fonction des domaines d'activité, du niveau de détails attendu

et du type d'application, cette connaissance se doit d'être plus ou moins approfondie.

Très souvent, l'expertise est chez le client. Sans être expert, l'analyste doit être capable de faire appel aux experts, de comprendre leur langage, de se plonger dans leurs procédures, d'analyser leurs processus ; en d'autres termes, de connaître leur métier sans pour autant le pratiquer.

### **La connaissance des techniques de modélisation**

La représentation graphique des processus, des procédures, des données, des traitements sont souvent nécessaires, lorsque le langage écrit ne suffit pas ou manque de précision. Dans un tel cas, un bon schéma vaut mieux qu'un long discours.

Cependant, il ne suffit pas de savoir dessiner un schéma ou un diagramme pour savoir modéliser. Encore faut-il comprendre la sémantique, et les conséquences profondes de certains choix. Ainsi, sans empiéter sur le travail des techniciens, élaborer un modèle graphique demande quelques connaissances techniques. Par exemple, pour élaborer un modèle de données, il est utile de connaître les principes des bases de données relationnelles. Les schémas seront lus par les futurs utilisateurs qui ont besoin de clarté et de simplicité, mais aussi par les futurs développeurs qui ont besoin de cohérence et de précision.

### **La connaissance des métiers du développement**

L'analyste des exigences ne « développe » pas de logiciel au sens courant du terme (on a vu que l'analyse des exigences fait partie du cycle de développement). Avoir pratiqué le développement est cependant un avantage qui permet de savoir si un besoin exprimé est réaliste, et à quel prix il peut être réalisé. Cela permet aussi, chaque fois que nécessaire, de dialoguer avec les développeurs et de se faire comprendre.

Enfin, cela permet de comprendre certaines contraintes et des compromis à faire : plus de sécurité au détriment de la facilité d'utilisation, plus de richesse fonctionnelle au détriment de la fiabilité...

### **La connaissance des technologies**

Il n'est pas indispensable d'être un bon technicien pour spécifier des exigences et élaborer un cahier des charges. C'est souvent l'inverse qui est vrai, car un bon technicien a tendance à rechercher des solutions techniques plutôt que de chercher à comprendre le besoin de son client. De bonnes connaissances techniques constituent néanmoins un sérieux atout. Cela évite de chercher l'impossible là où des solutions toutes prêtes sont sur le marché. Cela permet aussi de connaître les limites imposées par certaines techniques.

## Le savoir-faire

---

### L'art de poser les bonnes questions

La question est à l'analyste ce que le bistouri est au chirurgien : un outil de travail quotidien, très simple, mais difficile à manier. Comme le bistouri, la question doit être tranchante, bien affûtée, et faire le moins de dégâts possible.

Poser la bonne question au bon moment est une affaire de talent, d'entraînement, mais aussi, et surtout, de logique et de méthode. On peut avoir un répertoire de questions génériques « prêtes-à-poser »<sup>1</sup> pour éviter d'être pris au dépourvu. Mais un bon analyste parcourt mentalement un arbre de décision et sait lier les questions entre elles dans un ordre logique, en cohérence avec le point qu'il examine et les objectifs à atteindre.

---

1. Voir en annexe la liste de questions générales.

---

### Une aptitude à négocier

La personne chargée du recueil des besoins et de l'élaboration du cahier des charges joue un rôle d'intermédiaire entre maîtrise d'ouvrage et maîtrise d'œuvre. Elle doit être « bilingue » ; non seulement connaître le vocabulaire, comprendre le métier, mais également pouvoir se mettre dans la peau des différents acteurs pour comprendre leur point de vue, les contraintes de chacun, la manière de s'exprimer. Spécifier des exigences, c'est avant tout « traduire » le langage du médecin, du juriste ou du banquier en un langage intermédiaire compris de tous les acteurs.

### Les qualités d'animateur

Le talent d'animateur est indispensable, car le recueil des exigences est souvent une activité d'équipe. Savoir animer un groupe de travail est une qualité essentielle. Différents moyens vont permettre aux participants de s'exprimer : brainstorming, maquettage sur papier, narration de situations vécues, discussions, jeux de rôle. Certaines techniques sont très efficaces.

Il faut aussi savoir créer la motivation, puis la maintenir, souvent pendant de longs mois, au fil des réunions.

### La qualité d'organisateur et de chef de projet

Définir les exigences demande une planification soignée, une préparation minutieuse, une organisation de son propre travail et de celui de son équipe, en tenant compte des contraintes du client et des autres parties prenantes. Une fois que le projet est lancé, on doit faire face à l'imprévu. À l'image de tout chef de projet, l'analyste des exigences doit être capable de mener à bien ces deux activités. Plus la planification sera rigoureuse, et plus il sera facile de parer l'imprévu.

### L'expression écrite

La capacité à écrire avec aisance et précision est fondamentale. Parmi les nombreuses manières de représenter les exigences, diagrammes, tableaux ou schémas, la plus usitée, et généralement la plus efficace, demeure la langue française. C'est en effet la seule représentation facilement compréhensible par les gens de tous les métiers.

La langue française (et toute autre langue « naturelle », en fonction du pays où l'on se trouve et du milieu professionnel) est un outil extrêmement complexe et puissant, qui demande un long apprentissage et une longue pratique. Nous la maîtrisons tous, avec plus ou moins de talent, mais élaborer un cahier des charges exige l'utilisation d'un langage précis et non ambigu.

### Le savoir-être

---

#### Une attitude de chef de projet

Trois dangers menacent l'analyste des exigences :

- En manquant de rigueur, vouloir toujours faire plaisir à tout le monde ; cela conduit à l'enlisement du projet.
- En manquant de souplesse, se faire un ennemi parmi les parties prenantes ; il risque de se faire « griller ».
- Face à des besoins mal définis, laisser le flou s'installer, voire l'amplifier ; or, sa mission exige de la précision.

Face à ces trois menaces, la bonne attitude consiste à être rigoureux sur le processus et souple avec les personnes. C'est cette neutralité bienveillante qui lui permettra de s'en sortir indemne.

#### La curiosité

Le don d'observation et la créativité vont de pair avec la curiosité. Faire de la veille technologique est indispensable, et il faut la faire activement. La curiosité mêlée de créativité donne envie de « piquer des idées » et de les formaliser ensuite. Rappelons que les idées n'appartiennent à personne. Copier les idées en y ajoutant les siennes propres est légitime (surtout si, par respect du travail des autres, on cite ses sources).

#### L'écoute

Pour recueillir les besoins, il est plus important de savoir écouter que de savoir bien parler. Une bonne écoute est pourtant une qualité rare, et *très* appréciée des utilisateurs.

Un analyste doit être capable de mettre son interlocuteur en confiance ; de l'encourager à parler de son métier ; de formuler une question en



tenant compte du contexte et du métier de son interlocuteur ; de le relancer pour avoir des précisions ; de stimuler son imagination et de le mettre en situation pour décrire une situation réelle ou imaginer une situation à venir ; de l'encourager à descendre dans les détails ou, à l'inverse, de prendre de la hauteur par rapport à sa situation.

Une bonne écoute est liée au talent, à l'expérience et à la personnalité de l'analyste. Mais des interviews bien préparées et bien structurées aident le talent à s'exprimer et les moins talentueux à s'en sortir honnêtement. Nous donnerons quelques règles simples de préparation des interviews dans le chapitre consacré au recueil des besoins.

### Un excellent relationnel

Cette qualité est principalement utile lors de l'animation de groupes de travail. Lors d'une telle réunion, un analyste doit être capable de maintenir la motivation, de laisser chacun s'exprimer, de cadrer et de recadrer pour rester aligné sur les objectifs, de résoudre les conflits, de maintenir la bonne humeur, de permettre aux plus timides de s'exprimer, et de canaliser les plus bavards.

Comme les qualités d'écoute, les qualités relationnelles sont grandement dépendantes du talent et de la personnalité de l'analyste. Et comme pour l'écoute, une bonne préparation et des techniques d'animation peuvent pallier le manque d'expérience.

### L'observation

Une des nombreuses manières de recueillir les besoins consiste à observer les personnes en train de travailler, dans leur milieu professionnel. Mais le don d'observation est utile à tout moment. Lors d'une interview, il est utile d'observer comment l'interlocuteur réagit, et comment les participants à une réunion interagissent et communiquent entre eux ; tout noter, sur le papier ou dans sa tête, sous forme écrite ou sous forme de croquis. Parfois, deux yeux et deux oreilles ne suffisent pas, et lors de certaines réunions un coanimateur peut être utile.

### La créativité

L'analyste des exigences n'est pas toujours un être neutre et incolore. Il est là pour aider le client et l'utilisateur à définir, préciser et formaliser le besoin. Mais le futur produit peut aussi contenir des caractéristiques nouvelles auxquelles le client seul ne pensera jamais. L'analyste ou le consultant, qui est en contact avec des clients parfois très différents, peut, et doit, apporter des idées nouvelles.

Il peut adapter à un domaine des idées tirées d'un autre domaine. Par exemple, l'exigence fonctionnelle « relancer périodiquement les prospects » d'un

logiciel commercial peut être adaptée à un logiciel de cabinet dentaire pour relancer les patients. Ce sera ensuite aux futurs utilisateurs (en l'occurrence, les dentistes) de dire si cette idée est à conserver. Encore faut-il que cette idée soit proposée.

Si tous les concepteurs se contentaient d'écouter « la voix du client », il n'y aurait que très peu d'innovations. Cela est particulièrement le cas des clients qui sont des utilisateurs expérimentés d'un produit existant. Souvent, ils demandent à ce que le nouveau produit fasse « la même chose » que le précédent, mais en mieux. Il faut être capable de leur proposer le « mieux », qui peut aller du simple lifting de l'interface graphique à des améliorations fonctionnelles conséquentes.

### L'esprit d'analyse et de synthèse

Un cahier des charges est une arborescence d'exigences. Il faut être capable de travailler sur les branches de haut niveau pour définir les grandes fonctions, puis descendre dans le détail de chacune jusqu'aux branches les plus fines, sans se perdre, puis remonter à nouveau pour donner une vision globale au donneur d'ordres. Tout cela doit être fait avec un « scalpel mental » à la main, qui va permettre de séparer ce qui appartient à l'univers des exigences (à conserver dans un cahier des charges) de ce qui est de l'ordre de la solution (à manier avec précaution).

L'analyste des exigences doit également posséder une sorte d'alarme intérieure qui le prévient de toutes les contradictions entre exigences (son bon relationnel doit l'aider à éviter que ces contradictions se transforment en conflits de personnes) et un « détecteur de flou », ou plutôt une aversion caractérisée pour les formulations floues, ambiguës, aux interprétations multiples.

### La clarté

Le métier d'expression des exigences est un métier de *communication*. Pour faire passer des messages entre client et fournisseur, et aussi entre utilisateurs de métiers différents, il est donc primordial de s'exprimer avec aisance et clarté, oralement et par écrit.

L'ingénieur des exigences utilise toute une palette d'outils : la parole dans un premier temps, puis le langage écrit ou graphique. Dans tous les cas, il doit utiliser le moins de jargon possible. Quand un mot nouveau ou « barbare »<sup>2</sup> (c'est-à-dire, appartenant au métier de « l'autre », et qui n'a pas la même signification pour toutes les parties prenantes) doit apparaître dans le discours, il doit devenir pédagogue et clarifier le concept.

L'analyste des exigences doit manier plusieurs langages. Le langage parlé et écrit en est un et occupe une place prépondérante. Mais il existe

---

2. « Chacun appelle barbarie ce qui n'est pas de son usage », Montaigne.

---

de nombreux langages graphiques, plus ou moins adaptés aux interlocuteurs des différentes parties prenantes. Il doit savoir les manier avec discernement. Certaines personnes, de par leur profession, sont très familiarisées avec le langage graphique. C'est le cas des ingénieurs. D'autres professions, comme les juristes, préfèrent l'écrit. Un schéma qui paraîtra limpide à l'un pourra être abscons pour l'autre. Il en est de même des tableaux. Représenter les informations sous forme tabulaire va souvent de soi pour un ingénieur ou un comptable, moins pour un juriste. Inversement, décrire un processus opérationnel au moyen d'un long discours écrit peut couler de source pour un homme de loi et rebuter un ingénieur qui a besoin de s'appuyer sur des schémas pour faciliter sa compréhension. Ces différences culturelles doivent être impérativement respectées si l'on veut éviter les incompréhensions, voire les conflits.



# Exigences et cycle de vie du logiciel

Dans ce chapitre, nous allons situer les activités de l'ingénierie des exigences dans le cycle de vie du logiciel. Cela va nous permettre de prendre la mesure de l'importance de ces activités, et de mieux comprendre les rôles et responsabilités des différents acteurs, ainsi que ceux du maître d'œuvre, du maître d'ouvrage, et de l'analyste et de leurs engagements réciproques.

### Le cycle de vie du logiciel

---

Il existe plusieurs façons de représenter le cycle de vie du logiciel. Le terme de cycle est d'ailleurs souvent utilisé abusivement. Voici (source : IEEE Std 982.2-1988) un modèle de cycle de vie en huit phases :

- Objectifs et concepts : définition des objectifs du produit à construire, et première vision du logiciel.
- Exigences : recueil, analyse et formalisation des besoins, élaboration du cahier des charges.
- Conception : spécifications détaillées fonctionnelles et techniques, en fonction des objectifs et des exigences précédemment définis.
- Réalisation : programmation et tests unitaires.
- Tests et validation : tests fonctionnels, d'intégration et validation.
- Installation : mise en œuvre opérationnelle et déploiement.
- Exploitation et maintenance : mise à disposition auprès des utilisateurs.
- Retrait : arrêt de l'exploitation, éventuellement migration vers un autre logiciel ou nouvelle version du même logiciel.

Ce cycle en huit phases est véritablement circulaire, ce qui permet de prendre conscience que, dans la vraie vie, un logiciel arrive rarement sur

un terrain vide, et ne laisse pas souvent un vide en partant (figure 4-1). Bien avant de mettre un logiciel hors service, on se préoccupe de son successeur. C'est en général l'occasion de redéfinir le concept et les objectifs, ainsi que les exigences qui en découlent.

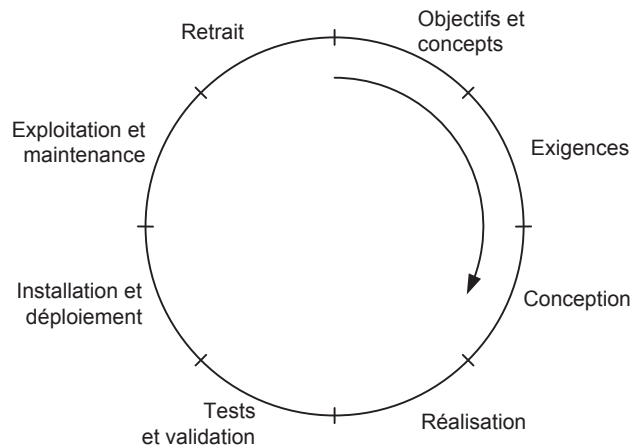


Figure 4-1 : Le cycle de vie, modèle IEEE 982.2

Ce cycle en huit phases donne, bien sûr, une vision schématisée de la réalité (c'est précisément ce qui est demandé à un modèle). Dans la pratique, les phases ne sont égales ni en temps ni en coûts, et elles ne sont pas séparées par des cloisons étanches.

Entre deux versions d'un même logiciel, ou entre un logiciel et son successeur, on peut constater un retard de plusieurs phases. Il n'est pas rare que, pendant qu'un logiciel est encore en validation, on définisse déjà les besoins de son successeur.

## Maîtres d'ouvrage et maîtres d'œuvre

---

Le cycle de vie ainsi représenté nous permet de visualiser les relations et le partage des responsabilités entre parties prenantes. Le maître d'ouvrage est celui qui demande, spécifie, paie ou utilise le système, ou plus généralement en tire un bénéfice. Le maître d'œuvre est celui qui le conçoit, le réalise, le met en œuvre.

Si on examine le cycle de vie du logiciel au regard de cette distribution des rôles (figure 4-2), on s'aperçoit que le maître d'ouvrage (ou client) est responsable de « l'hémisphère nord » du cycle et que le maître d'œuvre

occupe « l'hémisphère sud ». On a donc, dans la partie haute du cycle, ceux qui définissent et utilisent le *quoi*. Et dans la partie basse, ceux qui sont responsables du *comment*.

Les stratégies, ceux qui décident de ce que sera, dans ses très grandes lignes, un nouveau produit, sont au pôle nord du cycle. Les experts, qui maîtrisent dans le détail les techniques pointues, sont au sud.

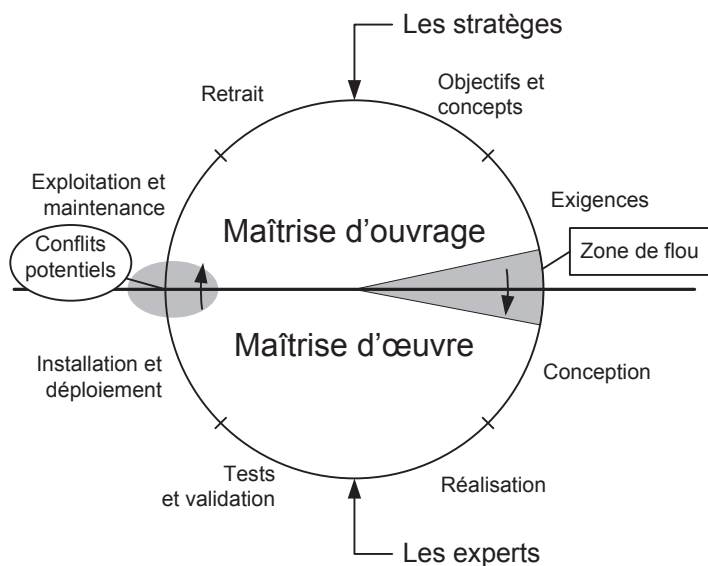


Figure 4-2 : Cycle de vie et partage des rôles

Bien entendu, une même personne peut, au cours de sa carrière, se trouver alternativement client ou fournisseur. Mais au cours d'un même projet, les rôles sont en principe séparés, ainsi que les manières de travailler. À la frontière de ces deux mondes se trouve le cahier des charges, qui devient ainsi un contrat entre deux parties. De la qualité de ce document dépendra grandement la bonne coopération entre client (maîtrise d'ouvrage, donneur d'ordres) et fournisseur (éditeur, intégrateur, vendeur).

Entre maîtrise d'ouvrage et maîtrise d'œuvre, il y a souvent des difficultés de compréhension, dues à des différences de métier et de langage. De ce fait, la tentation est grande de livrer un cahier des charges qui contient du flou, des zones d'ombre, destinées à pallier les difficultés de compréhension. C'est ce qui arrive souvent, et ne fait que repousser le problème : les difficultés reviennent à la surface au moment de la livraison, générant des tensions, voire des conflits.

## Bâtisseurs et exploitants

---

Le cycle de vie du logiciel peut être découpé verticalement. La partie droite est celle des bâtisseurs, qui imaginent, conçoivent, réalisent. La partie gauche est celle des exploitants, qui installent, exploitent, maintiennent. Entre ces deux populations, c'est surtout la motivation qui est différente. Les bâtisseurs sont attirés vers le nouveau, imaginent toutes les possibilités, veulent créer. La motivation des exploitants est la continuité et la régularité du fonctionnement, et la correction des erreurs et anomalies. Pour cela, ils s'appuient sur des procédures rigoureuses. Ici aussi, il faut trouver un langage commun entre les différentes populations.

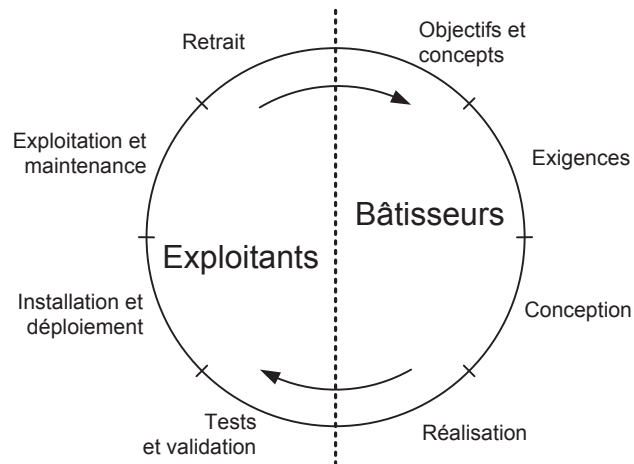


Figure 4-3 : Cycle de vie et répartition des métiers

## La phase d'exigences

---

La phase d'exigences sera décrite très en détail dans les chapitres qui suivent. Elle comporte les activités et tâches suivantes :

- identifier les différents profils utilisateurs de l'application ;
- recueillir les besoins de chaque profil utilisateur ;
- analyser les besoins des utilisateurs, en éliminer les incohérences et les redondances ;
- traduire les besoins exprimés oralement en spécifications ;
- aligner la spécification des exigences (le cahier des charges) aux objectifs définis par le maître d'ouvrage ;
- assurer la complétude du cahier des charges ;



- déterminer, avec le maître d'ouvrage et les utilisateurs, la priorité relative de chaque exigence et son niveau de flexibilité (obligatoire, facultatif) ;
- faire valider par le maître d'ouvrage les exigences spécifiées.

Comme on le voit sur la figure 4-4, la phase d'exigences fait partie de la construction du logiciel ou du système, et elle est sous la responsabilité du maître d'ouvrage.

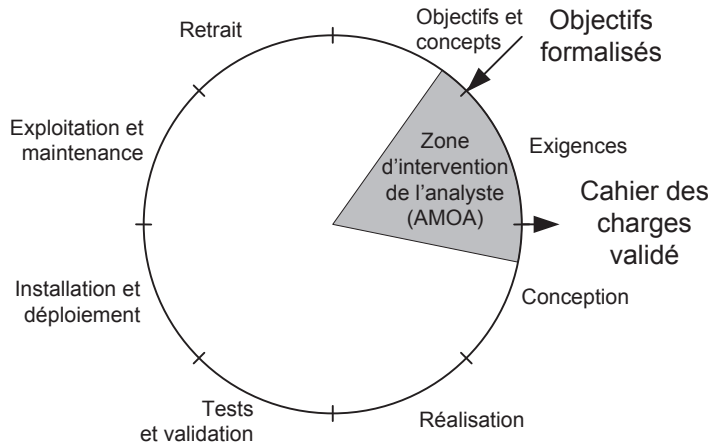


Figure 4-4 : La phase d'exigences dans le cycle de vie

Dans la pratique, le travail de l'analyste va au-delà de la phase d'exigences proprement dite. Il est fréquent que les concepts et les objectifs du logiciel ne soient pas clairement définis et ce sera à l'analyste des exigences de les formaliser et de les faire valider. Et, bien que cela dépasse le strict cadre du cahier des charges, l'analyste s'aventure souvent dans la zone floue entre les exigences et la conception.

## Les engagements réciproques

L'élaboration du cahier des charges est sous la responsabilité d'une personne, l'analyste des exigences, souvent appelé *assistant à la maîtrise d'ouvrage*. Étant donné l'importance du cahier des charges, son rôle est crucial, et devrait également être régi par un contrat, entre son client et lui. C'est souvent un contrat moral, mais il gagne à être mis par écrit.

Un consultant pourra intégrer ces engagements réciproques à une proposition commerciale d'élaboration d'un cahier des charges. Cependant, même sans aucun écrit de part et d'autre, l'analyste des exigences a tout

intérêt à connaître la liste de ces engagements, et de s'y tenir. Il devra également mettre son client face à ses responsabilités.

Voici un modèle d'engagements réciproques :

### **Engagements réciproques**

#### **L'analyste (assistant à la maîtrise d'ouvrage) s'engage à :**

- aligner son travail sur les objectifs de son client,
- se former au métier du client et se tenir informé et de ses pratiques,
- s'exprimer en utilisant le vocabulaire de son client,
- informer son client de l'avancement du cahier des charges,
- tenir son client informé des méthodes et outils utilisés,
- spécifier les besoins en termes aisément compréhensibles par le client,
- choisir les techniques et outils les plus adaptés, et les mettre en œuvre,
- animer une démarche collaborative respectueuse des utilisateurs,
- animer les groupes de travail avec une neutralité bienveillante,
- apporter des idées nouvelles, dans le respect de cette neutralité,
- estimer les charges, délais et risques, et en informer son client,
- optimiser son temps et ses efforts, et ceux des groupes de travail,
- s'efforcer à spécifier, dans les règles de l'art, un logiciel de qualité.

#### **Le client (maître d'ouvrage) s'engage à :**

- exprimer clairement l'objectif,
- s'investir dans l'expression des besoins,
- respecter la démarche d'expression des besoins mise en place,
- inciter les parties prenantes à participer aux groupes de travail,
- informer le consultant sur son métier et ses pratiques,
- exprimer les besoins avec clarté, exactitude et précision,
- lever les ambiguïtés sur l'expression d'un besoin,
- indiquer les priorités sur les exigences exprimées,
- respecter les estimations faites par le consultant ou l'expert,
- valider les documents intermédiaires,
- valider le cahier des charges,
- communiquer sans délai les modifications d'exigences.

Les engagements de l'analyste peuvent faire fonction de charte de déontologie. Les engagements du maître d'ouvrage, même s'ils ne sont pas formalisés, peuvent servir de check-list au maître d'ouvrage. Pour l'analyste, c'est la check-list de ce qu'il peut légitimement demander à son client.

## Chapitre 5

# La démarche

Pour être efficace, la définition des exigences doit devenir une activité systématique et organisée, faisant appel à des acteurs dont les relations sont formalisées ; en d'autres termes, une activité d'ingénierie. Dans le chapitre précédent, nous avons vu comment cette activité s'insère dans le cycle de vie du logiciel. Nous allons maintenant rentrer un peu plus dans le détail et décrire globalement le processus de développement des exigences.

### Décrire, documenter, communiquer

---

Tout au long de cet ouvrage, nous décrivons des techniques qui, regroupées, s'appellent *ingénierie des besoins* ou *ingénierie des exigences*. Voilà une belle expression, mais dans quelle mesure le terme d'ingénierie est-il justifié ? Recueillir, analyser, spécifier les besoins sont en grande partie des actions de *description*. Il s'agit de décrire un besoin en le formulant pour la première fois, en le reformulant différemment, de manière plus claire, plus constructive, plus cohérente ; en le montrant sous différents angles, comme un dessin d'architecte montre un bâtiment selon des vues différentes ; en le formulant dans un langage que tous (en tout cas tous ceux qui ont besoin de cette description), comprennent et interprètent de la même façon.

Décomposer un ensemble de besoins en sous-ensembles plus petits, de manière arborescente, le décrire sous différents aspects va immanquablement générer beaucoup de *documentation*. Il s'agit là d'un aspect fondamental de ce métier. Nous devons gérer des masses d'informations et les présenter de manière cohérente. Il faudra également maintenir cette cohérence dans le temps, gérer cette évolution. La gestion des versions et des changements fait partie du métier.

Toutes ces informations devront être partagées par tous ces acteurs. La définition des besoins est donc aussi une activité de *communication*. Cette communication se fait bien sûr dans les deux sens : écoute et spécification.

Nous avons dans ce triptyque l'essence du métier d'analyste : décrire, documenter, communiquer (figure 5-1).

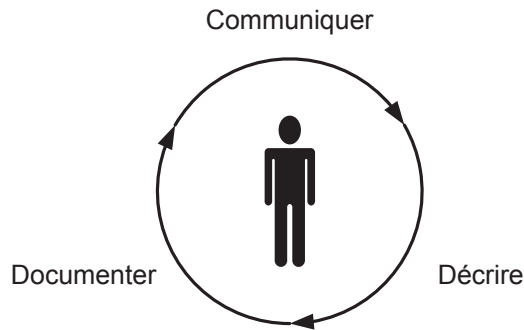


Figure 5-1 : L'essence de notre métier

## Les différents niveaux d'exigences

---

Les activités de recueil, d'analyse, de spécification et de validation des exigences peuvent se faire à plusieurs niveaux :

- Les objectifs stratégiques<sup>1</sup> fixés par la direction générale ou par le donneur d'ordres, et dont le recueil et la formalisation demandent un soin particulier.
- Les exigences de haut niveau, liées à l'organisation (en anglais *business requirements*), découlant directement des objectifs ou de contraintes organisationnelles ou réglementaires.
- Les exigences de niveau intermédiaire correspondent au point de vue d'un utilisateur (en anglais *user requirements*) : exigences comportementales, cas d'utilisation, exigences de qualité.
- Les exigences élémentaires (en anglais *atomic requirements*), qui décrivent sous forme de phrases simples (du type « le système doit... ») des exigences fonctionnelles ou non fonctionnelles, des contraintes, d'ordre technique ou autre, ou des exigences d'interface.

Cette classification (qui peut varier selon les modèles de cahiers des charges) permet de structurer la spécification en « couches » d'exigences de plus en plus détaillées (figure 5-2). Les exigences à un niveau doivent être alignées aux exigences de niveau supérieur.

---

1. Objectifs : voir au chapitre 6.

Les exigences de haut niveau (les objectifs) feront l'objet d'un document particulier, qui pourra être repris dans le cahier des charges. En fonction de sa destination, le cœur du cahier des charges descendra plus ou moins dans les détails. Par exemple, un cahier des charges destiné à développer un logiciel contiendra plus d'exigences détaillées qu'un cahier des charges pour le choix d'un progiciel sur étagère.

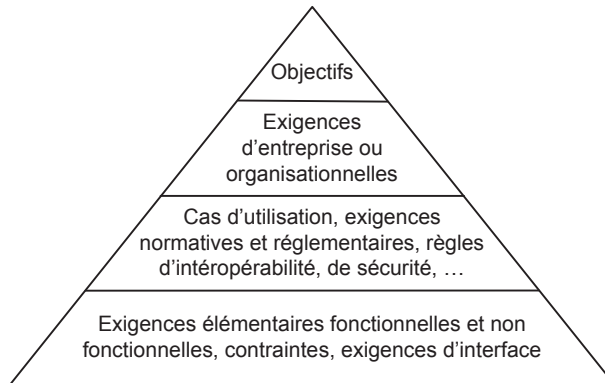


Figure 5-2 : Niveaux d'exigences

## Les étapes de l'élaboration

---

Élaborer un cahier des charges consiste donc avant tout à traduire des besoins flous, imprécis, et parfois inconnus, en exigences structurées et organisées ; à les traduire donc, depuis le langage du client en un langage compréhensible de tous (client, fournisseur et observateurs extérieurs). Présentons les différentes tâches qui vont du recueil des besoins au cahier des charges.

La première tâche consiste à **découvrir les enjeux, les objectifs, et les contraintes du projet**. Les enjeux constituent la raison profonde du lancement d'un projet, les intentions derrière les objectifs. Certains enjeux seront clairs et pourront être exprimés dans le préambule du cahier des charges. D'autres enjeux pourront être cachés (ambitions personnelles, projet inavoué de dépasser un concurrent). Qu'ils soient publics ou tenus secrets, ils devront être analysés, pris en compte. Les objectifs, eux, devront être formalisés. Ils constituent les exigences de plus haut niveau. Les contraintes ne sont qu'une forme particulière d'exigences, présentées « en creux ».

Une tâche importante consiste à **identifier les différents acteurs du projet** (les parties prenantes, en particulier les représentants des futurs utilisateurs), à connaître les enjeux les plus importants pour chacun d'eux, à établir un dialogue, à les faire participer au projet d'élaboration.

---

2. En anglais, le terme utilisé pour le recueil est *elicitation*. Il peut se traduire par « extraction »

---

**Recueillir les besoins** est une tâche beaucoup plus complexe qu'il n'y paraît. Les besoins sont rarement à la surface du sol<sup>2</sup>. Ils sont en général cachés, et diverses techniques permettent de les « déterrer » : les réunions de travail, les interviews des utilisateurs, l'observation directe, la lecture de documents (y compris d'autres cahiers des charges). Il n'y a pas de technique plus efficace qu'une autre, tout dépend du contexte, et c'est une combinaison de techniques qui permettra un recueil complet.

Ainsi recueillies, les exigences seront **analysées**. Analyser les exigences, c'est les faire passer par des « filtres », les examiner pour détecter les exigences manquantes, contradictoires, les incohérences à divers niveaux. Une analyse efficace des exigences permet de toujours rester au meilleur niveau de détail. Le niveau de détail nécessaire dépend des intentions du projet. Par exemple, on n'exprimera pas le même niveau de détails pour un projet de développement spécifique et un choix de logiciel sur étagère. De plus, il faudra établir des priorités entre exigences.

Il va falloir ensuite **rédiger les exigences**, généralement sous forme de cahier des charges. La forme peut être **textuelle** ou **graphique**. Il n'y a pas de forme a priori meilleure qu'une autre. Tout dépend de celui qui va lire le cahier des charges, l'important étant de donner une vision et une compréhension partagées des exigences. La représentation graphique fait appel à des techniques de **modélisation** des données ou des traitements informatiques.

Vu du rédacteur, le cahier des charges n'est qu'un lieu de stockage des exigences. Mais les exigences évoluent. **Gérer les évolutions** fait également partie de ce qu'il est convenu d'appeler l'ingénierie des exigences. À un instant donné, une version du cahier des charges peut être en cours d'utilisation (par les équipes de concepteurs, par exemple), une autre en cours d'élaboration. Ce sont alors des techniques de gestion des versions et de la configuration qui seront utilisées.

Enfin, les exigences devront être validées par les différentes parties prenantes. Cette validation se fait à plusieurs niveaux. Les futurs utilisateurs valideront souvent au fur et à mesure de la rédaction, le donneur d'ordre validera généralement l'ensemble du document. Le **suivi de la validation des exigences** n'est donc pas une tâche facile.

## Description formelle du processus global

---

Formellement, l'ingénierie des exigences comporte deux types d'activités :

- le développement des exigences, qui consiste à définir les besoins et à élaborer un cahier des charges ;

- la gestion des exigences, qui consiste à gérer les changements et les évolutions des exigences dans le temps.

Le développement des exigences comporte quatre étapes très fortement imbriquées selon un processus cyclique :

- *le recueil*, qui consiste à faire exprimer les besoins et à rechercher les besoins déjà exprimés ;
- *l'analyse*, qui consiste à examiner les exigences sous différentes facettes, et à maintenir la cohérence entre les exigences ;
- *la spécification*, qui consiste à décrire et documenter les exigences de manière à la fois formelle et compréhensible par toutes les parties prenantes ;
- *la validation*, qui consiste à obtenir, de la part de toutes les parties prenantes, un accord formel sur les exigences spécifiées.

Comme on peut le voir, il y a des boucles de rétroaction entre les quatre activités.

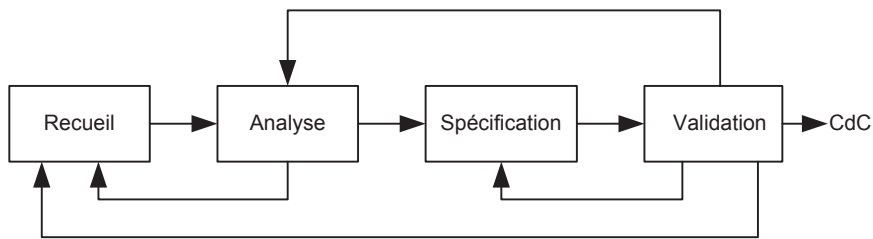


Figure 5-3 : Le processus global

Le découpage en quatre étapes permet de mieux comprendre le métier de définition des exigences. En pratique, ces étapes sont imbriquées dans le temps, chaque étape contenant une partie des autres. Par exemple, lors d'une même entrevue avec un utilisateur du futur produit (activité de l'étape de recueil), on peut écouter les besoins (recueil), tenter de comprendre comment ce besoin se rattache aux objectifs de la direction (analyse), essayer de reformuler le besoin de manière claire (spécification) et demander à l'utilisateur son opinion sur cette formulation (validation).

## Le processus en pratique

Nous avons vu qu'en pratique les quatre activités de recueil, d'analyse, de spécification et de validation, ne sont ni linéaires ni bien distinctes. À ces quatre activités s'ajoute la gestion des évolutions des exigences, qui en réalité commence dès le début du recueil. S'y ajoute une activité en amont de définition des objectifs (en toute rigueur, c'est une phase

3. Voir au chapitre 16 un modèle de processus qui peut être adapté.

précédente du cycle de vie, mais dans tous les cas, elle devra être déclinée pendant la phase d'exigences). Et, cela va sans dire, une activité transverse de gestion de projet.

Il n'existe cependant pas un modèle unique de processus<sup>3</sup>, autrement dit aucune « recette de cuisine » qui nous indiquerait les tâches à mener à chaque étape, et dans quel ordre.

Chaque projet est différent, et la première tâche de l'équipe chargée de définir les besoins consiste à ajuster le processus de définition aux contraintes propres au projet. Ce qui est cependant certain, et vérifié dans la pratique, c'est que les étapes en amont (la préparation), si elles sont bien menées, facilitent grandement la définition des besoins proprement dite (la production du cahier des charges). En d'autres termes, définir les besoins devrait être une activité « descendante » (*top-down*) dont les étapes sont les suivantes :

- Étapes en amont (préparation de l'élaboration du cahier des charges) :
  - définir le concept et préciser les objectifs,
  - analyser les parties prenantes : rôles, responsabilités,
  - définir les catégories (classes) d'utilisateur,
  - sélectionner, dans chaque catégorie, les représentants des utilisateurs,
  - choisir, en fonction du contexte et des contraintes, les techniques de recueil à mettre en place,
  - définir les contours du futur produit (périmètre),
  - définir les échanges entre le futur produit (vu comme une boîte noire) et le « monde extérieur »,
  - identifier les cas d'utilisation métier (*business use cases*),
  - établir des priorités entre cas d'utilisation,
  - sélectionner les cas d'utilisation qui seront informatisés.
- Étapes de définition des besoins (production du cahier des charges) :
  - décrire les cas d'utilisation,
  - décrire les exigences non fonctionnelles,
  - décrire les contraintes,
  - modéliser les données,
  - définir les exigences fonctionnelles,
  - passer en revue les spécifications d'exigences,
  - développer, si nécessaire, des maquettes,
  - développer, si nécessaire, des prototypes d'une partie du futur système,



- spécifier précisément les exigences fonctionnelles dans le cahier des charges,
- faire valider le cahier des charges.
- Étapes transverses :
  - gérer les demandes de changement des exigences,
  - gérer l'équipe de définition des exigences,
  - planifier et gérer les groupes de travail de recueil, d'analyse ou de validation des exigences,
  - gérer le projet (coûts, délais, relation avec le donneur d'ordres).

### Check-list

---

La liste des tâches à effectuer vue au paragraphe précédent peut servir de check-list lors de l'établissement d'un plan-projet pour l'élaboration d'un cahier des charges.



# Définir le concept et les objectifs

Cette étape préalable de définition du concept et des objectifs est cruciale pour la suite. C'est un miniprocessus de définition des exigences, en plus dense, plus rapide, et au plus haut niveau. Son livrable est une sorte de cahier des charges en concentré. Il conditionne la réussite du projet. Elle doit donc être menée avec le plus grand soin.

### Une activité préalable indispensable

La première activité à laquelle doit s'attaquer l'équipe en charge de définir les besoins consiste à définir le concept et ses objectifs du produit, les objectifs du cahier des charges lui-même, ainsi que les exigences au plus haut niveau. Ces tâches seront menées en parallèle, car elles sont pratiquement indissociables. En effet, elles permettent de répondre aux quatre questions suivantes :

- Concept du produit : en quoi consistera le produit ? De quoi sera-t-il fait ? En quoi se distinguera-t-il des produits existants, qu'ils soient concurrents ou complémentaires ?
- Objectifs du produit : quelle utilisation sera faite du produit ? Dans quel but ? Pour servir qui ? Pour servir à quoi ? Pour gagner quoi ?
- Objectifs du cahier des charges : que veut-on faire du cahier des charges ? Choisir un produit sur étagère, développer une application spécifique, acquérir un progiciel, le paramétrer et le déployer ?
- Exigences au plus haut niveau : quelles sont les quatre ou cinq grandes fonctions que le produit doit remplir ? Quelles sont les deux ou trois exigences non fonctionnelles (qualité, performance) auxquelles le produit doit répondre en priorité ? Quelles seront éventuellement les contraintes techniques incontournables ?

Cette étape fait déjà appel aux techniques classiques de recueil, d'analyse, de spécification et de validation des exigences, mais étant donné son caractère stratégique, nous décrivons ces techniques spécifiquement adaptées à cette étape préparatoire plus en détail dans les paragraphes qui suivent.

### Objectifs, périmètre et parties prenantes

---

1. **Cadrage.** La littérature anglophone parle de document de « vision et portée » (*vision and scope*).

Lors de cette étape préalable<sup>1</sup>, il s'agit de définir :

- l'objectif ;
- le périmètre, ou champ de l'étude ;
- les parties prenantes.

Par parties prenantes nous entendons toutes les personnes ou organisations impactées (positivement ou négativement) par l'introduction du système ou susceptibles d'influencer son choix, son développement ou son déploiement.

Ces trois activités sont fortement interdépendantes (figure 6-1). Toutes les exigences formalisées devront être alignées sur l'objectif précédemment défini. Mais qui va exprimer l'objectif ? En grande partie, le donneur d'ordres. Cependant, en tant que personne physique, il va rarement exprimer un objectif de manière formelle. Il s'appuiera sur des experts, il devra tenir compte d'autres parties prenantes (actionnaires, associations, syndicats, sociétés savantes...). Or, l'influence de ces parties prenantes sur l'objectif dépend du périmètre.

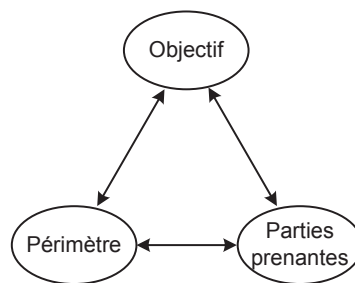


Figure 6-1 : Trois activités préalables indissociables

Ces trois activités sont présentées dans les paragraphes qui suivent. Le document résultant de ces activités est un document de cadrage.

## Recueillir les objectifs

---

### Les objectifs : des exigences au plus haut niveau

Les objectifs du système (à choisir, à développer ou à mettre en œuvre) ne sont rien de plus que des exigences de haut niveau.

Voici un exemple d'objectif, pour un logiciel de prescription de médicaments :

Assurer la sécurité, la traçabilité et la qualité des prescriptions conformément à l'arrêté du 31 mars 1999.

Cet objectif contient en réalité trois exigences de très haut niveau (sécurité, traçabilité et qualité des prescriptions de médicaments). Chacune d'elles sera déclinée en une multitude d'exigences fonctionnelles (quel scénario suit le médecin qui prescrit, quelles informations doivent être saisies, comment le logiciel vérifie les interactions médicamenteuses) et non fonctionnelles (le temps d'affichage du nom du médicament, la précision des informations affichées) ainsi que des contraintes techniques.

Comme on se l'imagine aisément, les objectifs d'aussi haut niveau doivent être formulés avec le plus grand soin, puis validés au plus haut niveau hiérarchique possible, car ils conditionnent des pans entiers du système à venir.

### Préciser et formaliser l'objectif

Quelle sera la finalité du système ? S'agit-il de choisir un logiciel sur étagère, ou bien de développer un logiciel spécifique ? La forme, le contenu et la structure du futur cahier des charges en dépendent, et donc la façon dont il sera élaboré. Le coût de son élaboration peut varier dans un rapport de un à dix, le coût du logiciel installé de un à cent. Il est donc de première importance de préciser, en collaboration avec le donneur d'ordre, l'objectif à atteindre.

Cet objectif sera formulé sur une page maximum, et approuvé par le donneur d'ordre.

On peut formaliser l'objectif sous une forme finalité-avantage-mesure. Par exemple :

- finalité : assurer la sécurité des prescriptions de médicaments ;
- avantage : diminution du nombre d'erreurs médicamenteuses ;
- mesure : diminution de 20 % du nombre d'erreurs.

En fonction du contexte, on pourra conserver cette forme de présentation dans le document, ou se servir de cette formulation uniquement pour recueillir l'objectif auprès du donneur d'ordres et préférer une présentation plus littéraire pour le document.

## Techniques de recueil : réunions et interviews

Les techniques pour recueillir ces objectifs et besoins à haut niveau sont les réunions et les interviews. Il peut s'agir d'une réunion de lancement à l'initiative du donneur d'ordre, de réunions avec les représentants des utilisateurs, ou d'interviews individuelles.

Quand on est dans le flou, que les rôles et responsabilités sont mal définis, que les objectifs ne sont pas clairs, l'interview est la technique la plus efficace. À la fin de l'interview, on demandera à la personne interviewée quelles autres personnes sont susceptibles de nous fournir des informations complémentaires. Cette technique permet de définir de proche en proche les objectifs et le concept, en croisant les informations obtenues au cours des différentes entrevues.

## Déterminer le périmètre

---

### Le domaine d'application : l'utilité du produit

À qui et à quoi servira le futur système, et pour quoi faire ? Ces questions vont déterminer le domaine d'application. Il est important de le connaître, car il nous permettra de déterminer quels sont les principaux acteurs ; par exemple, les experts du domaine chez qui on va recueillir des besoins, et les futurs utilisateurs.

Le domaine large (par exemple, bancaire, santé, industrie automobile) est simple à déterminer. Mais il s'agit d'aller un peu plus dans le détail, en partant de l'objectif. Prenons un exemple : si l'objectif est de « gérer de manière centralisée les demandes de subventions des jeunes cinéastes expatriés », on s'aperçoit que l'on aura affaire, non à un, mais à plusieurs experts de domaines d'expertise très différents.

Si l'on ne veut pas tomber dans le piège de la décomposition fonctionnelle (description des différentes fonctions du produit), il est essentiel de se concentrer sur la question « à quoi sert le produit, à qui, pour quoi faire ? ». On a obtenu, lors de la détermination des objectifs, des réponses comme : gérer au quotidien les demandes de subvention. Savoir où et par qui sont gérées des demandes de subvention de ce type permettra de délimiter le domaine d'application.

### Le périmètre : les limites du produit

À ce stade, il n'est pas question de décrire le système à l'étude lui-même. Ce qu'il est possible de déterminer, c'est son périmètre : quelles sont les limites du produit, et quelles sont les interfaces qui permettent de communiquer avec l'extérieur. Ce monde extérieur est fait de logiciels, de

matériels et d'utilisateurs avec lesquels le système à l'étude va communiquer.

Il est essentiel de décrire le périmètre avec le plus grand soin, sous forme textuelle. Ce texte, qui devra faire l'objet d'un consensus entre toutes les parties prenantes, sera soumis à l'approbation du donneur d'ordre et des principales parties prenantes.

Décrire le périmètre est un exercice difficile et un investissement très rentable. L'élaboration de ce texte va permettre, non seulement de clarifier les concepts, mais également de découvrir des contraintes, de déterminer par la suite les parties prenantes, et de préciser l'objectif.

### Le diagramme de contexte

Le diagramme de contexte est un outil de communication intéressant, et il peut être élaboré dès cette première étape, quitte à être affiné par la suite. Un diagramme de contexte n'est ni plus ni moins qu'un diagramme de flux à un niveau très macroscopique, où le système à l'étude est au centre.

Ce diagramme peut être utilisé pour décrire l'existant (diagramme de contexte métier) ou pour situer le futur système dans ses interactions avec le monde extérieur (diagramme de contexte produit). Outre sa valeur pédagogique comme outil de communication, il sera très utile lors de l'étape de recueil des exigences pour déterminer les cas d'utilisation (*use cases*).

La figure 6-2 représente le diagramme de contexte (ici simplifié) pour un système de prise en charge médicamenteuse pour un centre hospitalier.

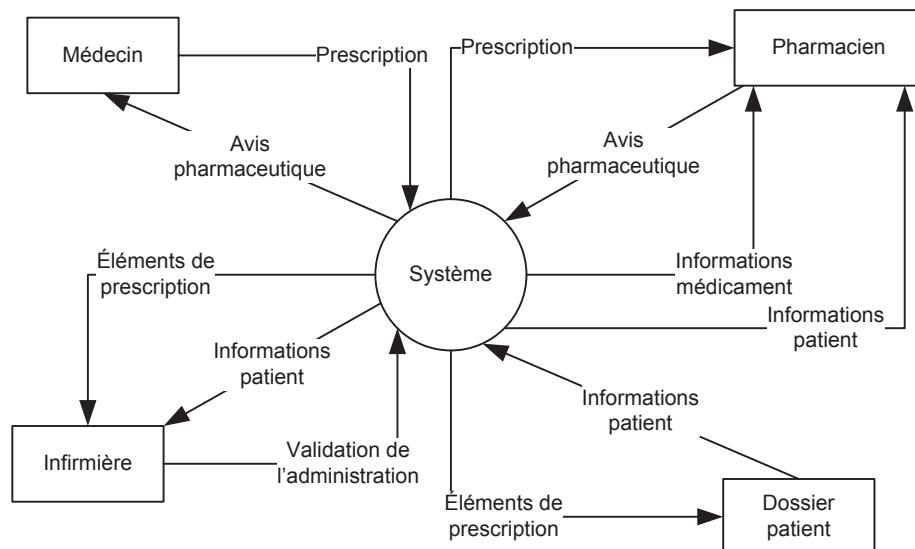


Figure 6-2 : Diagramme de contexte

## Analyser les parties prenantes

---

### Savoir qui a intérêt à quoi et quand

---

2. Partie prenante :  
en anglais, *stakeholder*.

---

On appelle partie prenante<sup>2</sup> toute personne, groupe de personnes ou organisation qui a un intérêt (positif ou négatif) dans le projet ou le produit.

Qui paye pour le développement, la mise en service, l'exploitation du futur logiciel ? Qui va le réaliser ? Qui va l'utiliser ? Qui va conduire le projet de réalisation ? Et... qui va essayer de torpiller le projet à tout prix ? Il est important de le savoir d'avance.

On identifiera les personnes ou les profils (par exemple, assistante administrative) ayant un intérêt dans l'élaboration du cahier des charges ou dans les applications qui en font l'objet, le rôle que ces personnes pourront jouer pour l'élaboration des cahiers des charges (par exemple, expert, conseiller, relecteur, futur utilisateur), leurs rôles, intérêts et objectifs.

Cette étape d'analyse a une faible visibilité à court terme et un fort impact à long terme, ce qui fait qu'il est facile de la rater. À défaut de cette analyse, on court le risque de se focaliser sur quelques partenaires, de passer à côté de besoins importants, et d'occulter des pans entiers du cahier des charges. L'analyse des parties prenantes va permettre, en temps voulu, de fixer les priorités entre des besoins très divers et parfois contradictoires, d'arbitrer, de gérer les conflits, et surtout, de connaître le véritable objectif, de savoir si (et quand) on s'en éloigne, de manière à corriger les écarts.

### Liste des parties prenantes

Une première liste des parties prenantes a été établie lors de la planification du projet de définition des besoins. Il s'agit maintenant de dresser une liste complète et détaillée et de n'oublier personne. La liste peut être très différente d'un projet à l'autre. Voici à titre indicatif une liste, à affiner pour chaque projet :

- le donneur d'ordres, ou maîtrise d'ouvrage stratégique, propriétaire du produit, c'est-à-dire celui qui paye pour le produit ;
- la maîtrise d'ouvrage opérationnelle, qui représente les utilisateurs ;
- les utilisateurs, qui vont travailler avec le produit, directement ou indirectement (il faudra détailler les différents profils) ;
- l'assistant à maîtrise d'ouvrage, analyste ou consultant, qui va recueillir, analyser et spécifier les besoins, et interagir avec l'équipe de développement ;
- les concepteurs, architectes, réalisateurs, qui vont recevoir le cahier des charges et devront l'interpréter et le réaliser ;
- l'équipe de test et de validation, qui va vérifier que le produit développé respecte le cahier des charges ;



- les personnes chargées de rédiger la documentation du produit, conformément aux spécifications ;
- les concepteurs et réalisateurs d'autres produits, qui vont interagir avec le système à l'étude ;
- les personnes chargées du support technique ou fonctionnel du produit à choisir ou à développer ;
- le marketing du produit ;
- les juristes ;
- les organismes de normalisation (leurs représentants) ;
- les experts métier ;
- les experts techniques.

Cette liste n'est pas exhaustive. Elle sera nominative, du moins en ce qui concerne les parties prenantes à fort pouvoir de décision.

### La grille impact-pouvoir-influence

Après avoir dressé la liste des parties prenantes, on doit évaluer le pouvoir, l'intérêt et l'influence de chacun d'eux, de manière à mieux orienter les efforts de recueil des exigences. Il y a plusieurs manières de représenter l'influence des parties prenantes sur le projet (grilles pouvoir-influence, pouvoir-impact, pouvoir-intérêt, etc.) dont chacune a ses avantages. La manière la plus utile semble être la grille impact-pouvoir-influence :

- Impact : dans quelle mesure la personne, le groupe ou le profil utilisateur sera-t-il impacté (dans son travail, dans son pouvoir) par l'introduction du système ?
- Pouvoir : quel est son pouvoir d'influence ?
- Influence : cette influence sera-t-elle positive, négative ou neutre ?

Par exemple, le donneur d'ordres a beaucoup de pouvoir, une influence positive sur le projet et sera généralement fortement impacté. Une association professionnelle ou un syndicat peuvent avoir beaucoup de pouvoir (positif ou de nuisance) sans être directement impactés. A contrario, un utilisateur final a généralement beaucoup d'intérêt et peu de pouvoir (figure 6-3).

L'utilité d'une telle grille est multiple :

- connaître les intérêts de chaque partie prenante, ce qui permet déjà de détecter des exigences ;
- en fonction de l'impact et du pouvoir, avoir une première idée des priorités entre exigences ;
- détecter les risques potentiels, de manière à les limiter ;
- savoir qui informer, quand, et de quelle manière ;
- détecter les freins à l'avancement du projet.

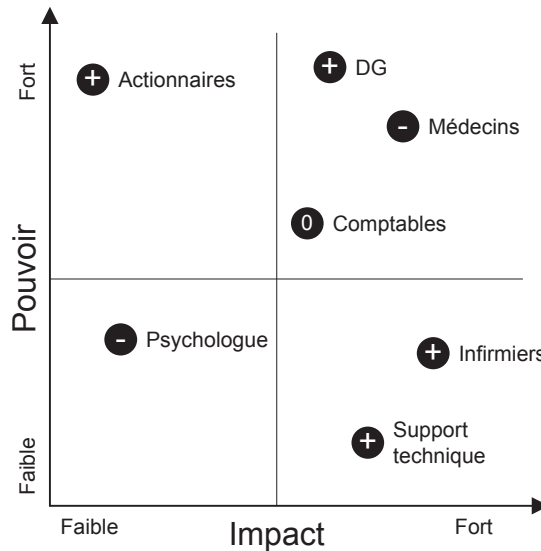


Figure 6-3 : Grille impact-pouvoir-influence

## Grille de questionnement

La grille de questionnement suivante peut servir de trame (par exemple, lors d'une interview) pour la détermination des objectifs, du contexte, et des parties prenantes, et pour anticiper les techniques à utiliser :

- **Demandeur.** Qui est à l'origine de la demande ?
- **Motivation.** Qu'est-ce qui, d'après vous, a motivé la demande ?
- **Objectif.** Quel est l'objectif de l'application ? Quel est l'objectif du cahier des charges ? Que veut-on obtenir ? Quel problème veut-on résoudre ?
- **Critères.** Comment saurons-nous que l'objectif est atteint ?
- **Bénéfices attendus.** Qu'espère-t-on gagner avec ce projet ?
- **Acteurs.** Qui va utiliser le produit ? Qui va payer ? Qui va développer, maintenir, exploiter ? Quels sont les décideurs ?
- **Contenu.** Que va-t-on mettre dans le cahier des charges ? Que va-t-on exclure : les fonctions ? les exigences non fonctionnelles (qualité, performances) ? les contraintes techniques ? les contraintes de projet ? les contraintes d'exploitation ?
- **Portée.** Quelles fonctions va-t-on décrire dans le cahier des charges (portée fonctionnelle) ? Qu'est-ce qui sera englobé par le cahier des charges : l'entreprise entière, le système, un composant du système (portée de conception) ?

- **Niveau.** À quel niveau d'observation va-t-on se placer : un niveau stratégique, au niveau utilisateur, ou au niveau très détaillé ? Quel niveau de détail ou d'abstraction veut-on obtenir ?
- **Usage.** Que veut-on en faire du cahier des charges : choix de progiciel, développent spécifique... ?
- **Parties prenantes.** Qui sera le correspondant de la personne chargée de rédiger le cahier des charges ? Quelles seront les autres parties prenantes ?

On adaptera ces questions au contexte et aux interlocuteurs. On pourra poser toutes les questions à chaque interlocuteur, de manière à se faire une idée des demandes, mais il est évident que le poids des réponses sera différent en fonction des interlocuteurs. Les utilisateurs ne sont pas les payeurs.

Plus que jamais, la difficulté de cet exercice de questionnement est de ne pas trop descendre dans les détails. Il ne s'agit pas, dans cette étape préliminaire, de faire le cahier des charges par anticipation, mais d'apporter une « vision », une conceptualisation de ce que sera le futur système.

### Tableau des parties prenantes

Grâce à la grille impact-pouvoir-influence et suite aux informations recueillies au moyen de la grille de questionnement, on peut établir le tableau des parties prenantes. La grille du tableau 6-1 est donnée en exemple. Notons que les rubriques des différentes colonnes peuvent varier selon le contenu, la portée, le niveau et l'usage du cahier des charges (voir paragraphe précédent).

**Tableau 6-1 – Grille des parties prenantes**

Partie prenante	Rôle	Objectifs	Problématique	Besoins particuliers	Apport d'expertise
Actionnaire	Finance le projet	Efficience	Coût du développement		Non
DG	MOA stratégique		Respect de la réglementation		Non
Médecins	Utilisateurs directs Prescripteurs	Prescrire sans perte de temps Aide à la prescription	Erreurs médicamenteuses	Facilité d'utilisation	Oui
Infirmiers	Utilisateurs directs Administrent les médicaments			Disponibilité de l'application	Oui

## Le document de cadrage

---

Il est important de formaliser l'objectif, le champ de l'étude et le tableau des parties prenantes dans un document unique, et faire valider ce document par les parties prenantes les plus importantes (les décideurs).

Une fois validé, un tel document a plusieurs vertus :

- C'est un cahier des charges en condensé.
- C'est une aide à la gestion de projet : il permet de chiffrer l'effort nécessaire pour élaborer le reste du cahier des charges, à trouver les experts, à planifier.
- Il peut faire fonction de lettre de mission pour l'analyste.

Le document de cadrage est le livrable de la phase de préparation du processus. Il contiendra donc les éléments suivants :

- le concept ;
- les objectifs ;
- les parties prenantes : rôles, responsabilités ;
- les catégories (classes) d'utilisateur ;
- les contours du futur produit et échanges avec l'extérieur : diagramme de contexte ;
- les cas d'utilisation métier (*business use cases*) dans leurs grandes lignes, par priorité ;
- les cas d'utilisation qu'on a décidé d'informatiser.

## Check-list

---

La check-list suivante permet de s'assurer que les conditions sont réunies pour réussir le projet, qu'il s'agisse de choisir et mettre en œuvre un progiciel ou de développer une application. Si toutes les conditions ne sont pas réunies, élaborer un cahier des charges sera de peu d'utilité.

- Les objectifs sont-ils clairement spécifiés ?
- A-t-on identifié, analysé et formalisé la liste des parties prenantes ?
- Les objectifs spécifiés sont-ils approuvés par les parties prenantes ?
- Les objectifs sont-ils réalistes, en termes de délais, de charge, de budget et aussi de risques ?
- A-t-on analysé les risques ?
- Y-a-t-il consensus entre parties prenantes sur le périmètre ?
- Le périmètre a-t-il été formellement approuvé par le donneur d'ordres ?
- Les parties prenantes se sont-elles engagées à participer ?

# Planifier le projet d'élaboration

Élaborer un cahier des charges est un projet en soi. Dans ce cadre, il nécessite une véritable planification. Après avoir défini les parties prenantes, le champ et l'objectif, nous avons en main tous les éléments pour élaborer le plan projet. Celui-ci consiste à déterminer les techniques, méthodes et outils à mettre en œuvre pour mener à bien le projet d'élaboration du cahier des charges, avec un maximum d'efficacité pour un minimum d'effort. Le livrable de cette étape préalable est un plan projet.

## Le plan projet

---

Ce document vient en complément du document de cadrage et sert de feuille de route à l'équipe d'élaboration du cahier des charges. L'ensemble forme le « cahier des charges du projet de cahiers des charges ».

Le plan projet contient un devis estimatif des coûts et délais, ainsi qu'une description brève (une ou deux pages) de l'organisation et de la méthode de travail : taille de l'équipe, moyens à mettre en œuvre, formations à prévoir, circuit de l'information, fréquence des réunions, validation des documents, en fonction d'un certain nombre de paramètres (taille fonctionnelle, maille du cahier des charges...).

Cette manière descendante (*top-down*) de procéder présente plusieurs avantages :

- À la fin de cette étape, vous disposez de tous les outils nécessaires à l'élaboration du cahier des charges.
- Vous avez une idée précise des paramètres du projet : coût, délais, qualité, ressources nécessaires.
- Vous pouvez ainsi négocier avec le donneur d'ordres sur des bases solides.

Les paragraphes qui suivent détaillent les activités de cette étude préliminaire : cadrer la méthodologie et élaborer le plan projet.

## Cadrer la méthodologie

---

### Définir le processus global

Le processus de recueil des besoins et d'élaboration du cahier des charges peut varier d'un cas à l'autre. Une bonne pratique consiste à décrire ce processus et l'enchaînement de chacune de ses activités (recueil, analyse, spécification, validation, gestion) selon un schéma de processus détaillé. Cette étape est utile, voire indispensable, lorsque l'élaboration du cahier des charges est faite par un consultant externe. Dans ce cas, il doit décrire schématiquement sa méthode de travail dans sa proposition technique et commerciale.

### Choisir les modèles de représentation à utiliser

---

1. Diagrammes : voir au chapitre 10.

---

On choisira les modèles<sup>1</sup> de représentation des exigences qui seront utilisés pour l'élaboration des cahiers des charges (par exemple, cas d'utilisation, diagrammes de séquence...). Le choix se fera en fonction du domaine (technique, administration, banque...), des normes et standards dans l'entreprise, mais aussi en fonction de la sociologie des parties prenantes. En effet, en fonction des milieux professionnels, les personnes sont plus ou moins sensibles à une représentation plutôt qu'à une autre.

### Déterminer et adapter les techniques

---

2. Ces techniques sont décrites dans les chapitres 7 à 14.

---

Il n'y a pas une méthode unique de gestion des exigences. Les techniques<sup>2</sup> classiques d'acquisition, d'analyse, de spécification, de validation et de gestion des exigences devront être adaptées au contexte, aux interlocuteurs, au type de système à l'étude.

### Étudier les outils

---

3. Outils : voir au chapitre 19.

---

On fera une analyse comparative d'outils<sup>3</sup> d'aide à l'élaboration de cahiers des charges et de gestion des exigences. Il s'agira, dans cette étude de définition, de faire un choix d'outils, ou simplement de mieux connaître l'offre du marché, de manière à mieux adapter les techniques, déterminer les profils des intervenants, anticiper sur les coûts de formation, les charges d'élaboration.

### Élaborer les documents types

Pour chaque modèle de représentation des exigences choisi, on élaborera le modèle de document approprié (modèle générique) ; par exemple, modèle de cas d'usage (*use case*).

Cet ouvrage donne plusieurs modèles de documents types, ainsi que des adresses Internet utiles permettant d'en télécharger.

### Identifier et adapter les modèles

Il s'agit de déterminer, avec la collaboration des parties prenantes les plus impactées, la structure<sup>4</sup> du document final, son niveau (par exemple, niveau stratégique, niveau de l'utilisateur, ou niveau des sous-fonctions), sa portée (organisation, système, sous-système, composant), son contenu dans les grandes lignes (en dehors des exigences fonctionnelles, les autres exigences abordées).

On élaborera le modèle de document (le *template*) de cahier des charges.

---

4. Modèles de cahiers des charges : voir au chapitre 14.

---

## Élaborer le plan projet

---

### Identifier les profils utilisateurs

Cette activité fait suite à l'analyse des parties prenantes en la détaillant en ce qui concerne les profils utilisateurs. On identifiera et analysera les profils utilisateurs types pour le logiciel qui fera l'objet de cahiers des charges, leurs attentes, leurs besoins génériques, les conflits potentiels entre types d'utilisateurs, leur niveau de maîtrise des technologies, etc. Cette activité sera affinée lors de l'étape de recueil des besoins.

### Établir la liste des sources d'exigences

Les exigences qui seront formulées dans les cahiers des charges proviendront de sources diverses : études en amont, études théoriques, étude de la concurrence, ... et, bien sûr, les besoins des différentes parties prenantes, en particulier les utilisateurs.

On établira dans un premier temps la liste des sources possibles d'exigences : tableau comprenant les sources d'exigences, leur utilité, les techniques de recueil d'exigences, les ressources nécessaires, la charge de travail relative. Cette liste sera affinée et détaillée lors de l'étape de recueil.

### Estimer les charges et les délais

Comme pour tout projet, l'estimation des charges et des délais est un exercice difficile, à l'issue incertaine, mais néanmoins indispensable. La charge d'élaboration dépend de plusieurs paramètres :

- la facilité de recueil des besoins : les sources d'exigences et les techniques à utiliser en fonction des sources et de l'objectif à atteindre ;
- la taille fonctionnelle du logiciel à définir : nombre de fonctions et leur importance ;
- la destination du cahier des charges : choix d'un progiciel ou développement d'un logiciel spécifique ;
- le niveau de détail avec lequel on veut décrire le produit ;
- la maturité de l'équipe de développement ;
- l'habitude de collaborer entre maîtrise d'ouvrage et maîtrise d'œuvre ;
- l'expérience et l'habileté de l'assistant à maîtrise d'ouvrage.

Toute la suite de cet ouvrage s'attache à optimiser le temps et l'effort d'élaboration du cahier des charges, en minimisant les risques et en améliorant la qualité.

### Identifier les ressources

Un projet d'élaboration d'un cahier des charges fait appel à plusieurs compétences. Ces diverses compétences peuvent être réunies chez une seule personne et, dans de nombreux cas, un consultant seul peut animer le projet d'élaboration du cahier des charges, en réalisant plusieurs tâches : recueil des besoins, animation des groupes de travail, rédaction des spécifications, etc. Cependant, pour les projets d'envergure, il sera peut-être utile de faire appel, même ponctuellement, à des personnes différentes.

**Le directeur de la mission**, ou directeur de projet, d'assistance à la maîtrise d'ouvrage pour l'élaboration du cahier des charges. C'est le correspondant unique du client. Il définit et gère le projet et anime les interactions entre acteurs. Sans être nécessairement un expert du domaine d'application. Il suit de bout en bout les opérations d'élaboration du cahier des charges et y participe en grande partie.

**L'expert métier**. C'est un consultant expérimenté ou senior connaissant le domaine d'application, l'organisation du client. Souvent un ancien utilisateur, il connaît de préférence les applications concurrentes de celle à développer. Sans être informaticien, il connaît les éditeurs de logiciels du domaine d'application.



**Un consultant expert en gestion des exigences.** Il apporte à l'équipe expertise et conseil méthodologique. Il connaît les techniques de recueil des besoins, de modélisation, de représentation de l'information.

**Un expert technique,** qui pourra se prononcer sur les contraintes techniques.

### Identifier et analyser les risques

Une bonne pratique consiste à identifier les risques associés à l'élaboration de tous les cahiers des charges et de mettre au point des stratégies d'atténuation. Il ne s'agit pas ici d'aborder les risques inhérents à un projet de développement ou à l'exploitation d'un logiciel, mais aux risques spécifiques à l'élaboration du cahier des charges.

Parmi les risques les plus fréquents, citons l'indisponibilité de certains profils utilisateurs pour participer à des groupes de travail d'expression des besoins. Un autre risque est lié à la non-implication, ou à l'implication insuffisante, de la maîtrise d'ouvrage stratégique.

### Contenu du plan projet

Le plan projet contiendra au minimum les éléments suivants :

- description simple du processus global d'élaboration ;
- techniques de recueil à mettre en place : interviews, réunions, groupes de travail, maquettes ou prototypes, etc. ;
- modèles de représentation de l'information à utiliser ;
- outils à utiliser ;
- liste des documents types à utiliser ;
- liste des sources d'exigences ;
- représentants des utilisateurs retenus pour participer aux différents groupes de travail ;
- formations éventuellement prévues aux techniques et outils ;
- ressources nécessaires ;
- charges ;
- délais ;
- risques.

### Enregistrement des charges et des délais

Une organisation qui souhaite industrialiser le processus de développement et de gestion des exigences aura tout intérêt à établir et faire appliquer un suivi du temps passé aux différentes activités. Par exemple,

recueil, analyse, spécification, validation. Ou mieux (car plus proche de la pratique) :

- temps passé en réunions avec le maître d'ouvrage ;
- temps passé en groupe de travail ;
- rédaction et modifications du cahier des charges ;
- revues de documents.

## Check-list

---

- L'objectif du projet est-il clair, sans ambiguïté, sans langue de bois ?
- L'atteinte de l'objectif est-elle vérifiable et mesurable ?
- L'atteinte de l'objectif se traduira-t-elle par un bénéfice pour l'entreprise ou l'organisation ?
- Est-on assuré que l'objectif pourra être atteint avec les moyens alloués ?
- Y a-t-il un accord écrit sur le périmètre du projet ?
- A-t-on étudié les risques, en termes d'impact et de probabilité, et s'est-on assuré qu'ils sont raisonnables ?
- A-t-on estimé les coûts et s'est-on assuré qu'ils sont raisonnables ?
- Les parties prenantes sont-elles impliquées ?
- L'investissement prévu est-il justifié ?
- S'est-on assuré qu'il ne subsiste aucune zone d'ombre sur le processus d'élaboration du cahier des charges ?

# PARTIE 2

---

## Développement des exigences

Le projet d'élaboration du cahier des charges étant préparé et planifié, nous abordons maintenant le cœur de notre métier : le développement des exigences proprement dit. Nous décrivons les techniques et méthodes qui vont nous permettre, en partant des objectifs, de valider un cahier des charges, en optimisant les coûts, les délais et la qualité.

Souvenons-nous cependant que le processus est itératif, et que les résultats des étapes préparatoires ne sont jamais gravés dans le marbre. La planification initiale peut être remise en cause, comme pour tout projet. L'objectif et le périmètre peuvent varier. Des parties prenantes restées dans l'ombre peuvent toujours se manifester. Aussi est-il important, tout en gardant l'objectif en tête, de rester toujours à l'écoute des différents acteurs.



# L'étape de recueil

De toutes les activités de l'ingénierie des exigences, l'activité de recueil est celle qui demande le plus de qualités humaines. Il faut trouver la personne qui détient l'information, l'encourager à exprimer, puis écouter ce qu'elle a à dire, avec neutralité et bienveillance. Il faut animer des groupes de travail où des tensions entre individus peuvent apparaître. Raison de plus pour bien préparer cette étape, en laissant le moins de choses possibles au hasard.

## L'enjeu

---

Capter les besoins est a priori chose simple et facile. Il suffit de se réunir avec les futurs utilisateurs, interviewer le client, ou compiler normes et standards pour recueillir les besoins à la source. La réalité est bien loin d'être aussi si simple. À moins de vouloir faire développer une petite application simple pour une demi-douzaine d'utilisateurs (et encore...), on se trouve vite confronté à des difficultés diverses : résistances, contradictions, guerres des clans, dissensions.

Mais plus concrètement, demandez à un futur utilisateur du système d'exprimer ses besoins. Posez-lui la question « quel est le besoin ? ». Spontanément, il exprimera, selon le cas :

- un problème ou une difficulté à laquelle il est confronté ;
- une solution (plus précisément sa solution) à ce problème ;
- la manière dont cette solution sera mise en œuvre ;
- un vrai besoin, c'est-à-dire une fonction attendue du futur système.

Autrement dit, dans la majorité des cas, l'utilisateur interviewé n'exprimera pas spontanément un besoin. Quelle est donc la « bonne technique » à employer pour recueillir les besoins ? Comment aider le futur utilisateur

à s'exprimer ? Comment éviter que les réunions de travail ne s'éternisent, ou ne se transforment en pugilat verbal ? Il n'y a pas de recette miracle, mais des outils et des techniques. Le reste est affaire d'expérience et d'habileté, pour choisir le bon outil et l'utiliser à bon escient. Voici donc quelques techniques et outils, ainsi que des conseils d'utilisation.

### Le processus de recueil

Il n'y a pas de plan type de l'étape de recueil, avec un enchaînement immuable de tâches prédéfinies. Les techniques pour recueillir les exigences sont nombreuses et très variées, et le choix de telle ou telle technique dépend du contexte, de l'existant et des contraintes : la disponibilité des personnes et leur niveau d'expertise, les documents pouvant être consultés, la possibilité d'observer sur le terrain les pratiques existantes ou des systèmes analogues à celui à l'étude. Mais comme avec les autres étapes, la planification et la vérification jouent un rôle important.

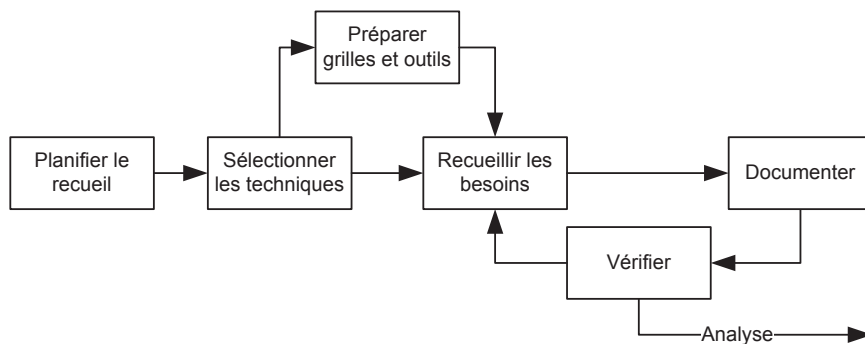


Figure 8-1 : Le processus de recueil

#### Planifier le recueil des besoins

C'est à partir de la liste des sources d'exigences et du tableau des parties prenantes que l'on va planifier dans le temps le recueil des besoins, et prévoir les ressources (humaines et matérielles) et la logistique. Cette planification doit être souple, car le volume et la qualité des informations que l'on va recueillir ne sont pas précisément connus à l'avance.

Rappelons que l'étape de recueil, en pratique, n'est pas isolée des autres. L'analyse, la spécification, et une grande partie de la validation des besoins se déroulent en parallèle avec le recueil. C'est une raison supplémentaire pour planifier soigneusement les interviews des différentes parties prenantes, et surtout les réunions des groupes de travail. Il faut souvent une semaine pour obtenir un rendez-vous, et plus d'un mois pour

réunir un groupe de travail. La lecture des documents est évidemment plus simple à planifier, mais peut être longue.

Et surtout, ne pas oublier de « prévoir l'imprévu ». Des réunions supplémentaires sont souvent nécessaires pour éclaircir des points obscurs.

### Préparer les grilles et les outils

En fonction des buts visés (développement de logiciel, ou choix d'un progiciel, par exemple), de la granularité à atteindre, des personnes à interviewer, on préparera des guides d'entretien ou de réunion.

Une partie des grilles d'interview ou des thèmes des groupes de travail devra être formalisée et envoyée par avance aux participants.

Pour recueillir des besoins à partir de documents écrits, il est également intéressant de se munir de grilles de lecture pour éviter d'être submergé par la masse de documents. La quantité d'informations à traiter peut rendre l'opération de recueil complexe, en particulier si l'on veut garder la traçabilité des exigences aux étapes suivantes.

Le principe de la construction d'une grille est simple. Il consiste à partir de l'objectif pour découvrir les exigences qui en découlent, et des exigences déjà exprimées pour découvrir les exigences plus détaillées. Nous verrons plus loin la mise en application de ce principe.

### Recueillir et documenter les besoins

Il existe une multitude de techniques de recueil des besoins, plus ou moins complexes et consommatrices de temps et de ressources. Elles sont décrites dans la suite de ce chapitre.

Quelle que soit la technique choisie, l'information recueillie doit être soigneusement répertoriée et stockée en vue d'une analyse et d'une spécification ultérieures. Cela est vrai même si recueil, analyse et spécification ont lieu au cours d'une même réunion d'un groupe de travail.

Une des difficultés découle de la surabondance des informations recueillies, et ce, quelle que soit la technique de recueil. Pour éviter de se noyer sous un flot de paroles ou sous des milliers de pages, il n'y a pas de recette miracle, mais un principe : travailler à partir de *grilles* en gardant en tête l'*objectif*.

### Vérifier les informations recueillies

On vérifiera l'exactitude des informations recueillies au moyen d'un compte-rendu de réunion ou d'interview. Il ne s'agit pas ici de valider les exigences, mais d'obtenir un feedback sur les informations recueillies. Cela permettra de corriger les informations au plus près de la source.

Si le recueil des besoins est fait par une équipe (c'est souvent le cas), il faut prévoir de faire le point pour échanger sur les informations recueillies.

### Le plan de recueil

---

Le plan de recueil servira de guide tout au long de l'étape de recueil des besoins. Il contient :

- Les objectifs du recueil : en particulier, il est important de savoir s'il s'agit de recueillir des exigences à haut niveau, ou de descendre dans les détails, ou de consolider des exigences déjà formalisées.
- Les livrables et leur forme, ainsi que leur niveau de formalisme ; cela peut aller d'une simple note sous forme textuelle, jusqu'à des modèles de données strictement formalisés.
- Les méthodes et techniques de recueil, qui peuvent varier selon les objectifs, les livrables et les interlocuteurs.
- Les risques induits, et les méthodes de contournement et d'atténuation des risques.
- La planification du recueil : coûts, dates, délais et ressources.

### Risques liés au recueil et atténuation

---

Les risques liés au recueil proviennent des sources de besoins, plus ou moins disponibles et fiables, ainsi que de la manière dont ses besoins sont exprimés.

Le premier facteur de risque est l'indisponibilité des sources, qu'il s'agisse de documents auxquels on ne peut accéder (détruits, perdus, corrompus, confidentiels) ou de personnes qui ne sont pas ou trop peu disponibles.

Les informations erronées, intentionnellement ou non, sont un deuxième facteur de risque. Les interlocuteurs se servent souvent d'une étude des besoins pour servir leurs intérêts propres, et non les objectifs visés par le système à construire. Pour éviter ce piège, il faut, d'une part, s'appuyer sur les objectifs préalablement formalisés et, d'autre part, croiser les sources.

En troisième lieu viennent les besoins non exprimés, ou exprimés de manière floue. C'est pour cette raison qu'il est utile d'utiliser les compétences d'un expert métier, qui maîtrise le vocabulaire de ses interlocuteurs. De plus, il est indispensable, pour recueillir des besoins opérationnels, de descendre au plus bas niveau hiérarchique possible, de consulter de vrais utilisateurs, et ne pas se contenter des seuls « représentants » des utilisateurs.



Un autre facteur de risque provient de la difficulté de se faire une idée d'ensemble d'un besoin. Interroger un seul utilisateur apportera une image déformée du besoin, et trop peu d'informations sur les pratiques quotidiennes. Inversement, avec trop de sources, trop d'interlocuteurs, il est difficile de gérer les contradictions sur le fond et sur la forme. Chercher à savoir « qui a raison » pousse à descendre dans les détails, et c'est là un risque supplémentaire.

Recueillir des besoins trop ou insuffisamment détaillés, par rapport aux objectifs du cahier des charges, est encore un risque. C'est pour cela qu'il est indispensable de connaître par avance les objectifs et le niveau de détail recherché. Des techniques de recueil, que nous verrons plus loin, permettent d'élever le niveau, ou au contraire, de descendre plus dans le détail.

## Détermination des profils utilisateurs

---

Après le donneur d'ordres, qui a exprimé l'objectif, les parties prenantes qui seront les plus impliquées sur le plan opérationnel dans le recueil des exigences sont les futurs utilisateurs du système.

### La typologie des profils

Pour mener à bien le recueil, il est nécessaire de déterminer les profils des utilisateurs, et de les classer en catégories, de manière à pouvoir, par la suite, déterminer les besoins par catégorie d'utilisateurs. Il y a plusieurs façons d'établir les profils utilisateurs, en fonction de leur métier, de leur position hiérarchique ou de leur emplacement géographique. Cependant, le plus efficace est de les grouper en fonction de leur interaction avec le système. Ce qui nous intéresse ici n'est pas l'individu, mais son rôle vis-à-vis du système, sachant qu'un individu peut jouer plusieurs rôles (par exemple, dans un restaurant, prendre les commandes et s'occuper de la comptabilité) et donc avoir plusieurs profils.

Les profils utilisateurs seront constitués en fonction :

- des fonctions du système auxquelles ils font appel ;
- du niveau de sécurité requis pour accéder à certaines fonctions ;
- de leurs propres exigences de sécurité, de fiabilité ou d'ergonomie ;
- de leur fréquence d'utilisation du système ;
- de leur pratique de systèmes analogues ;
- de leur connaissance du domaine.

### Un exemple

Un centre hospitalier a besoin de choisir et de mettre en œuvre un système de gestion du médicament. Les profils utilisateurs suivants ont été définis :

- prescripteur : médecin, sage-femme, ou toute autre personne amenée à prescrire des médicaments ;
- pharmacien ;
- préparateur en pharmacie ;
- infirmier ;
- gestionnaire du système.

Il n'y a évidemment pas une correspondance biunivoque entre un groupe de personnes et un profil utilisateur. Le profil « prescripteur » peut être attribué à deux personnes de métiers très différents (par exemple, un médecin et une sage-femme). Inversement, le pharmacien d'un hôpital peut, dans certains cas, tenir le rôle de gestionnaire du système.

Le tableau 8-1 des profils reprend, de manière plus détaillée, le tableau des parties prenantes qui a été élaboré lors de l'étape préparatoire. Il pourra être encore enrichi au fil de l'eau lors de l'étape de recueil.

**Tableau 8-1 – Grille des utilisateurs**

	Fonctions utilisées	Exigences particulières	Fréquence	Pratique
Médecin	Prescription	Facilité d'apprentissage	Quotidienne	Assez importante
Pharmacien	Validation pharmaceutique		Quotidienne	Très importante
Préparateur	Saisie de la dispensation du médicament		Quotidienne	Faible
Infirmier	Saisie de l'administration du médicament	Tablette mobile	Quotidienne	Faible
gestionnaire	Paramétrage		Mensuelle	

### Recherche des sources d'exigences

Les parties prenantes, et en particulier les futurs utilisateurs, sont la première source d'exigence. Tous n'ont pas le même domaine de

compétences, ni le même niveau d'expertise. Le tableau des parties prenantes va à nouveau nous permettre de trouver les personnes-ressources, mais cette fois il devra être utilisé pour élaborer un tableau nominatif. Toutes les personnes d'un même profil n'ont pas le même niveau d'expertise, ni la même facilité d'expression. Attention cependant : les opposants au projet ont parfois autant d'informations utiles à nous apporter que les plus moteurs. Il faudra les inviter à s'exprimer, en réunion ou en interview, et les écouter avec bienveillance.

Les articles, études comparatives, études de marché, forums sur Internet constituent également une bonne source. Ces informations ne sont pas toujours fiables, ni toujours à jour. Il faudra donc croiser les informations entre elles, et avec des sources plus sûres.

Les cahiers des charges existants contiennent souvent des exigences qui peuvent être reprises, après éventuelle mise à niveau. Si le cahier des charges est bien fait, la reprise peut être très simple.

On peut remonter au cahier des charges à partir de spécifications fonctionnelles générales ou détaillées d'un logiciel existant (voire d'un logiciel qui n'a jamais vu le jour), ou directement en observant le comportement du produit.

Les « sous-produits » sont également une mine d'informations pouvant être réutilisées, après analyse, pour élaborer les exigences : fiches d'anomalie, rapports d'essais, rapports au *help-desk*... en particulier, les fiches d'anomalie sont très utiles pour recueillir des exigences « négatives » ou, pour formuler les choses plus positivement, des exigences d'amélioration de l'existant. Il en est de même des rapports d'audit, des fiches de réclamations, et des demandes de modification, pour examiner ce qui peut être amélioré et ce qui manque.

Les normes, qui constituent à elles seules un recueil d'exigences, sont évidemment une source fiable. Il est nécessaire de faire attention à la date de publication, de s'assurer pour chaque norme qu'elle s'applique au contexte du système à l'étude. Il est à noter que les normes ne sont pas toujours compatibles entre elles, ni cohérentes sur le vocabulaire.

Enfin, la documentation d'un logiciel existant (à remplacer), ou celle des concurrents sont des sources à ne pas négliger.

## Techniques de recueil

---

La capture des exigences nécessite une bonne dose de créativité. On entend par là la créativité opérationnelle, et non artistique. Celle d'un commandant d'unité sur le théâtre des opérations. À condition d'être

orienté vers l'objectif et de respecter la doctrine, vous avez le droit, et même le devoir, d'être inventif et d'improviser.

Il en est de même pour la capture des exigences. Tant que vous avez l'objectif en vue, que vous respectez les bonnes pratiques, que vous vous en tenez à votre lettre de mission, vous avez la liberté d'utiliser toutes les techniques connues de recueil des besoins, et d'en inventer d'autres si nécessaire. Les techniques doivent être adaptées aux personnes et aux circonstances. Ainsi, rien ne vous empêche d'organiser un brainstorming avec certains utilisateurs et des interviews individuelles avec d'autres. Lorsqu'il s'agit de choisir outils et techniques, c'est l'expérience et l'intuition qui doivent servir de guides, et non les procédures.

Il n'y a pas en la matière de recette miracle. Mais il y a des techniques. Le reste est une affaire d'expérience, pour choisir la bonne technique et l'utiliser au bon moment. On trouvera dans les paragraphes qui suivent quelques techniques qui ont fait leurs preuves, à commencer par les plus classiques, ainsi que leur mode d'utilisation.

## L'analyse de documents

---

### Utilité de la technique

L'analyse de documents est un des moyens les plus directs de recueillir des exigences. Les documents existants constituent une des principales sources disponibles d'exigences, dont certaines sont pour ainsi dire « prêtes à l'emploi ». La difficulté provient en général de l'abondance des documents, qu'il faut soigneusement trier, et dont il faudra filtrer l'information utile.

Utiliser des informations préexistantes permettra de réduire le nombre et la durée des interviews et réunions, mais pas de les supprimer. Elle permettra également d'acquérir des connaissances sur le domaine d'application, les interviews ultérieures étant un moyen d'affiner et de consolider ces connaissances. Cela évitera aussi de perdre la face en posant des questions dont on peut trouver la réponse dans des documents disponibles.

### Les sources

Les sources sont en général abondantes (voire surabondantes) :

- cahiers des charges et spécifications de produits similaires ou concurrents, de versions antérieures du système à l'étude ;
- normes : sources de règles de gestion et de contraintes techniques ;

- procédures, descriptions partielles de processus métier ;
- documents de travail, comptes-rendus de réunions : sources de besoins, contraintes, frustrations des utilisateurs, priorités ;
- documentation technique : source de contraintes ;
- documentation conceptuelle ou théorique, source de spécifications innovantes ;
- formulaires papier : très utiles pour connaître les informations à saisir, celles devant apparaître dans un document de sortie ;
- fiches d'anomalie, demandes de modifications.

### Mode opératoire

Le mode opératoire est le suivant :

1. Commencer par lister les documents existants relatifs au domaine et/ou au système à l'étude.
2. Passer les documents rapidement en revue, de préférence en équipe, et noter leur contenu et leur utilité.
3. Faire examiner les documents par un ou plusieurs experts, en fonction de leur contenu et de l'appétence des experts pour la question.
4. Trier les documents : décider des documents à reprendre en l'état, ceux à retravailler, des parties de documents à extraire et conserver.
5. Dès que possible, valider la pertinence, la cohérence, la « fraîcheur » des informations fournies.
6. Maintenir un tableau de la documentation existante.

## La réunion d'un groupe de travail

---

### Utilité et difficulté

La méthode « naturelle », qui consiste à réunir des volontaires des différentes parties prenantes et à leur demander d'exprimer les besoins, est aussi la moins efficace. Ainsi, en réunissant vingt personnes de métiers très différents, en mélangeant utilisateurs, donneurs d'ordres, maîtres d'ouvrage, et autres parties intéressées, on risque de ne recueillir que peu d'informations, et très peu d'exigences exploitables. D'une part, par ce qu'au-delà de dix ou douze participants, une grande partie d'entre eux sera inactive et risquera même de perturber les autres, et d'autre part, par ce qu'il est difficile de défricher le terrain, d'écouter tous les besoins, et d'obtenir un consensus dans un milieu trop hétérogène.

### Les groupes de travail organisés

Une méthode des plus efficaces consiste à réunir les parties prenantes par profil : une première réunion avec le maître d'ouvrage stratégique, puis une réunion avec des représentants des utilisateurs d'un métier, puis d'un autre métier. Une fois que les besoins des différents contributeurs seront recueillis, on programmera une réunion « mixte » avec le représentant de chaque métier ou profil utilisateur.

De tels ateliers<sup>1</sup> peuvent également réunir maîtrise d'ouvrage et maîtrise d'œuvre, ou utilisateurs et développeurs, dans un même lieu.

L'animation de ces ateliers de travail, qui peuvent durer plusieurs jours, n'est pas chose aisée. Au moins deux animateurs sont nécessaires : l'analyste ou assistant à la maîtrise d'ouvrage, et une personne chargée de prendre des notes et de les organiser.

Les ateliers sont sans doute le moyen le plus puissant de recueillir les besoins, mais ils demandent un important travail de préparation, de la part de tous les participants, mais surtout de la part de l'animateur. De plus, certaines règles doivent impérativement être respectées :

- choisir avec soin un nombre limité de participants (quatre à cinq, plus les animateurs) ;
- faire respecter une bonne discipline de travail à tous les participants : arriver à l'heure, éteindre son téléphone mobile, éviter toute conversation en aparté, respecter les divers avis ;
- rester aligné avec les objectifs tels qu'ils ont été formalisés ;
- maintenir la discussion au bon niveau de détail : éviter de descendre trop dans les détails, ou au contraire de remettre en cause les objectifs ;
- si de nouvelles idées apparaissent et semblent intéressantes, éviter de dévier, mais conserver ces idées à part, de manière à y revenir plus tard, lors d'un autre atelier par exemple ;
- fixer un ordre du jour précis, avec une durée précise pour chaque point, et s'y tenir ;
- maintenir jusqu'au bout l'enthousiasme du début.

### L'interview structurée individuelle

---

#### Utilité et technique d'interview

Cette technique est utile, voire indispensable, pour connaître les *besoins individuels* des utilisateurs. Elle se focalise sur une partie du processus, en particulier celle dont l'utilisateur est expert ou praticien (mais pas

---

1. Dans *Requirements by collaboration*, Ellen Gottesdiener donne de nombreuses techniques et méthodes de recueil par ateliers de travail.

seulement). La difficulté est de trouver le bon échantillon d'utilisateurs, et de poser les questions adéquates en fonction des objectifs à atteindre. L'interview structurée permet généralement de recueillir ou de préciser des besoins dont les utilisateurs sont *conscients*. Ce point est à noter, car d'autres techniques, comme le brainstorming, l'observation directe, et dans une large mesure les ateliers de travail, permettent de faire émerger des besoins bien réels dont les utilisateurs n'ont pas conscience.

### Déroulement de l'interview

Les étapes d'une interview sont les suivantes :

1. **Préparation.** La préparation est indispensable à une interview efficace. Elle sera d'autant plus longue et importante que l'intervieweur sera novice dans cette technique et son expertise éloignée du domaine d'application.
  - Collecte et lecture de documents, relecture des interviews précédentes, de l'existant, du contexte.
  - Préparation des questions : questions génériques, questions générales de contexte, questions précises à l'interviewé, questions de complément d'autres interviews.
  - Planification de l'interview et ordonnancement des questions. Une interview dure environ une heure, maximum une heure et demie. Les questions doivent apparaître comme une suite logique, allant du général au particulier.
2. **Interview sur le terrain,** de préférence près du lieu de travail de l'interviewé (il sera plus à l'aise), mais pas directement sur son lieu de travail (il risque d'être interrompu).
  - Ouverture de l'interview : l'intervieweur se présente, rappelle l'objectif et du temps dont on dispose.
  - Questionnement : l'intervieweur pose la question de la grille et note la réponse. Il reformule si nécessaire et fait préciser les points qui ne lui paraissent pas clairs. L'opération est répétée jusqu'à ce que l'interviewé accepte la reformulation de l'intervieweur.
  - Clôture : l'interviewer rappelle les points principaux, relit les besoins tels que formulés par écrit et indique s'il y aura un compte-rendu et si ce compte-rendu devra être formellement validé.
3. **Finalisation.** L'intervieweur relit ses notes, structure les informations et rédige le compte-rendu. Il envoie le compte-rendu à l'interviewé pour validation. Ce dernier peut faire ses remarques. L'intervieweur peut compléter l'interview en posant des questions supplémentaires, souvent par téléphone.

### Poser des questions efficacement

L'interview contient en « modèle réduit » les quatre étapes du développement des exigences. Et dans le cadre d'une interview, chaque question posée est elle-même un microprocessus recueil-analyse-spécification-validation :

- **Recueil** : l'intervieweur pose une question et recueille de l'information.
- **Analyse** : en séance ou « à froid », l'intervieweur cherche à prioriser les réponses et cherche les incohérences. Il peut, en séance, dessiner des diagrammes pour s'assurer que lui et son interlocuteur ont compris la même chose.
- **Spécification** : en séance, l'intervieweur reformule la réponse ou pose des questions supplémentaires pour obtenir une exigence bien formulée le plus en amont possible. À froid, il rédige le compte-rendu d'interview avec précision.
- **Validation** : en séance, l'intervieweur cherche à associer à chaque exigence des critères d'évaluation. Par la suite, il demandera à l'interviewé de valider le compte-rendu.

Cette approche consistant à considérer chaque activité et chaque tâche comme un cycle contenant toutes les autres étapes est très efficace. L'idée est de découvrir le plus tôt possible les incohérences, incomplétudes et insuffisances des exigences, tant dans leur formulation que dans leur structure.

### Les types de questions

Une grille d'interview, préparée à l'avance, est utile. Elle contient des questions types à poser. Les questions ouvertes permettent de comprendre le contexte de travail, et les besoins immédiats. Ce sont des questions en « comment ». Par exemple, « comment faites-vous actuellement pour enregistrer une demande de prêt ? ».

Les questions en « quoi » permettent de dérouler un processus : « une fois que vous avez fini cette action, que faites-vous ? ». Et ainsi de suite. Cette manière de procéder permet de recueillir les exigences du même niveau appartenant à une séquence.

Les questions en « comment » permettent d'apporter plus de détails : « concrètement, comment faites-vous ? ». Ce type de question permet donc de descendre des exigences de haut niveau aux exigences opérationnelles.

Inversement, la question « pourquoi » est utile pour prendre de la hauteur (figure 8-2).



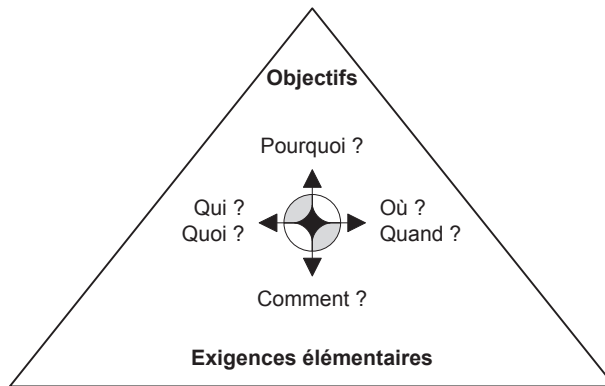


Figure 8-2 : Navigation entre niveaux d'exigences

Il est possible d'enchaîner les « pourquoi » pour remonter au niveau de besoin adéquat par rapport à l'objectif fixé. Voici une séquence d'interview en pourquoi :

« Il nous faut une case à cocher “membre du conseil” (expression d'une solution en termes d'interface utilisateur, beaucoup trop bas).

**Question**

Pourquoi une case à cocher ?

**Réponse**

On doit pouvoir indiquer au système qu'un de nos adhérents est membre du conseil d'orientation (il s'agit là d'une exigence fonctionnelle, correctement formulée, mais sans doute encore de trop bas niveau).

**Question**

Pourquoi doit-on pouvoir indiquer cela au système ?

**Réponse**

À tout moment, un utilisateur qui interroge le dossier d'un adhérent doit savoir s'il s'agit d'un membre du conseil d'orientation. »

En posant deux questions « pourquoi ? »<sup>2</sup>, on a ainsi trouvé une exigence globale, qui est un *invariant* du système à construire. Cela va sans dire, une exigence formulée de cette manière pose les bases d'un logiciel mieux construit et plus maintenable. L'exigence peut être spécifiée dans le cahier des charges :

À tout moment, au cours d'une consultation, le système doit donner à l'utilisateur la possibilité de savoir si un adhérent est membre du conseil d'administration.

2. Exercice : poser encore deux questions en pourquoi pour remonter jusqu'à un objectif.

En revanche, la question « pourquoi » posée à titre personnel est à proscrire, car elle peut être ressentie comme une agression. La première question de l'exemple précédent devient, en caricaturant « mais pourquoi voulez-vous à tout prix cliquer sur cette case ? ». Même sans aller jusque-là, une question « pourquoi » impliquant la personne interviewée doit être évitée.

### **Le secret des sept secondes**

Parfois, au détour d'une interview, on a la surprise d'entendre notre interlocuteur énoncer très clairement une exigence parfaitement bien formulée. L'analyste n'a plus rien à faire qu'à la noter verbatim et l'insérer, sans aucune reformulation, dans le cahier des charges. C'est là une surprise agréable qui a quelque chose de presque magique.

Que s'est-il passé ? Si notre interlocuteur formule un besoin avec autant de clarté et de précision, c'est très probablement qu'il y a longuement réfléchi avant, et l'a examiné sous toutes les coutures. En d'autres termes, notre interlocuteur a lui-même joué le rôle de l'analyste tout en étant un expert du métier avec les idées claires et un bon esprit de synthèse.

Existe-t-il une technique pour augmenter ses chances d'avoir des « spécifications spontanées » de la part d'un interlocuteur ? La réponse est oui, la technique est simplissime sur le principe, mais difficile à mettre en pratique. Il suffit d'écouter avec beaucoup d'attention, y compris, et surtout, les silences. Lorsque l'interviewé est en confiance, dans un environnement calme, qu'il n'est pas stressé par le temps, le silence est la meilleure réaction de la part de l'intervieweur.

Lorsque votre interlocuteur a fini de parler, laissez-lui encore sept secondes, au moins, avant de reprendre la parole. Sept secondes d'écoute du silence, c'est là le secret des meilleurs intervieweurs. C'est aussi le meilleur investissement que peut faire un analyste. Et surtout, un des moments les plus gratifiants de ce métier.

## **De l'ouverture à la précision**

Un des secrets de l'efficacité du processus de développement des exigences réside dans la capacité à faire passer les besoins par un « entonnoir », c'est-à-dire un tuyau très large à une extrémité et très étroit à l'autre. Cela se voit au niveau du macroprocessus : l'étape de recueil est ouverte, et autorise l'expression des besoins sous une forme large, peu précise. À l'autre bout du tuyau, la validation ne laisse passer que des exigences extrêmement précises. Entre les deux, les phases d'analyse, puis de validation, resserrent progressivement la précision.

Mais l'entonnoir peut aussi être appliqué au niveau du microprocessus. Une question peut être posée de manière très ouverte, puis resserrée par reformulations progressives. Et un des moyens pour y arriver consiste à filtrer les généralisations, les distorsions et les omissions.

### **Les généralisations**

On les détecte de deux façons.

- Par l'emploi de quantificateurs universels (tous, toujours, jamais, personne, chaque fois, etc.). Le travail de l'analyste consiste à reformuler en encourageant la précision. Par exemple :

**Interviewé (utilisateur)**

« Dans cet établissement, tout le personnel a accès à l'application par un identifiant et un mot de passe. »

**Analyste**

Tout le personnel ?

**Interviewé (utilisateur)**

Enfin, non, pas tous. Le personnel de l'entretien des locaux et les intérimaires n'y ont pas accès. »

- Une autre catégorie de généralisations se manifeste par les opérateurs modaux (il faut, on doit, etc.) Là aussi, c'est l'occasion de préciser les besoins, ou de détecter une faille dans le processus :

**Interviewé (utilisateur)**

« À son embauche, un employé doit s'inscrire auprès du service informatique pour obtenir un code d'accès. »

**Analyste**

Que se passe-t-il s'il ne le fait pas ?

**Interviewé (utilisateur)**

Eh bien, dans ce cas, il ne pourra pas accéder au système. »

La détection et le traitement des généralisations sont un bon moyen de trouver ce qui, dans un cas d'utilisation, s'appelle le *scénario alternatif*, ou les *exceptions*.

### **Les omissions**

Les omissions passent sous silence une partie de l'information. On trouve dans cette catégorie :

- Les omissions simples

**Interviewé (utilisateur)**

« Je traite le dossier. »

**Analyste**

Quel dossier ? (ou : en quoi consiste le traitement du dossier ?) »

- Les omissions par comparaison

**Interviewé (utilisateur)**

« Cette application est plus rapide. »

**Analyste**

Plus rapide que quoi ?

### **Interviewé (utilisateur)**

Plus rapide que l'application X du service Y. »

- Les omissions par manque d'index de référence qui omettent l'acteur

### **Interviewé (utilisateur)**

« On accède au système par un code d'accès temporaire.

### **Analyste**

Qui accède par un code temporaire ?

### **Interviewé (utilisateur)**

Les intérimaires et les sous-traitants. »

Le traitement des omissions permet de recueillir plus d'informations qu'une reformulation simple. Il améliore la précision et la complétude des informations recueillies sans attendre la phase d'analyse ou de spécifications.

## ***Les distorsions***

Enfin les distorsions :

- La divination

### **Interviewé (utilisateur)**

« Les secrétaires ne sont pas satisfaites de cette application.

### **Analyste**

Comment cela se manifeste-t-il ? »

La réponse à cette question va peut-être permettre de comprendre si les secrétaires sont véritablement insatisfaites de l'application ou si le problème est ailleurs.

- Le lien de cause à effet

### **Interviewé (utilisateur)**

« Le logiciel n'est pas utilisé, il manque d'ergonomie.

### **Analyste**

Est-ce vraiment le manque d'ergonomie qui freine l'adoption du logiciel ? »

Une manière moins abrupte de reformuler la question est de rebondir :  
« En dehors de l'ergonomie, que manque-t-il à ce logiciel ? »

Détecter les distorsions permet généralement d'augmenter la précision de la formulation du besoin.

## ***Attention aux effets inattendus***

Demander un peu plus de précision de la part de son interlocuteur est un bon moyen d'anticiper, lorsque cela est possible, sur les phases d'analyse

et de spécification<sup>3</sup>. Plus on anticipe au moment de l'interview, plus on capte d'informations précises, et plus on élimine rapidement les ambiguïtés, imprécisions, et incomplétudes. La recherche de la précision est un investissement rentable, car une accumulation d'imprécisions et d'incohérences coûte beaucoup plus cher à détecter et à éliminer après coup.

Cependant, une interview n'est pas un interrogatoire de police. Il faut faire attention à ne pas « forcer » l'interlocuteur. Il a le droit d'être imprécis, et même incohérent. Et surtout, il a le droit d'avoir son jargon. C'est à l'analyste de s'adapter. Il faut donc être prudent, surtout au début de l'élaboration du cahier des charges, surtout au début d'une interview. En tout état de cause, il est inutile de rechercher la précision si cela va jusqu'à énerver notre interlocuteur. Mieux vaut patienter, rechercher les informations à une autre source, que de risquer de bloquer le processus.

---

3. Lors de la phase de spécification et de validation, les imprécisions et incohérences seront détectées au moyen de check-lists.

---

## Le brainstorming

---

Technique créative, le brainstorming est surtout utile pour le développement d'un nouveau produit, lorsqu'il s'agit de faire jaillir de nouvelles idées.

Une séance brainstorming bien menée permet de recueillir des exigences dont les utilisateurs ne sont pas conscients.

Cette technique permet aussi de s'attaquer<sup>4</sup> à un problème qui sort des catégories habituelles. C'est le cas d'un projet qui fait l'objet de très nombreuses contraintes. Dans un tel cas, une nouvelle idée originale permet de résoudre ces contraintes en sortant du cadre habituel de réflexion.

Le brainstorming peut également s'avérer utile lorsque maîtrise d'ouvrage et maîtrise d'œuvre sont en conflit (larvé ou ouvert) et que les différents intervenants de la maîtrise d'ouvrage n'ont pas une idée très claire de ce qu'ils souhaitent. Recueillir les différentes fonctions ou contraintes sur des Post-It® et les étaler sur un mur ou un tableau peut permettre de clarifier la situation.

Une telle séance de brainstorming peut être suivie d'un diagramme des affinités ou d'une maquette papier.

---

**4. Brainstorming.**  
Terme inventé par Alex Osborne de brain, (cerveau), et to storm, (attaquer, assaillir). Il ne s'agit donc pas d'une « tempête dans un cerveau », mais de s'attaquer mentalement, collectivement, à une difficulté.

---

## Le diagramme des affinités

---

La méthode du diagramme des affinités est utile lorsqu'il s'agit de « faire le tour de la question », d'en voir les différents aspects, et de les classer par catégories. Les étapes de la méthode sont les suivantes :

- L'animateur expose le problème aussi clairement que possible.

- Chaque participant écrit une ou plusieurs réponses au problème, chacune d'elle sur une fiche (par exemple, une fiche adhésive de type Post-It®).
- L'animateur ramasse les réponses et les expose à l'ensemble des participants. Il étale les fiches sur une table ou les colle sur un mur.
- Les participants regroupent les fiches par catégories, sans avoir à se justifier ; un participant peut déplacer une fiche d'une catégorie à une autre.
- Le regroupement en catégories continue, jusqu'à obtention d'un consensus.
- On peut créer des sous-catégories.
- Les participants donnent un nom à chaque catégorie et sous-catégorie.

Plusieurs variantes à cette technique existent, en fonction du type de problème à traiter. Elle est surtout utile lors des premières étapes de recueil et d'analyse des besoins, par exemple, pour connaître les parties prenantes ou pour déterminer les grandes fonctions attendues du système (figure 8-3).

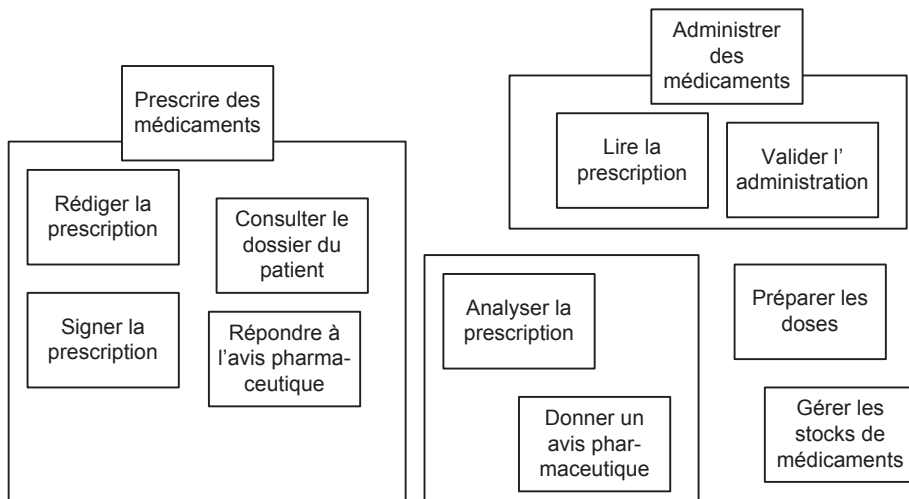


Figure 8-3 : Diagramme des affinités

## L'observation directe

Observer un utilisateur sur le lieu de son travail est souvent le moyen le plus simple de connaître ses besoins. C'est aussi un moyen de connaître ses difficultés et les problèmes qu'il rencontre (avec le logiciel actuel ou en l'absence de logiciel) de manière à formuler des exigences qui aboutiront à un produit bien adapté à leurs besoins.

Certains utilisateurs (ou futurs utilisateurs) sont capables de modéliser eux-mêmes leur travail, c'est-à-dire décrire la succession des tâches qui le constituent. D'autres sont beaucoup plus à l'aise lorsqu'ils sont dans l'action. L'observation peut être neutre, à distance, l'observateur se faisant le plus discret possible. Elle peut être plus intrusive, l'observateur posant à l'utilisateur des questions sur la manière dont il s'y prend.

Cette technique a ses limites. Elle permet de recueillir les besoins opérationnels pour des opérations visibles, les opérations intellectuelles étant difficilement observables et analysables de l'extérieur. Avec cette technique, on ne pourra décrire les besoins génériques ou de haut niveau qu'après analyse de nombreuses observations.

## Les questionnaires

---

Le questionnaire est un moyen facile de recueillir des opinions ou des données de la part d'un grand nombre de personnes. Cependant, cette technique n'encourage pas la créativité et l'émergence de nouvelles idées.

La technique la plus courante consiste à utiliser une échelle de *Likert*. Les catégories de réponses, sur une échelle comportant quatre à sept niveaux, sont formulées de manière à ce que leur signification soit claire pour tout le groupe de personnes interrogées.

Les questions fermées (cases à cocher) facilitent l'analyse. Mais l'élaboration de telles questions est loin d'être un exercice facile. Les questions doivent être très claires, spécifiques, neutres (ne pas induire la réponse) et formulées avec le vocabulaire de celui qui répond. Il est possible d'ajouter quelques questions ouvertes lorsque le questionnaire s'adresse à un petit nombre de personnes.

Pour élaborer le questionnaire, les questions à se poser sont :

- Quel est l'objectif du questionnaire ? Que veut-on savoir précisément ?
- Comment cette information peut-elle être fournie ?
- Quel est le degré de précision recherché ?
- Quel est le nombre de questions à poser pour avoir les informations recherchées ?
- Combien de questions est-il réaliste de poser ?
- Qui va exploiter les réponses et comment ?

## La réutilisation d'exigences

---

Un cahier des charges d'un autre logiciel, ou d'une version précédente du même logiciel, peut contenir des exigences réutilisables. Souvent, le recueil consiste alors à simplement reprendre les exigences, éventuellement en mettant à jour le vocabulaire. On vérifiera sa compatibilité avec les autres exigences, pour éviter redondances et incohérences.

## L'analyse de produits existants

---

À défaut de cahier des charges existant, on peut examiner un logiciel déjà en production. Il sera très riche en enseignements.

Il est bien entendu possible d'examiner un produit déjà en production, qui arrive en fin de vie, que l'on souhaite remplacer ou redévelopper. L'analyse d'un produit existant permet dans ce cas de découvrir des exigences implicites, des fonctions utilisées au quotidien, mais dont les utilisateurs n'ont pas conscience.

On peut également analyser un produit concurrent de celui que l'on souhaite développer ou faire développer. Si la loi interdit de « décompiler » un logiciel, rien n'interdit en revanche de l'analyser sur le plan comportemental. À ce propos, un éditeur d'un logiciel pourtant « pointu » à qui on disait que son produit était le meilleur du marché a répondu qu'il avait « acheté, installé et testé tous les produits de la concurrence ». Cela ne l'empêchait pas de proposer des fonctions innovantes introuvables ailleurs.

## Rechercher l'information là où elle se trouve

---

Nous avons vu qu'il y a différents niveaux d'exigences. À chaque niveau, les exigences seront recueillies auprès d'un interlocuteur différent, avec des techniques différentes.

Le donneur d'ordres va apporter les exigences du plus haut niveau, c'est-à-dire les objectifs. On utilisera l'interview structurée, ou les réunions en petit comité. Les techniques comme le brainstorming ou le diagramme des affinités peuvent être très utiles, mais rarement possibles dans la pratique. Les objectifs doivent être recueillis le plus souvent chez des dirigeants ou les cadres supérieurs, et il est difficile de réunir tous ces décideurs pendant une heure trente sur le thème « brainstorming ». L'interview face à face est la technique la plus indiquée dans ce cas.



Pour recueillir les besoins des utilisateurs, les groupes de travail d'une journée à intervalles de deux ou trois semaines, programmés longtemps à l'avance, sont les plus efficaces.

L'observation directe, sur site, est consommatrice de temps pour l'analyste. Si celui-ci est habile et discret, un grand nombre d'observations peuvent être faites avec un minimum de perturbation des utilisateurs au travail. L'observation directe alternée avec les groupes de travail est très utile, l'une permettant de valider l'autre.

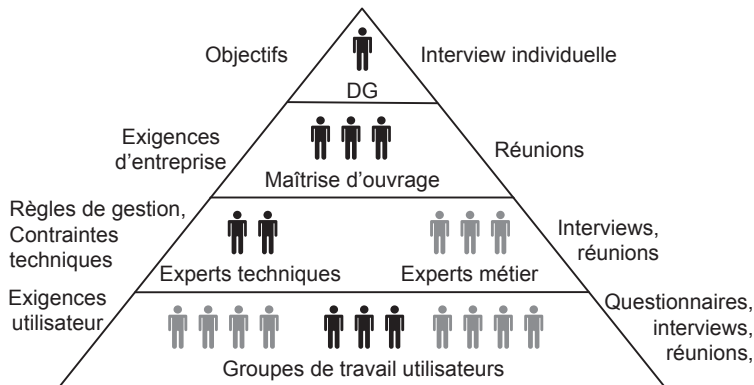


Figure 8-4 : Recueillir l'exigence au bon niveau

## Les bonnes pratiques

Que ce soit au cours d'une interview ou lors d'une réunion de travail, un certain nombre de bonnes pratiques doit être respecté si l'on veut travailler efficacement.

### Recueillir les besoins au bon niveau

Comme nous l'avons vu, les exigences devront être recueillies auprès des interlocuteurs idoines. On recueille les objectifs (exigences du niveau le plus élevé) auprès de la direction générale, les exigences les plus détaillées auprès des utilisateurs sur le terrain.

Ce principe général pose la question de l'ordre dans lequel seront menées les interviews. A priori, l'ordre idéal est celui qui consiste à partir des objectifs (donc du maître d'ouvrage stratégique), puis à descendre progressivement vers plus de détails jusqu'aux utilisateurs les plus opérationnels. Outre le fait que cela n'est pas toujours possible, cela pose d'autres difficultés : si l'on veut éviter de poser au directeur général des questions dont la réponse est facile à trouver auprès des opérationnels

(mais a priori inconnue de l'intervieweur), il faudrait les avoir interviewés avant.

### Apporter un retour rapide aux interlocuteurs

Suite à un entretien ou à une réunion de recueil, il est utile de rédiger un compte-rendu. C'est une bonne pratique bien connue, et également une marque de courtoisie à l'égard de ses interlocuteurs. Rédiger un compte-rendu détaillé et précis présente aussi quelques inconvénients car cela prend du temps qui pourrait souvent mieux être utilisé ; on se retrouve avec un grand nombre de comptes-rendus, qui constituent une documentation lourde à gérer.

Une manière plus efficace de procéder consiste à :

- lors d'une interview ou réunion, donner aux interlocuteurs un feedback immédiatement après l'expression d'un besoin ;
- suite à une interview ou une réunion, utiliser le cahier des charges lui-même comme compte-rendu.

Très concrètement, au cours du recueil, il est utile de donner un retour direct (un feedback) à ses interlocuteurs, sous forme de reformulation de ce qui vient d'être dit, éventuellement en utilisant un moyen de communication différent, par exemple un schéma. Cela rassure les interlocuteurs et évite de partir dans une fausse direction.

Puis, suite à l'entrevue ou la réunion, on intégrera ce qui a été formulé dans le cahier des charges, et on le soumettra à la validation<sup>5</sup> des participants. Attention : il ne s'agit en aucun cas de court-circuiter les étapes d'analyse et de spécification, mais de faire une analyse partielle et une spécification des besoins qui ont été formulés lors de la réunion.

---

5. Une astuce : utiliser le mode révision d'un traitement de texte (si on maîtrise cette technique), ou utiliser des paragraphes de couleurs différentes pour visualiser les modifications nouvellement intégrées.

---

### Parler le langage du client

De manière générale, si on veut faciliter la communication, il faut apprendre et utiliser le langage du maître d'ouvrage plutôt que de forcer ce dernier à utiliser notre jargon technique.

Par langage, on entend bien sûr le vocabulaire, mais également la manière de représenter l'information. Un juriste a l'habitude de travailler sur du texte. Si un schéma simple peut éventuellement l'aider, un schéma complexe (ceux dont raffolent justement les informaticiens) est à éviter. Inversement, si les utilisateurs sont des informaticiens ou des ingénieurs, un schéma sera souvent plus parlant qu'un texte.

### Lever toute ambiguïté au fil de l'eau

Les mots ont souvent un sens différent d'un métier à l'autre, même au sein d'une même organisation. Rappelons à ce sujet que tout terme un

tant soit peu ambigu devra être inclus dans un glossaire. Ce glossaire, qui fait partie du cahier des charges, devra être enrichi tout au long du processus de son élaboration.

### Maintenir la cohérence le plus en amont possible

Les étapes d'analyse, de spécification et de validation sont présentées dans les chapitres suivants de cet ouvrage. Mais certaines des opérations de ces étapes peuvent être effectuées en amont, en phase de recueil :

- suite à une interview, réaliser un examen rapide de la cohérence avec les notes d'autres interviews, afin de détecter des incohérences ou des manques ;
- lors d'une réunion de recueil, représenter l'information sous forme de schéma ;
- à la suite d'une interview, et si nécessaire, envoyer un bref compte-rendu à la personne interviewée pour validation.

En cas de conflit entre exigences, il est important qu'il soit résolu avant de poursuivre. L'analyse des parties prenantes qui a été faite pendant l'étape préparatoire permet de savoir qui a plus de poids. Trouver le décideur qui tranchera le conflit est important, mais cela demande de l'habileté. Le décideur n'est pas toujours le futur utilisateur, il n'est pas au-dessus des lois et des règlements, mais il joue un rôle clé.

Cette activité relève de la *négociation des exigences* (que certains auteurs considèrent comme une étape à part entière dans le processus de développement et de gestion des exigences). Si les conflits entre exigences ne sont pas réglés pendant la phase d'exigences (et de préférence pendant le recueil, ou juste après), qui se chargera de le faire ? Les équipes de la phase suivante, c'est-à-dire les concepteurs-développeurs. Ce n'est pas leur rôle.

### Recueillir les besoins et non les solutions

Dès l'étape de recueil, il est important de ne retenir que les besoins, et non les solutions. Même chez les assistants à maîtrise d'ouvrage expérimentés, exprimer des solutions en lieu et place des besoins est une erreur fréquente. En effet, il existe une « zone floue » entre les exigences et la conception, et il n'est jamais facile de savoir quand s'arrêter.

### Recueillir les besoins alternatifs et exceptionnels

On a naturellement tendance à recueillir le « flux normal » d'un processus, et à exprimer les besoins correspondant à des conditions normales. Or, un système doit traiter les situations d'exception, par définition rares et improbables. Très souvent, pour un seul cas normal d'utilisation du système, il peut exister de nombreux cas exceptionnels.

### Alternatives et exceptions

Les cas exceptionnels sont, par définition, des exigences qui se présentent rarement. Mais leur recueil peut être difficile et leur spécification longue. Ce sont eux qui posent problème, à toutes les phases du cycle de vie, depuis les exigences jusqu'à l'exploitation. Raison de plus pour les recueillir, les analyser et les spécifier avec soin.

Penser à tous les cas exceptionnels dans le cas d'un retrait à un distributeur automatique de billets, par exemple en cas d'erreur ou de malveillance. Penser aussi aux anomalies à traiter suite à une défaillance du système lui-même.

En plus des cas exceptionnels (par exemple, dépassement d'une limite autorisée ou erreur de la part de l'utilisateur), il faut penser au comportement que le système devra adopter en cas d'anomalie du système lui-même.

### Recueillir les besoins positifs, mais aussi négatifs

Les besoins négatifs, ou exprimés négativement, sont fort utiles à l'analyste. Le « questionnaire négatif » permet de mettre en lumière des besoins qui n'auraient pu être exprimés que difficilement de manière positive. C'est aussi un bon moyen de recueillir les besoins des personnes résistantes au changement. La négation est souvent la porte de l'amélioration, et les personnes résistantes au changement peuvent être très créatives.

#### Recueil des besoins négatifs

Comme pour les besoins positifs, les besoins négatifs peuvent être recueillis par des questions ouvertes, semi-ouvertes, ou fermées : Qu'est-ce qui vous dérange le plus dans le système actuel ? Si quelque chose vous dérangeait avec le système actuel, ce serait quoi ? Que voudriez-vous que le système à venir fasse, que le système actuel ne fait pas ? Faisons la liste de toutes les fonctions que le système actuel accomplit mal.

### Écouter le silence

Il y a des besoins que l'on recueille plus facilement en creux, dans le non-dit ou dans le « presque rien ». En posant des questions ouvertes, générales, voire imprécises, vagues, ambiguës... la réponse vient souvent spontanément. Le client complète la question imprécise par une phrase précise, en employant ses propres mots.

L'imprécision déclenche la précision. Une question imprécise est souvent un moyen d'aboutir à une formulation précise.

Le langage flou permet de formuler un discours où chacun entend ce qu'il veut bien. Proscrit de l'étape de spécifications (c'est de la langue de bois), il est ici utilisé pour provoquer de la précision de la part de l'interviewé.

### **Questions vagues qui apportent de la précision**

Quelle fiabilité attendez-vous du futur système ?

Quel niveau de richesse fonctionnelle exigez-vous ?

En ce qui concerne la fonction X.... (laisser un silence) ?

### **Tester la validité des exigences au plus tôt**

Il ne s'agit pas de valider les exigences, mais de les tester au plus tôt. Sont-elles utiles ? Reflètent-elles les vrais besoins ? Sont-elles réalisables à un coût raisonnable ?

Il s'agit de parcourir, avec l'interviewé, le cheminement mental qui a conduit à l'expression de ce besoin. Parfois, les besoins exprimés n'ont aucune utilité. C'est en particulier le cas lorsqu'un utilisateur a pris ses habitudes avec un système ancien et veut retrouver les mêmes fonctionnalités dans le nouveau système, alors qu'elles n'ont pas lieu d'être.

Un autre moyen consiste, après avoir interviewé un utilisateur et lui avoir fait préciser un besoin, de l'exprimer à une autre personne et de lui demander son avis. Les interviews individuelles sont parfois plus efficaces que les groupes de travail, surtout si un besoin « obsolète » a été exprimé par un supérieur hiérarchique : le subordonné ne s'exprimera que s'il a confiance en l'intervieweur.

### **Check-list de fin d'étape de recueil**

---

La « fin d'étape » est hypothétique, car les besoins évoluent en permanence, qu'on ne peut jamais être certain d'avoir épuisé la liste potentielle d'exigences, et que les étapes sont imbriquées : on aura commencé à analyser, spécifier et valider bien avant d'avoir coché tous les points de la check-list.

Cette check-list peut être utilisée à tout moment pour connaître l'avancement du recueil des exigences et éviter de passer à côté de points importants.

- On a soigneusement établi la liste des parties prenantes.
- On a formalisé les attentes de chaque partie prenante.
- Toutes les parties prenantes sont représentées et actives.
- On a soigneusement établi la liste des sources d'exigences.
- Les exigences ont été recueillies chez les vrais utilisateurs.
- Toutes les catégories d'utilisateurs sont représentées.
- Toutes les catégories d'utilisateur se sont exprimées.

- On a recueilli les exigences positives, et aussi négatives.
- On a donné un feedback à tous les participants.
- Les participants ont relu ou revu les besoins exprimés.
- Les exigences dérivent des objectifs.
- On n'a retenu que les besoins, non des éléments de solution.

## Chapitre 9

# Les cas d'utilisation

Les cas d'utilisation (*use cases* en anglais) sont devenus célèbres grâce, en particulier, à UML (*Unified Modeling Language*). Cependant, les adeptes d'UML utilisent le plus souvent les diagrammes de cas d'utilisation, ce qui est un peu réducteur. Les cas d'utilisation sous forme textuelle offrent, eux, une grande richesse d'expression.

### Qu'est-ce qu'un cas d'utilisation ?

---

Un cas d'utilisation est un contrat de comportement, entre un *système* (par exemple, un logiciel) ou un sous-système et des *acteurs* qui interagissent avec lui. Ces acteurs peuvent être des utilisateurs (humains) du système ou des systèmes voisins.

Un cas d'utilisation regroupe un ensemble de *scénarios* (figure 9-1) qui peuvent, soit aboutir, soit échouer. Pour formuler un cas d'utilisation, on doit respecter les contraintes suivantes :

- Un cas d'utilisation se rapporte à un objectif et un seul.
- Il décrit un comportement utile à l'atteinte de cet objectif.
- La description prend le point de vue d'une (ou plusieurs) partie(s) prenante(s).
- Il comporte un acteur principal et un seul.
- Il débute avec un événement déclencheur.
- Il se poursuit jusqu'à ce que l'objectif soit atteint ou abandonné.

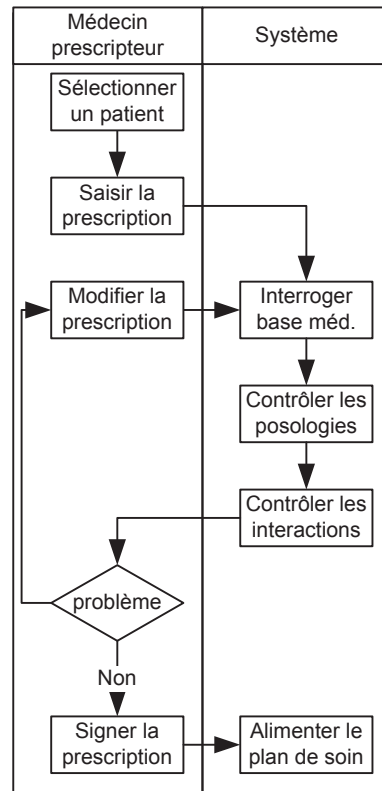


Figure 9-1 : Scénario de cas d'utilisation

## Le contenu d'un cas d'utilisation

Un cas d'utilisation s'exprime essentiellement sous forme textuelle. Les différents items qui le composent varient selon les auteurs<sup>1</sup>, le type d'application, le degré de précision que l'on veut atteindre grâce aux spécifications. Voici une structuration assez classique :

- **Acteur principal** : c'est la personne qui, dans le présent cas d'utilisation, interagit avec le système. À eux deux, ils « exécutent » le cas d'utilisation.
- **Autres acteurs** : toutes les personnes impactées par la mise en œuvre du cas d'utilisation.
- **Déclencheur** : il s'agit de l'événement qui déclenche le cas d'utilisation. En principe, c'est un événement métier (voir le diagramme de contexte).
- **Description** : description courte du déroulement du cas d'utilisation.

1. Le plus connu est Alistair Cockburn. Voir *Rédiger des cas d'utilisation efficaces* publié aux éditions Eyrolles.



- **Préconditions** : conditions qui doivent être vérifiées, pour que le cas d'utilisation démarre.
- **Postconditions** : également appelées *garanties en cas de succès*. État du système à la fin de l'exécution du cas d'utilisation.
- **Garanties minimales** : état du système lors de l'exécution du cas, quelle qu'en soit l'issue.
- **Scénario nominal** : c'est un scénario formulé comme une alternance d'actions de l'utilisateur et de réponses du système. Il décrit l'exécution du cas d'utilisation dans des conditions attendues. L'exécution du scénario permettra d'atteindre l'objectif énoncé dans le titre du cas d'utilisation. Généralement, les actions sont numérotées.
- **Scénario alternatif**<sup>2</sup> : séquence alternative du scénario. La numérotation permet de se raccrocher au flux normal. Par exemple, si on décrit une alternative à la séquence 2-5, on notera les actions 2-5.a, 2-5.b, etc.
- **Exceptions** : séquence déclenchée par une anomalie ou une erreur lors de l'exécution du flux normal ou d'un flux alternatif.
- **Cas inclus** : liste des cas d'utilisation inclus dans le cas d'utilisation présent. Les cas inclus sont en quelque sorte des « sous-cas » (analogues à des sous-programmes). Ils permettent de décrire des séquences qui se retrouvent dans plusieurs cas d'utilisation, ou d'améliorer la lisibilité.
- **Fréquence d'utilisation** : nombre estimé de fois où ce cas d'utilisation sera exécuté par les acteurs, dans l'unité de temps appropriée (jours, heures...).
- **Règles métier** : règles métier auxquelles ce cas est subordonné.
- **Exigences particulières** : exigences supplémentaires (exigences non fonctionnelles, contraintes...) qui s'appliquent à ce cas d'utilisation.
- **Notes et questions** : remarques particulières.

---

2. Cockburn utilise le terme *extensions* qui regroupe scénarios alternatifs et exceptions.

---

Cockburn et ses adeptes ajoutent les points suivants :

- **Portée de conception** : c'est la largeur du champ de la description ; par exemple l'entreprise, le système, ou un sous-système.
- **Niveau d'objectif** : désigne le niveau de l'objectif que l'on veut atteindre : niveau stratégique, niveau utilisateur (intéresse l'acteur principal), ou niveau d'une sous-fonction.

Ces deux derniers points sont intéressants, et pour Cockburn, ils devraient être obligatoires pour décrire un cas d'utilisation. C'est ainsi dans les projets importants où l'on développe un grand nombre de cas d'utilisation. Ces notions, bien que très utiles, ne sont pas toujours faciles à manier.

## Avantages des cas d'utilisation

---

Les cas d'utilisation ont un énorme avantage : ils concilient les littéraires (au sens large du terme : c'est-à-dire les personnes portées sur le langage écrit plutôt que sur les schémas) et les informaticiens. Pour qui a pratiqué la programmation, un cas d'utilisation rappelle un algorithme écrit en pseudo-code. Il présente néanmoins les exigences sous une forme aisément compréhensible par tous, ce qui permet d'obtenir facilement une vision commune, et donc d'aboutir à un consensus entre parties prenantes (c'est le but du cahier des charges).

L'autre avantage est celui du point de vue. Les cas d'utilisation sont écrits dans la langue de l'utilisateur, et centrés sur ses actions. Cela augmente les chances d'obtenir un produit lui-même centré sur l'utilisateur.

Enfin, les exigences rédigées de cette manière modélisent une activité réelle, et donc de vrais besoins ; avec ce procédé, il y a peu de chances qu'un utilisateur exprime des besoins « gadgets » pour se faire plaisir (ce qui arrive souvent quand on commence par décrire l'interface utilisateur).

Les cas d'utilisation offrent plus de chances à l'équipe de développement de réaliser un logiciel robuste. Ils évitent les zones floues qui pourraient être mal interprétées pendant le développement. En décrivant tous les scénarios possibles (combinaisons du scénario nominal, des scénarios alternatifs et des scénarios en cas d'échec), on a plus de chances de donner aux concepteurs un document cohérent qu'en leur fournissant une liste de fonctions. Enfin, à partir des cas d'utilisation, il est relativement facile de déduire les cas de test.

Les cas d'utilisation sont donc le « couteau suisse » de la modélisation des exigences. Ils peuvent être utilisés à tous les niveaux, depuis les objectifs stratégiques jusqu'à la description d'un comportement très détaillé. Ils permettent de décrire des processus, de négocier entre parties prenantes, de faire ressortir (et donc d'anticiper) les problèmes. À partir de là, il sera possible de faire une première estimation des coûts de développement, de reboucler sur les objectifs définis plus en amont, puis de passer à la phase de conception. Les concepteurs pourront par la suite se servir de cas d'utilisation pour décrire, non plus les besoins, mais la solution, et la maîtrise d'ouvrage reprendre la main pour préciser les scénarios de recette.

## Élaboration des cas d'utilisation

---

Les cas d'utilisation sont un excellent outil d'analyse, mais aussi de recueil et de spécification. Ils s'avèrent utiles pour assembler les besoins

recueillis auprès de plusieurs acteurs, travailler avec les différents acteurs, interagir avec eux, et faire valider les exigences correspondantes. En d'autres termes, c'est un bon moyen de transformer un besoin en exigence.

Une séquence d'élaboration d'un cas d'utilisation peut être la suivante :

- recueillir les besoins de haut niveau auprès de la maîtrise d'ouvrage ;
- en déduire les cas d'utilisation ;
- recueillir les besoins auprès de différentes sources (documentation client, produits existants, etc.) ;
- compléter le recueil auprès de plusieurs utilisateurs ;
- consolider l'ensemble et dégager un ou plusieurs scénarios ;
- décrire ces scénarios au moyen d'un cas d'utilisation relativement simple ;
- distribuer le cas d'utilisation aux participants d'un groupe de travail ;
- travailler en groupe sur le cas d'utilisation pour l'affiner ;
- si nécessaire, itérer sur les étapes précédentes.

Un autre moyen efficace de déterminer les cas d'utilisation consiste à partir du diagramme de contexte :

- Chaque flèche (entrante ou sortante) du diagramme de contexte est un événement métier (*business event*).
- L'événement métier est formulé sous forme de phrase à l'infinitif (par exemple, *prescrire un médicament*).
- L'événement métier devient la description du cas d'utilisation.
- Une extrémité de la flèche pointe vers le système ; l'autre extrémité est l'acteur principal (par exemple, *médecin prescripteur*).
- On travaille avec les parties prenantes, en particulier un représentant des utilisateurs (correspondant à l'acteur principal et aux acteurs secondaires, par exemple un médecin et un pharmacien) pour décrire les scénarios (nominal et alternatif).

Dans la pratique, comme toujours dans la définition des besoins, on peut combiner ces deux approches, par raffinements successifs et allers et retours entre un diagramme de contexte et des cas d'utilisation.

### **Capter et spécifier les exigences à partir des cas d'utilisation**

Les cas d'utilisation constituent un excellent moyen de recueil et de spécification d'exigences plus détaillées (exigences élémentaires, en anglais *atomic requirements*). La méthode consiste à se poser les questions suivantes pour chaque étape du scénario :

- Quelles sont les données en entrée, en sortie, ou stockées lors de cette étape ?
- Quels sont les traitements (calculs, vérifications) faits par le système à cette étape ?

## Un exemple

---

Voici un exemple extrait d'un cahier des charges du domaine des systèmes d'information hospitaliers. L'acteur principal est le prescripteur (médecin ou sage-femme). Les acteurs secondaires, pharmacien et infirmier, traitent la prescription.

Nom du C.U.	Prescrire
Acteur principal	Prescripteur
Autres parties prenantes	Pharmacien, infirmier
Description	Le prescripteur prescrit des médicaments à un patient hospitalisé.
Préconditions	Le prescripteur est identifié. Le système dispose des informations utiles sur le patient. Le patient est admis dans l'unité. Le prescripteur a effectué son observation médicale. Le prescripteur a consulté le dossier patient. Le livret thérapeutique est accessible au prescripteur.
Garanties minimales	Le système tient à jour un journal de toutes les opérations. Les informations saisies par le prescripteur ne seront accessibles aux autres acteurs qu'une fois la prescription signée. La prescription est une opération atomique. Elle ne peut réussir entièrement ou échouer.
Garanties en cas de succès	La prescription peut être validée par la pharmacie. Mise à jour du plan de soins avec le plan d'administration prévisionnel. Intégration de l'ordonnance dans le dossier médical du patient.
Flux normal	1. Le prescripteur sélectionne dans la liste des patients de l'unité de responsabilité médicale le patient pour lequel une prescription doit être faite. 2. Le prescripteur saisit un élément de prescription autant de fois que nécessaire. [saisir un élément de prescription est un cas inclus]. 3. Le système contrôle les posologies et les interactions médicamenteuses en interrogeant la base médicamenteuse. 4. Le système contrôle les interactions physiopathologiques. 5. Le système signale les éventuelles interactions et anomalies.

Nom du C.U.	Prescrire
Flux normal (suite)	<p>6. En cas d'anomalie, le prescripteur peut modifier la prescription ou la maintenir et en indiquer les raisons.</p> <p>7. Le système inscrit la prescription dans le plan de soins de l'infirmière avec un statut « en attente d'analyse par la pharmacie ».</p> <p>8. Le prescripteur signe la prescription.</p> <p>9. Le prescripteur peut imprimer l'ordonnance.</p>
Flux alternatif	<p>[On traite ici d'une alternative à l'étape 1] :</p> <p>1a - Si le prescripteur décide une hospitalisation immédiate en consultation :</p> <p>1a1 - La prescription est faite sur une unité de consultation.</p> <p>1a2 - Le prescripteur indique l'unité d'hospitalisation.</p> <p>1a3 - Le système affecte la prescription à l'unité d'hospitalisation.</p> <p>[Autre alternative à l'étape 1] :</p> <p>1b - Si la même équipe médicale et soignante travaille sur deux unités :</p> <p>1b1 - Le prescripteur consulte une liste de patients comportant les patients des deux unités.</p>
Cas inclus	Saisir un élément de prescription
Fréquence d'utilisation	À chaque consultation

Les cas d'utilisation peuvent être plus ou moins détaillés, selon le niveau de précision du cahier des charges.

## Difficultés et risques des cas d'utilisation

Le premier risque est lié à l'apparente facilité d'écriture. On peut faire l'analogie avec un ouvrage littéraire : un cas d'utilisation bien écrit est très facile à lire par tous les intervenants d'un projet. Son écriture est en revanche beaucoup plus difficile, et demande de savoir maîtriser les trois concepts de *portée*, d'*acteur principal* et de *niveau*, ce qui demande un certain entraînement.

Les cas d'utilisation sont également difficiles à comprendre (et a fortiori à élaborer) lorsqu'ils décrivent un processus qui n'existe pas et qui devra être informatisé. Par exemple, un médecin comprendra très facilement le

cas d'utilisation appelé *prescrire un médicament*, mais aura du mal à réagir face à un cas d'utilisation décrivant une pratique professionnelle qu'il ne maîtrise pas encore.

D'autre part, les cas d'utilisation ne sont pas adaptés à toutes les situations. Ils sont de peu d'utilité pour des systèmes où les interactions et transactions sont peu fréquentes, comme les programmes de calcul scientifique ; et également dans des cas où les interactions sont très complexes, comme les systèmes temps réel. Il en est de même pour les systèmes s'appuyant sur des règles de gestion nombreuses et complexes. Enfin, ils ne sont pas du tout adaptés à la description des exigences non fonctionnelles. Dans tous ces cas, il est préférable d'utiliser d'autres modèles de description.

Vouloir spécifier la totalité des exigences sous forme de cas d'utilisation est donc inefficace. Et certaines exigences, même parmi celles pouvant être décrites de cette façon, gagent à l'être au moyen de formalismes différents.

D'une manière générale, lors de l'élaboration d'un cas d'utilisation, il faut rester simple et utiliser cet outil pour ce à quoi il est destiné : décrire un ensemble de scénarios permettant d'atteindre un objectif dans l'intérêt des acteurs. Les cas suivants doivent être considérés comme des signaux de danger :

- des cas d'utilisation trop nombreux, dans lesquels on se perd ;
- des cascades de cas inclus ;
- des cas d'utilisation complexes, trop longs, incompréhensibles par certains acteurs ;
- des cas d'utilisation qui décrivent l'interface utilisateur, ou les données : ils ne sont pas faits pour ça.

## Les diagrammes de cas d'utilisation

---

Que penser des diagrammes de cas d'utilisation ? Les adeptes d'UML en font un usage abondant, alors qu'Alistair Cockburn ne les utilise qu'avec grande parcimonie, en mettant le lecteur en garde contre les risques que présente leur utilisation.

En réalité, dans les cas simples, les diagrammes de cas d'utilisation sont faciles à dessiner et à comprendre, mais sont de peu d'utilité. Dans les cas complexes, ils sont peu maniables et ont tendance à devenir incohérents et ambigus. Finalement, ils ne sont utiles qu'à petites doses, à titre illustratif.

### **Verbatim**

« Si vous passez beaucoup de temps à étudier les graphiques et les relations, et à les prendre en compte, c'est que votre énergie est mal utilisée. Placez-la plutôt dans la rédaction de textes facilement lisibles. En prose, les relations entre cas d'utilisation sont directes. Après cela, vous aurez du mal à comprendre pourquoi certains se font des nœuds au cerveau à leur sujet.

[...] Ceux qui écrivent de bons cas d'utilisation textuels ne sont tout simplement pas confrontés aux problèmes que rencontrent ceux qui bricolent avec des personnages stylisés, des ellipses et des flèches préconisés par UML. Les relations se dégagent naturellement de l'écriture du récit ; elles ne deviennent problématiques que lorsqu'on en fait un abcès de fixation. Cette idée fait son chemin à mesure qu'un nombre croissant de consultants accumulent une expérience double des cas d'utilisation textuels et d'UML ».

Alistair Cockburn, *Écrire des cas d'utilisation efficaces*, éditions Eyrolles, 2001.





# L'étape d'analyse

Le maître d'ouvrage, les utilisateurs et autres parties prenantes expriment généralement leurs besoins en langue naturelle. Dans le cahier des charges, ces besoins seront reformulés sous forme d'exigences, dans la même langue naturelle, mais sous une forme différente, plus formelle. L'expression des exigences est donc une traduction entre le langage (et souvent le jargon) des utilisateurs et un langage très précis, compréhensible par tous les acteurs. Entre les deux, une représentation sous forme graphique, sous forme de scénario, ou sous forme de prototype, permet d'obtenir une compréhension commune, de lever toute ambiguïté et incohérence.

## Utilité de l'étape d'analyse

Il est illusoire de penser que l'utilisateur ou son représentant va nous présenter une liste bien ordonnée de besoins bien formulés. Les besoins sont en général exprimés sous forme de grappes, dont il faudra trier les grains. Plusieurs besoins très différents peuvent être formulés dans une même phrase, comme « on a besoin d'avoir toutes les informations d'un client sous les yeux, et de pouvoir choisir dans un menu les différents produits à lui proposer, et le système doit réagir très vite sans jamais se bloquer ».

Dans cet exemple, le client :

- N'indique pas l'acteur (qui est « on » ?).
- Exprime plusieurs besoins dans une même phrase.
- Mélange des besoins fonctionnels et non fonctionnels.
- Mélange des besoins généraux (vision globale) et de détail (choix).

- Formule une solution (choix dans un menu) à la place d'un besoin.
- Formule des besoins non mesurables (« très vite »).
- Formule une exigence probablement irréaliste (« jamais »).

Au cours d'une réunion d'un groupe de travail, on peut recueillir une vingtaine de formulations de ce type. Et l'élaboration d'un cahier des charges peut nécessiter une quinzaine de réunions de travail. On admettra facilement que trois cents phrases de ce type mises bout à bout ne constituent pas un cahier des charges, mais une liste qui deviendra vite incohérente. Pour travailler efficacement, il sera donc nécessaire d'analyser, de reformuler, de trier et de classer les besoins au fur et à mesure qu'ils se présentent, ou en tout cas le plus tôt possible.

## Le processus d'analyse

---

Les objectifs de l'étape d'analyse sont les suivants :

- classer et structurer les exigences ;
- développer une compréhension partagée des besoins qui ont été recueillis ;
- détecter les incohérences, redondances et incomplétudes et faire le nécessaire pour les réduire, voire les éliminer.

Ces deux objectifs sont indissociables. Pour les atteindre, l'étape d'analyse fait largement appel à l'expérience et à la créativité, car il n'y a pas de méthode miracle pour détecter toutes les incohérences et incomplétudes. La méthode empirique, qui consiste à relire les exigences et à en discuter en groupe de travail, a son utilité, mais on en voit vite les limites. Ce chapitre va détailler des techniques spécifiques qui seront mises en œuvre, et en particulier :

- structurer et organiser les exigences par types ;
- établir un dictionnaire de données ;
- analyser les règles métier ;
- classer les exigences par priorités ;
- modéliser les exigences sous forme graphique ;
- réaliser une « maquette papier » ou un prototype ;
- élaborer des cas de test ;
- tester la faisabilité et le coût des exigences.

La figure 10-1 présente le processus général de l'étape d'analyse.

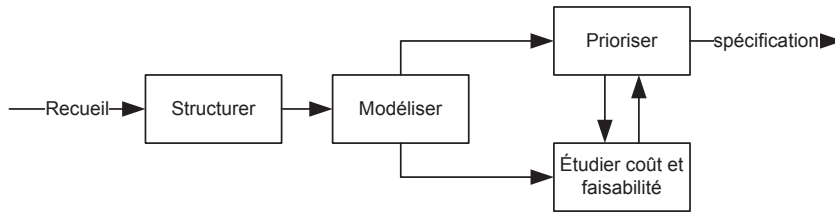


Figure 10-1 : Le processus d'analyse

Si l'analyste a une formation d'ingénieur, il sera tenté, du moins dans un premier temps, de concevoir l'étape d'analyse comme une activité purement technique. En réalité, il ne sortira rien de bon si les différentes parties prenantes, et en particulier les représentants des utilisateurs, ne sont pas impliquées. L'analyste doit maîtriser UML (ou un autre langage graphique), mais il doit être suffisamment pédagogue pour rendre les diagrammes compréhensibles par tout lecteur potentiel.

## Structurer et organiser les exigences

Les exigences doivent être classées en catégories, soit au fil de l'eau lors du recueil, soit au plus tard pendant l'élaboration du document d'exigences (cahier des charges). La classification qui suit est inspirée de Karl Wiegers (voir bibliographie), mais la plupart des classifications s'en approchent.

**Les objectifs** sont des exigences de haut niveau. Un objectif est formulé sous forme de phrase avec un verbe à l'infinitif, par exemple « diminuer de 20 % le temps d'attente au guichet ». Les objectifs sont exprimés, dans la majorité des cas, par le donneur d'ordre. Le verbe sous-entendu est le verbe *vouloir* (ou *devoir*, dans le sens d'une volonté) : nous (l'entreprise) *voulons* diminuer le temps d'attente (il s'agit là de la volonté du maître d'œuvre).

**Les cas d'utilisation** ou scénarios correspondent à des besoins des utilisateurs et s'expriment également comme une phrase à l'infinitif, par exemple « enregistrer un nouveau client ». La phrase sous-entendue *est avoir besoin de*, exprimée à la première personne : « j'ai besoin d'enregistrer tout nouveau client ».

**Les règles** sont des exigences réglementaires ou des règles métier (également appelées règles de gestion). Le verbe sous-entendu ou explicite est *devoir* : le logiciel doit se conformer à telle réglementation, dans des conditions bien définies. Par exemple : « un retrait d'espèces ne peut être effectué que si le compte est positif » (formulé autrement, le compte *doit* être positif pour qu'un retrait puisse être effectué).

**Les exigences fonctionnelles** constituent le cœur et souvent la partie la plus importante d'un cahier des charges. Elles expriment un *comportement* requis de la part du système. Elles dérivent des catégories précédentes. Par exemple « si le retrait d'espèces n'est pas autorisé, le système envoie un message au client ».

**Les exigences de qualité**, également appelées *exigences non fonctionnelles*, bien qu'elles n'en constituent qu'une partie. Elles s'expriment sous forme d'adjectif (rapide, facile...) ou de la propriété correspondante (rapidité, facilité...).

**Les exigences d'interface** qui expriment le besoin d'une communication entre le système à l'étude et le monde extérieur : matériel, logiciel et personnes.

**Les contraintes techniques**, comme l'utilisation d'un système ou d'un langage particulier, ou de technologies spécifiques, comme un protocole de communication ou de sécurité. Il est important d'analyser l'origine de ces contraintes et leur utilité réelle, car elles peuvent avoir des répercussions négatives sur la fonctionnalité ou la qualité.

**Les exigences sur les formats de données**, comme un code postal en cinq chiffres, un code pays, etc.

**Les suggestions de solutions** qui n'ont pas lieu de se retrouver en tant que telles dans un cahier des charges, et qu'il faudra analyser pour remonter à une exigence d'un des types précédemment définis. Par exemple, si l'exigence est « la somme doit s'afficher en rouge », il est important de savoir pourquoi. Cela peut correspondre à une règle métier (certains chiffres doivent s'afficher en rouge dans telle condition), d'une règle d'ergonomie (homogénéité : l'affichage doit être le même d'une application à l'autre) ou d'un souhait personnel.

**D'autres informations ou exigences** peuvent être formulées. Elles pourront ou non se retrouver dans le cahier des charges, ou en préambule, ou en annexe de celui-ci : description de l'existant, contraintes sur les délais, sur les coûts, sur la formation des utilisateurs, le découpage en lots lors d'un appel d'offres, et toutes informations utiles à ceux qui vont répondre au cahier des charges.

Il s'agit là d'une première classification qui, avec l'habitude, se fait mentalement au cours du recueil des exigences. Une classification plus précise se fait lors de la rédaction du cahier des charges, et dépend donc précisément du modèle de cahier des charges utilisé. D'ailleurs, disposer d'un squelette (ou modèle) de cahier des charges sert précisément à cela : c'est une armoire de rangement pour les exigences.

## Établir un dictionnaire de données

---

Comme les exigences fonctionnelles, la description des données constitue une passerelle de communication entre maîtrise d'ouvrage (le vocabulaire des utilisateurs) et maîtrise d'œuvre (la compréhension des concepteurs et développeurs).

Les mêmes données apparaissant d'une exigence à l'autre, il est utile de les spécifier une fois pour toutes. Par exemple :

```
Client = id_client [9 chiffres] + nom [15 lettres] + prénom  
[15 lettres]
```

La syntaxe peut être plus ou moins complexe, en fonction de l'usage qui sera fait des spécifications d'exigences (développement spécifique ou choix de progiciel). Certaines interfaces (normes d'interopérabilité) peuvent nécessiter une description extrêmement précise des formats de données échangées, mais c'est rarement le cas pour un cahier des charges fonctionnel élaboré dans le cadre d'un appel d'offres, par exemple.

L'important est que toutes les personnes intéressées soient d'accord sur le contenu de telle ou telle donnée, afin d'éviter toute ambiguïté.

## Analyser les règles métier

---

### Qu'est-ce qu'une règle métier ?

Les règles métier, également appelées règles de gestion (en anglais *business rules*) dérivent des lois, règlements, procédures, codes de bonnes pratiques en vigueur pour un métier donné, règles de calcul, etc. Ce sont là des exigences en puissance, mais elles sont généralement énoncées sous une forme inappropriée pour apparaître dans un cahier des charges. Autrement dit, ce sont des intermédiaires entre un texte ou une pratique (texte réglementaire, pratique professionnelle) et une exigence. Plus précisément, une règle de gestion peut être :

- un fait ;
- une contrainte (seul un acteur X peut accomplir l'action Y) ;
- une règle d'action (un acteur X doit accomplir l'action Y) ;
- une inférence (si... alors...) ;
- une règle de calcul (par exemple, calcul de la TVA).

Les règles métier ne sont pas des besoins utilisateurs et ne sont pas considérées comme des exigences à proprement parler. En général, elles n'apparaissent pas en tant que telles dans un cahier des charges, mais y sont déclinées sous forme d'exigences.

Plutôt qu'un long discours, donnons quelques exemples de règle métier :

- Tout médicament est livré à la pharmacie dans un emballage avec un code-barres (fait).
- Le directeur d'un établissement de santé établit la liste des personnes habilitées à prescrire des médicaments (règle d'action).
- Le pharmacien doit analyser la prescription avant de livrer un médicament (règle d'action).
- Le pharmacien doit s'assurer qu'il n'y a pas interaction médicamenteuse entre plusieurs médicaments prescrits (règle d'action).
- Si la date de péremption inscrite sur l'emballage est supérieure à la date du jour, alors le produit est périmé (inférence).

Il est clair que toutes ces règles n'ont pas à se retrouver dans une spécification d'exigences. Il serait inutile de les inclure dans le cahier des charges lui-même, du moins sous cette forme, car elles n'ont pas d'impact direct sur le produit. Mais, d'une part, elles participent à l'analyse des exigences et, d'autre part, il est nécessaire, pour des raisons de traçabilité et de maintenabilité, de pouvoir connaître en permanence le lien entre une règle métier et une exigence du cahier des charges<sup>1</sup>.

---

1. De fortes contraintes de traçabilité et de maintenabilité peuvent rendre indispensable le recours à un outil de gestion des exigences (voir au chapitre 19).

---

### De la règle métier à l'exigence

À l'origine, les règles métier n'ont pas été écrites pour figurer dans un cahier des charges. Même si cela peut en choquer certains, on peut dire que, vis-à-vis du cahier des charges, ce sont des « exigences brutes ». Pour en faire des exigences aptes à entrer dans un cahier des charges, le travail de recueil, d'analyse et de spécification est analogue à celui de l'analyse des besoins. Il consiste à :

- recueillir les règles de gestion
  - rechercher les sources des règles de gestion potentielles (législation, normes, recueils de pratiques professionnelles, protocoles...) qui sont en rapport avec l'objectif et dans le champ de l'étude,
  - recueillir l'ensemble des règles, procédures, pratiques, susceptibles d'avoir un impact sur les exigences ;
- analyser ces éléments
  - filtrer en fonction des objectifs,
  - rechercher les incohérences, redondances et incomplétudes,
  - si nécessaire, représenter ces éléments en diagrammes, sous la forme appropriée ;
- traiter les règles de gestion
  - formuler les éléments précédents sous forme de règles métier,

- stocker ces règles dans le référentiel approprié,
  - décider lesquelles de ces règles métier seront automatisées ;
- spécifier
  - formuler ces règles sous forme d'exigences,
  - inclure ces exigences dans le cahier des charges ;
- gérer
  - maintenir la traçabilité entre les règles de gestion et les exigences fonctionnelles en aval,
  - analyser l'impact d'une règle de gestion sur l'ensemble des exigences métier.

La différence avec le traitement des besoins réside donc dans le stockage des règles métier dans une zone particulière, en vue de les inclure, si nécessaire, dans le cahier des charges. Cette zone de stockage peut être un simple tableau, une base de données, ou mieux, un outil de gestion des exigences.

Les règles métier vont généralement donner lieu à des exigences fonctionnelles de haut niveau, parfois appelées *exigences métier* (*business requirements*). Elles seront déclinées sous forme d'exigences plus fines, en fonction du niveau de détail requis.

## Définir les priorités d'un projet

---

### Pourquoi et quand définir les priorités ?

Tout projet est limité par le budget alloué, les délais exigés, le risque admissible et les ressources disponibles. Il est donc indispensable de définir les priorités entre exigences. La priorisation des exigences est utile :

- lors de la définition des objectifs, (dans toute la mesure du possible) ;
- lors du développement des exigences (élaboration du cahier des charges) ;
- lors du choix d'une solution.

La gestion des priorités permettra de résoudre les conflits au plus tôt, de faire des compromis, et éventuellement de planifier la mise en œuvre de certaines exigences pour une version ultérieure du système.

Il est important de définir les priorités entre fonctions (ou exigences non fonctionnelles) le plus tôt possible, quitte à revoir la liste des priorités par la suite, avec le recul. Cela minimise le risque de voir des conflits s'envenimer ou de devoir remettre en cause l'existant au dernier moment.

### Comment définir les priorités ?

Gérer les priorités est loin d'être une tâche facile. Tout utilisateur a tendance à penser que a) toutes ses exigences sont prioritaires sans exception et que b) ses exigences sont prioritaires sur celles des autres.

Les priorités ne peuvent être définies par les seuls utilisateurs (ou leurs représentants). Elles doivent être confrontées au réel, c'est-à-dire à ce qui est faisable ou pas, et à quel coût. Il est donc nécessaire d'organiser des réunions entre experts métier (utilisateurs ou représentants) et experts techniques (développeurs, chefs de projet ou leurs représentants) et de confronter les points de vue.

Il n'y a pas de méthode miracle, la gestion des priorités reposant sur des principes de négociation. En cas de conflit entre priorités, il faut se poser les questions suivantes :

- Qui serait insatisfait si cette exigence n'était pas mise en œuvre ?
- Peut-on satisfaire cette exigence par d'autres moyens ?
- Comment se comporterait le système sans cette fonction ?
- Le système dans son ensemble serait-il mis en cause ?
- Quelle est la limite acceptable en termes de délais ? de coûts ?
- Est-on prêt à assumer le risque de supprimer cette fonction ?

La priorité d'une exigence est fonction de son urgence et de son importance (principe d'Eisenhower) :

- les exigences urgentes et importantes seront en haute priorité ;
- les exigences importantes, mais non urgentes, viendront ensuite ;
- les exigences urgentes et non importantes sont de priorité faible ;
- les exigences ni urgentes ni importantes seront mises de côté.

Outre l'urgence et l'importance, doivent entrer en ligne de compte le risque à faire et à ne pas faire (ce que l'on gagne en faisant, ce que l'on perd en ne faisant pas), le coût de réalisation, ainsi que le risque lié à la réalisation (lié au manque de savoir-faire de l'équipe de développement, à l'immaturité des technologies mises en œuvre, etc.).

Il est facile de construire (ou de trouver sur Internet) une matrice de priorisation des exigences, permettant de faire une somme pondérée des différents paramètres.

### La méthode des cent points : mode opératoire

La méthode des cent points est un grand classique. Elle est très utilisée par les consultants pour animer un groupe de travail de sélection de logiciel. Dans ce dernier cas, les exigences s'appellent critères de



choix, mais l'idée est la même. Sous sa forme simplifiée, les étapes de la méthode sont les suivantes :

1. Les participants dressent la liste des exigences à prioriser. On utilisera de préférence un tableur et la liste ne doit être ni trop longue, ni trop courte (règle du pouce : imprimée, la liste doit tenir sur une page A4).
2. On se fixe la règle suivante : chaque exigence aura un poids compris entre 1 (peu prioritaire) et 4 (très prioritaire), et la somme des poids des exigences devra être égale à 100. On peut jouer sur ces chiffres, mais on évitera de les modifier en cours de route.
3. Les participants attribuent à chaque exigence un poids. Ce poids doit être attribué par consensus entre participants. Si un consensus ne peut être trouvé, on vote (c'est un pis-aller) ou quelqu'un doit trancher (c'est un pis-aller de plus). C'est là que les talents d'animateur de l'analyste seront utiles.
4. Lorsque chaque exigence aura reçu un poids, si la somme des poids est inférieure ou supérieure à 100, on cherchera à augmenter ou réduire le poids de certaines exigences jusqu'à atteindre 100. Ici aussi, on cherche le consensus. Si on utilise un tableur, il est facile de jeter de temps en temps un coup d'œil sur la somme des poids. Avec un peu d'expérience, l'animateur sait qu'il doit forcer sur les pondérations ou au contraire les modérer, bien avant d'arriver au bout de la liste.
5. Lorsque toutes les exigences sont pondérées et que la somme des poids est égale à 100, les exigences sont priorisées.

Il est clair que cette méthode demande du doigté de la part de l'animateur (analyste ou consultant) et une atmosphère consensuelle, sinon la séance tournera vite au pugilat. À certains moments, l'animateur peut assouplir les règles : ajouter quelques exigences supplémentaires, autoriser les dépassements, ou modifier la liste en cours de route. Ces entorses sont somme toute normales, la « règle du 100 » étant plutôt arbitraire.

Il arrive d'ailleurs souvent qu'en discutant, les participants remettent en question les exigences elles-mêmes, ou fassent émerger de nouvelles exigences. Ce phénomène est normal, voire souhaitable, mais doit être contrôlé pour éviter les dérives : augmentation exponentielle du nombre d'exigences, spécifications rampantes, ou retours incessants en arrière.

Une variante de la méthode consiste à donner à chaque participant cent points, qu'il pourra utiliser à sa guise. S'il y a cinq participants, la somme pondérée des priorités sera égale à 500. Le consultant ou analyste qui anime le groupe n'a pas de points. Cette variante permet d'aller plus vite, mais en évitant la recherche d'un consensus, elle n'encourage pas la réflexion sur les exigences elles-mêmes.

La méthode peut être enrichie, en donnant des poids liés non seulement au besoin des utilisateurs, mais aussi au coût, au risque, au temps de développement, etc.

### Autres méthodes

Même avec une matrice simple (somme pondérée des bénéfices, coûts, risques) on obtient le plus souvent des résultats satisfaisants. Le plus important est d'amener les différents acteurs à se mettre d'accord sur un processus de priorisation et à travailler ensemble pour définir les priorités.

Pour les projets importants, des méthodes plus complexes de priorisation, comme AHP (*Analytical Hierarchical Process*) ou QFD (*Quality Function Deployment*) peuvent être mises en œuvre. La méthode AHP est une méthode d'aide à la décision qui consiste à comparer deux à deux des critères (et dans le cas des exigences) classés hiérarchiquement. Le nombre total de comparaisons est  $n \times (n-1)/2$  où  $n$  est le nombre d'exigences à prioriser à chaque niveau hiérarchique, ce qui augmente très vite le nombre de comparaisons à faire. Cette méthode est puissante, mais en pratique demande à être outillée<sup>2</sup>.

---

2. En France, la société Adexys commercialise une gamme d'outils d'évaluation basés sur la méthode AHP.

### Modéliser sous forme graphique

---

La modélisation graphique est à la fois un outil d'analyse, de communication entre acteurs, et de validation. Certains traitements ou processus sont plus simples à comprendre lorsqu'ils sont représentés graphiquement, et les éventuelles incohérences « sautent aux yeux ». Un bon schéma, c'est bien connu, vaut bien cent pages de texte.

Cependant, ne nous faisons pas d'illusions. Contrairement à ce que l'on écrit parfois, la modélisation sous forme graphique n'est pas un outil universel et ne remplace pas le texte écrit. Il y a à cela plusieurs raisons :

- Contrairement à ce que l'on pense souvent, des schémas qui paraissent très simples à certains seront difficiles à comprendre pour d'autres.
- Un schéma peut être facile à lire, mais beaucoup plus difficile à élaborer qu'un texte.
- Élaborer un schéma paraît facile, mais c'est là une simplicité trompeuse, car il faut utiliser un formalisme précis et connaître la sémantique sous-jacente.
- À moins d'un formalisme extrême, une même représentation peut avoir des interprétations différentes selon les auteurs et lecteurs, et induire en erreur.

- Il n'existe pas de notation graphique universelle qui permettrait de représenter tous les besoins.

Les diagrammes peuvent être insérés dans un cahier des charges, en complément du texte écrit. Ils peuvent également être très utiles en tant que représentations intermédiaires, uniquement dans un but d'analyse, sans nécessairement apparaître dans le cahier des charges.

### La vraie valeur d'un diagramme

Un diagramme ne sert pas seulement à communiquer des exigences au maître d'œuvre, ni même à les faire valider par la maîtrise d'ouvrage. Il sert aussi, et surtout, à changer de point de vue.

En effet, analyser un problème, c'est le décomposer en ses différents constituants, dans le but de mieux le comprendre. Le représenter sous différents formalismes oblige à l'observer sous plusieurs angles et à le décomposer de plusieurs manières différentes. Et surtout, cela oblige à « changer de cerveau », car notre cerveau traite différemment le texte et les images. Cela augmente considérablement la probabilité de détecter une incohérence ou une incomplétude dans la spécification.

On donne ici quelques diagrammes parmi les plus utilisés. Les paragraphes qui suivent n'ont pas pour objectif de remplacer un ouvrage sur l'analyse des systèmes d'information, mais de montrer leur utilité pratique dans la définition des besoins.

## Le diagramme d'activités ou carte de processus

C'est le plus classique des diagrammes. On le trouve dans diverses méthodologies, sous des noms différents, avec des symbolismes variés. Il s'appelle *process map* en anglais, et son équivalent UML est le *diagramme d'activité*. Il permet de représenter les étapes d'un processus métier, leur séquence, leurs entrées et sorties, les acteurs responsables de chaque étape. Ces diagrammes peuvent être utilisés pour représenter un processus existant ou à venir, automatisé ou manuel. C'est donc un outil assez généraliste.

Les étapes peuvent être réparties sur des « bandes » (en anglais *swim lanes*, par analogie avec les couloirs d'une piscine), chaque bande étant sous la responsabilité d'un acteur. Certains analystes réservent une bande aux étapes automatisées ou à automatiser.

En plus d'être le « couteau suisse » de l'analyste, un diagramme de processus est un excellent outil de dialogue avec les utilisateurs, et de négociation avec la maîtrise d'ouvrage, à condition :

- de tenir sur une page A4 ;
- d'être « autoexplicatif » ;
- de ne pas comporter trop d'étapes (une douzaine au plus).

Les grands schémas synoptiques sur une grande feuille A3 ou A2 peuvent être utiles aux analystes et concepteurs, rompus aux schémas complexes. Mais ils ne devraient pas sortir de leurs tiroirs. Une « usine à gaz » risque d'effrayer un interlocuteur et d'avoir l'effet exactement inverse de celui escompté.

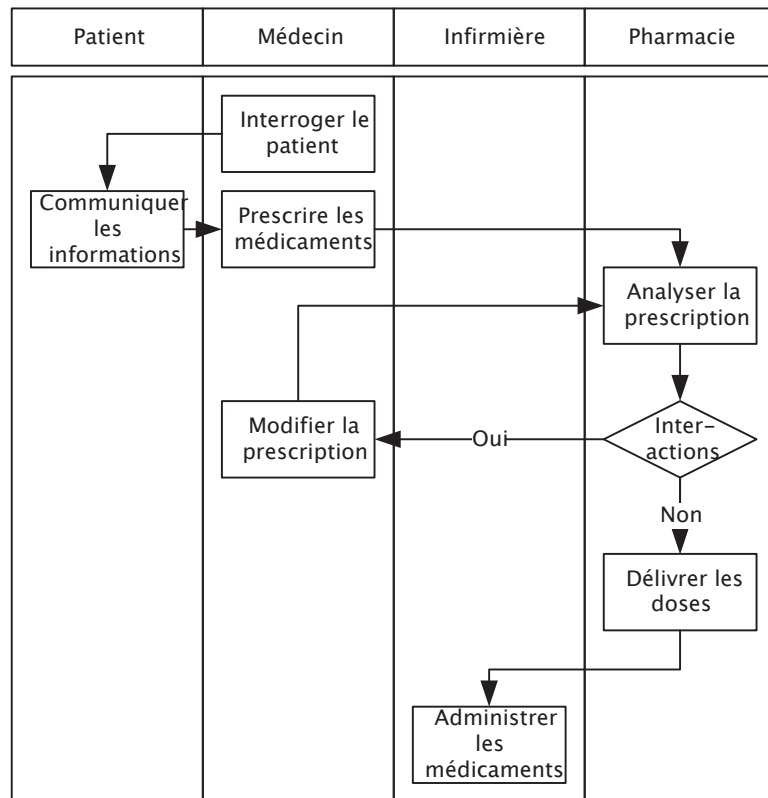


Figure 10-2 : Carte des processus

### Le diagramme de flux de données (DFD)

Un diagramme de flux de données (ou plus simplement, diagramme de flux) permet de représenter la manière dont les données sont traitées (transférées, modifiées) par le système d'information (personnes et applications).

Plusieurs modèles de représentation existent. Dans le modèle le plus couramment utilisé (figure 10-3) :

- Les processus sont représentés par des cercles.

- Les entités externes au système (personnes physiques ou morales, groupes de personnes ou autres systèmes) sont représentées par des rectangles<sup>3</sup>.
- Les flux de données sont représentés par des flèches.
- Les zones de stockage (bases de données, fichiers...) sont représentées par deux segments parallèles.

3. Remarquons que les rectangles ne font pas partie du système.

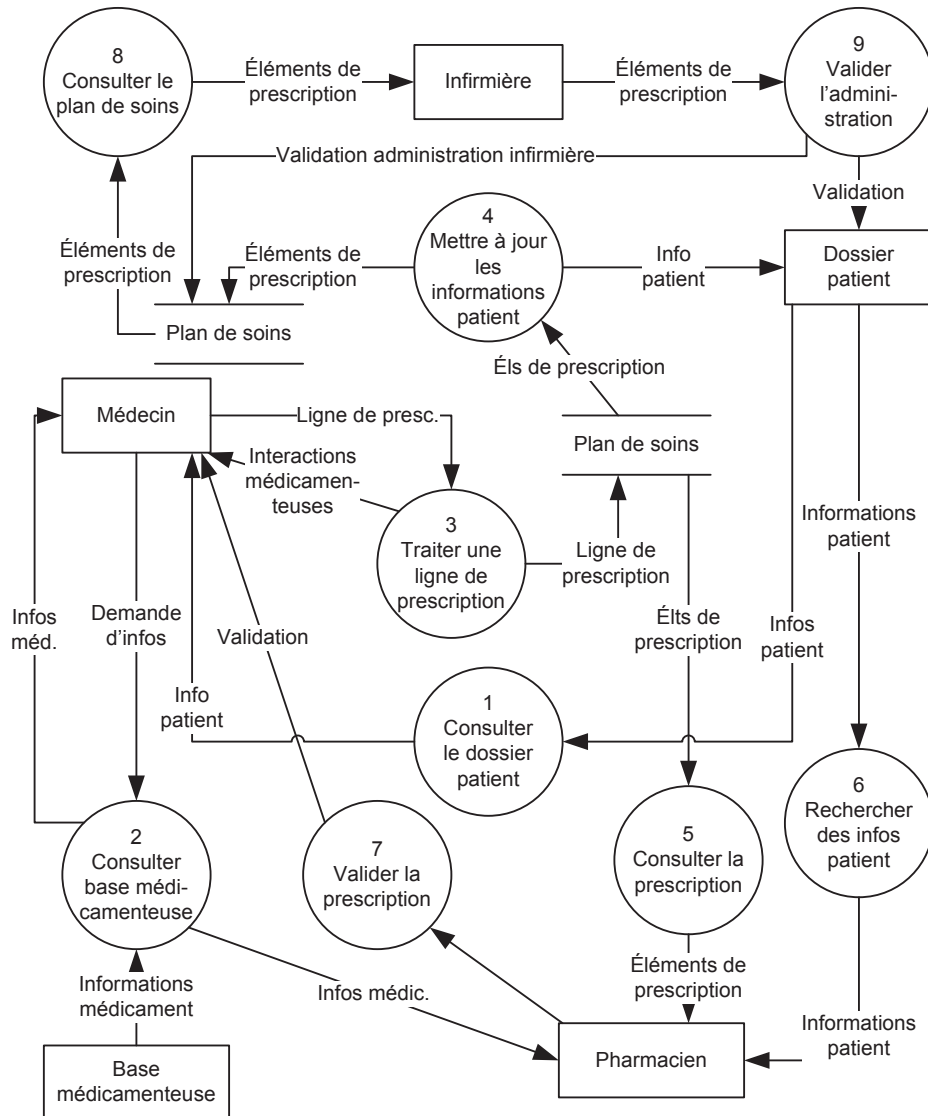


Figure 10-3 : Un diagramme de flux de données

Chaque processus d'un diagramme peut être éclaté en processus plus détaillés dans de nouveaux diagrammes de flux. À ce propos, remarquons que le diagramme de contexte, dont il a déjà été question dans cet ouvrage, est le plus simple des diagrammes de flux. Il représente le processus global correspondant au système à l'étude, entouré des personnes ou logiciels avec lesquels il réagit.

Le diagramme de flux met en évidence le partage du travail entre le système à l'étude et son environnement. C'est donc un bon moyen de donner une vision globale des différents processus et de leurs relations entre eux, et il est généralement bien compris des utilisateurs. Il peut donc être utilisé lors du recueil des besoins dans le but d'éclaircir ou de valider ce qui a été formulé oralement. Bien entendu, il peut apparaître dans un cahier des charges.

Quelques règles d'élaboration du diagramme :

- Tout processus doit comporter une flèche en entrée et une flèche en sortie.
- Une flèche ne peut pas relier directement deux processus.

Quelques règles pratiques :

- Numéroter les processus facilite grandement la lecture.
- Ne pas mettre trop d'entités dans un diagramme. Suivre la règle du « sept plus ou moins deux ». Au-delà de neuf, le diagramme est illisible.
- Si un processus est complexe, on peut le modéliser dans un autre diagramme. Dans ce cas, on utilisera une numérotation pointée pour numéroter les sous-processus (par exemple, le processus numéro 4 se décomposera en 4.1, 4.2, etc.).

### Le diagramme de séquence

Le diagramme de séquence (figure 10-4) permet de modéliser les flux d'informations entre acteurs, en faisant ressortir l'enchaînement chronologique des activités d'un processus.

Horizontalement, de gauche à droite, on représente les « lignes de vie » des différents acteurs. Des messages, synchrones ou asynchrones, transitent d'un acteur à l'autre. Verticalement, on peut lire l'enchaînement des actions dans le temps.

Le diagramme de séquence est un outil d'analyse plus que de recueil ou de spécification. En dehors des domaines techniques où la synchronisation des tâches est complexe (télécoms), il est rare de le rencontrer dans un cahier des charges. Il est en revanche très utile pour alimenter la réflexion permettant de décider des opérations à automatiser et sur les processus à optimiser à la suite d'une informatisation. Le diagramme de

la figure 10-4 représente un processus manuel. L'automatisation de ce processus permettra de supprimer environ la moitié des tâches (mises à jour, transmission manuelle d'informations et archivage).

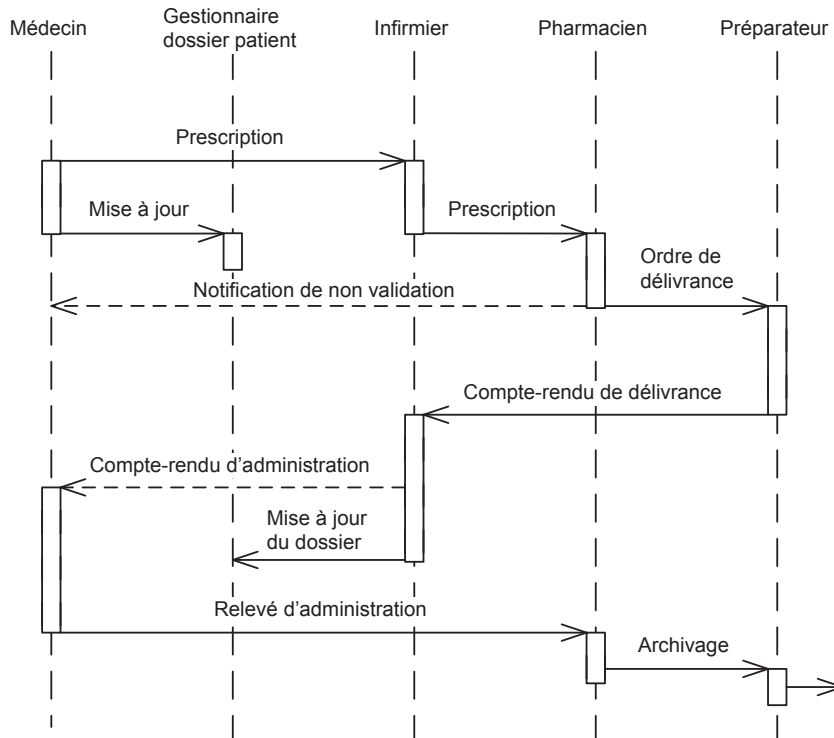


Figure 10-4 : Un diagramme de séquence

## Le diagramme entité-association ou diagramme de classes

Ce diagramme, très technique, peut être utile dans certains cas, mais n'a pas toujours sa place dans un cahier des charges (figure 10-5). Il permet de représenter les attributs des données et les relations entre les données du système. Dans le cadre de la phase de définition des besoins, élaborer ce modèle permet d'avoir une vision exhaustive des données manipulées par le système. C'est donc un *modèle conceptuel des données*, indépendant des aspects techniques, que l'on utilise. Après transformation, il aboutira à un modèle physique, qui lui, dépend de l'implémentation physique (base de données).

L'expérience montre que la lecture (sans parler de l'élaboration même) d'un tel modèle n'est pas toujours aisée pour les utilisateurs non informaticiens. Il est utile d'accompagner le diagramme de texte explicatif.

Lors du recueil, les questions à se poser, et à poser aux représentants des utilisateurs, sont :

- Quelles sont les données manipulées par le système ?
- Quelles données le système devra-t-il stocker ?

Les entités sont des données, ou plus généralement, des ensembles de données : un client, un compte, une facture... Elles comportent des attributs (par exemple, le nom du client). Les entités sont généralement représentées par des rectangles.

Les relations comportent, pour chaque entité reliée, une cardinalité. Par exemple, la relation entre un client et ses comptes en banque est 1-N, signifiant que chaque client peut avoir un ou plusieurs comptes, et chaque compte appartenant à un client et un seul. Les relations sont généralement représentées par des lignes dont la terminaison indique la cardinalité. Les relations peuvent elles-mêmes comporter des propriétés (des ovals avec Merise, des rectangles avec UML).

Les questions supplémentaires à poser sont donc :

- Quelles sont les données contenues dans cette entité ?
- Quelles sont les relations entre ces deux données ?

Il y a des dizaines de façons de représenter les entités et les relations (à la Merise, à la UML, rectangles, ovals, losanges...). Le plus important est que tous les acteurs utilisent le même langage graphique. On s'adaptera à la culture ou aux standards du client ou de l'entreprise pour laquelle on travaille. La figure 10-5 utilise un formalisme proche d'UML pour représenter les entités et relations dans le domaine de la prescription médicale.

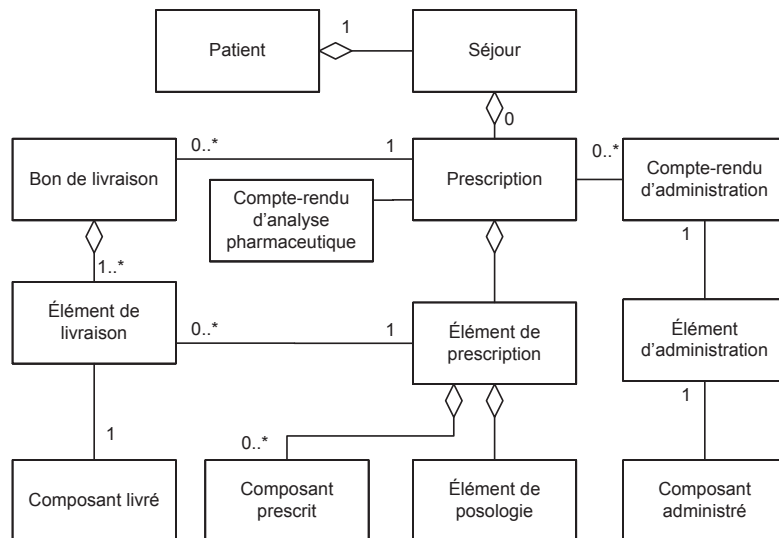


Figure 10-5 : Un diagramme entité-association



Les adeptes d'UML utilisent des diagrammes de classe. Une classe d'objets contient à la fois des attributs et des traitements (ou opérations). En principe, un modèle de classes devrait être différent d'un modèle entités-relations. Dans la pratique, on s'aperçoit que les modèles UML sont souvent réduits à des entités et des relations (même dans les ouvrages consacrés à UML).

L'idée ici n'est pas d'entrer dans les détails de la conception d'un modèle de données (il y a d'excellents ouvrages pour cela), mais de montrer comment un tel modèle peut être utilisé comme outil d'analyse et de spécification dans le cadre d'un cahier des charges. Les six tâches (imbriquées et cycliques, bien sûr) sont les suivantes :

- **Identifier les entités.** Les entités sont des objets ou des agrégations de données. On les représente par des rectangles.
- **Identifier les attributs** (propriétés). On indiquera celles utiles au système à l'étude (rappelons qu'il ne s'agit pas, au stade du cahier des charges, de modéliser une base de données). L'un de ces attributs est l'identifiant (équivalent à la clé primaire sur un modèle physique).
- **Décrire** les entités et attributs, de préférence dans le dictionnaire de données.
- **Identifier les relations** entre données. On peut commencer ce travail à tête reposée, mais le modèle ainsi construit (ou plutôt, en cours de construction) sera dès que possible soumis à l'avis des représentants des utilisateurs.
- **Définir les cardinalités** des relations.
- **Vérifier la cohérence** de l'ensemble et remonter à la première étape si nécessaire.

Au risque de nous répéter, rappelons-le : il s'agit ici de modéliser les besoins, et non la solution. Au fur et à mesure que l'on décrira de nouvelles exigences fonctionnelles, on les confrontera au modèle de données, en cherchant à éliminer les incohérences. Le processus de construction est donc itératif et imbriqué aux autres processus.

### Principe de simplicité

Le principe de simplicité a été énoncé par plusieurs philosophes. Une des formulations les plus connues de ce principe est le rasoir d'Ockham, qui s'énonce ainsi : « *Entia non sunt multiplicanda praeter necessitatem* », littéralement « Les entités ne doivent pas être multipliées par-delà ce qui est nécessaire ». On gagne à utiliser ce principe sur tous les types de diagramme, et particulièrement sur le diagramme entités-associations. Inutile de procéder à un découpage fin en microentités si cela n'aide pas à la réflexion sur les besoins. Parfois, ce principe gagne même à être poussé jusqu'à sa limite : ne pas faire de diagramme là où cela ne s'avère pas nécessaire.

## Le diagramme états-transitions

Rappelons qu'un diagramme états-transitions représente, sur le plan informatique, un automate d'états finis. De ce fait, il permet de représenter des processus de manière concise, précise et non ambiguë, facilement compréhensible par les concepteurs et réalisateurs du système.

On retrouvera ce type de schéma dans des cahiers des charges pour des logiciels temps réel, mais pas seulement. Il peut être très efficace pour décrire un processus administratif, par exemple. La figure 10-6 présente un diagramme d'états-transitions pour un logiciel de gestion de dossier. Il a été élaboré par un groupe de travail composé de fonctionnaires de l'administration publique, animé par un assistant à la maîtrise d'ouvrage.

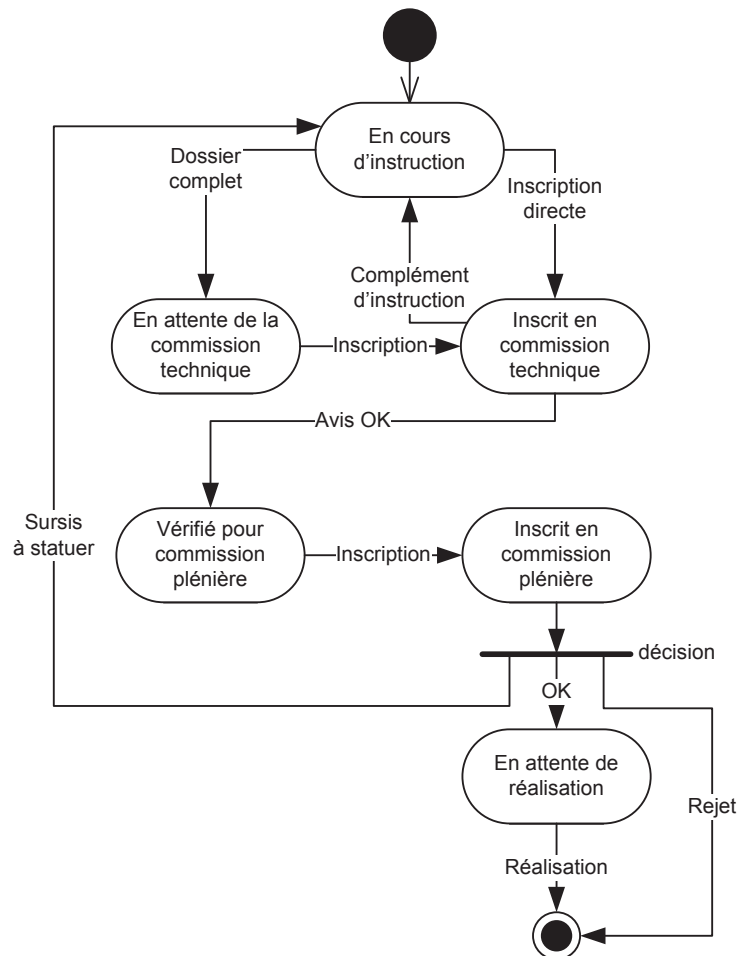


Figure 10-6 : Un diagramme états-transitions

Voici une séquence classique d'élaboration de ce type de diagramme :

1. Lors d'une première réunion du groupe de travail avec le donneur d'ordre (direction, administration centrale), l'analyste recueille des informations sur les étapes du cheminement du dossier. Il ne présente pas de diagramme (risque de confusion).
2. En interviews individuelles d'utilisateurs sur le terrain, l'analyste recueille les pratiques (qui peuvent différer sensiblement de la procédure officielle).
3. Seul, le consultant élabore le diagramme états-transitions, en notant les incohérences, en tentant de les résoudre si possible.
4. L'analyste parcourt les textes réglementaires. À nouveau, il note les écarts et les incohérences. Il complète ou modifie le diagramme.
5. Lors d'une deuxième réunion du groupe de travail, l'analyste présente le diagramme états-transitions au donneur d'ordres et lui demande son avis. Si aucun consensus n'est trouvé, on peut remonter aux étapes 2 à 4.
6. L'analyste met le diagramme au propre. Il demande au groupe de travail de le valider.
7. Si cela est possible, le diagramme est envoyé à la maîtrise d'œuvre pour avoir son avis.

Les questions à poser aux différents acteurs sont :

- Quel est le cheminement (du dossier, d'un objet) ?
- Dans quels états peut se trouver (le dossier, l'objet) ?
- Quel événement va modifier (le dossier, l'objet) ?
- Que se passe-t-il suite à cet événement ?
- Que se passe-t-il si cet événement n'a pas lieu ? (exception)

Comme pour les autres types de diagramme, le diagramme états-transitions peut servir à modéliser un processus existant ou un processus à informatiser. Il est souvent long à constituer, surtout lorsque les interlocuteurs sont plus familiarisés avec des procédures écrites (procédures administratives) qu'avec des schémas.

### Les organigrammes, arbres et tables de décision

Des règles de gestion fort complexes, par exemple sur le plan juridique, peuvent parfois s'exprimer très facilement, et devenir très claires pour la maîtrise d'œuvre, si elles sont exprimées sous forme de tables ou d'arbres de décision, ou sous forme d'organigramme.

Lors de l'élaboration d'un cahier des charges pour une application de portée nationale, nous avons déterminé 24 réponses à la question de savoir qui, parmi le père, la mère ou autre représentant légal à droit

à l'information sur un enfant, en fonction de la situation. Un tableau à 24 lignes et 10 colonnes a rendu la situation beaucoup plus facile à comprendre. Quant aux informaticiens, selon leurs propres termes, « ils n'avaient plus qu'à coder ».

### La boîte à outils

La modélisation sous forme graphique est une boîte à outils. Pour représenter la même information, on a parfois le choix entre plusieurs outils. Inversement, un outil peut avoir plusieurs usages. Il ne s'agit pas de « tout » représenter avec le même outil, ni d'utiliser tous les outils à tout prix. Rappelons que l'objectif premier est ici de se faire comprendre de tous les acteurs et d'ajouter de la précision au texte. Par exemple, dans un cahier des charges pour une application de gestion, on pourra utiliser un diagramme états-transitions pour montrer le circuit d'une demande, suivi d'un diagramme de séquence, ou un seul diagramme de flux. La question à se poser est : comment représenter l'information de manière à ce qu'elle soit compréhensible de tous, sans ambiguïté ?

Il y a un travers dans lequel nous avons tous tendance à tomber : nous n'utilisons que les outils que nous maîtrisons. Et comme nous maîtrisons mieux les outils que nous utilisons souvent, nous finissons par n'utiliser qu'un outil ou deux. Pour ne pas tomber dans ce travers, il ne faut pas hésiter à expérimenter.

## Maquettes et prototypes

---

### La maquette papier

Une maquette est un modèle qui préfigure un produit futur. Rappelons qu'à la différence d'un prototype, la maquette est toujours un produit jetable. Le but ici n'est pas de commencer à développer le produit, ni même à le spécifier, mais de visualiser son aspect ou son comportement. Il ne s'agit pas de spécifier le produit, et encore moins de développer un prototype. Plus le support sera temporaire<sup>4</sup> (tableau blanc, notes auto-adhésives), et moins on sera tenté de faire des spécifications détaillées et de concevoir le produit.

La maquette papier est un moyen de recueillir des exigences, de les analyser interactivement et de les faire valider rapidement par les interlocuteurs. Concrètement, cela consiste à schématiser sur le papier, ou sur un tableau blanc, un dessin d'écran, par exemple. Plusieurs dessins d'écrans sur feuilles de papier A4 disposés sur une grande feuille de papier blanc permettent de simuler des enchaînements d'écrans et des interactions avec les utilisateurs. On peut utiliser des moyens d'expression plus

---

4. Maquette « faible fidélité » (*low-fi*) comparée à un prototype.

riches, comme des outils de présentation, HTML, voire un dessin animé. Mais l'idée est toujours la même.

Une maquette papier est utile pour faire visualiser les fonctions attendues à des personnes peu habituées à travailler sur des modèles abstraits, de les aider à imaginer eux-mêmes une préfiguration du futur produit. Une fois que les participants se sont mis d'accord, il est plus facile de formaliser ces exigences, sous forme de cas d'utilisation par exemple.

### Le prototypage

Contrairement à la maquette papier, un prototype est « vivant ». C'est du logiciel. Il peut être jetable ou au contraire réutilisable, si on a l'intention de l'intégrer dans le produit fini. Prototyper, c'est donc entrer provisoirement (et par exception) dans la phase de réalisation pour expérimenter un fragment du produit à venir.

Plusieurs formes de prototypes sont envisageables, en particulier :

- Le prototype « horizontal » : il simule l'application visuellement (enchaînement d'écrans), sans que toutes les fonctions soient présentes. Il est utilisé en particulier pour montrer et tester les enchaînements d'écrans. Il est visuellement proche de l'application à venir, mais les écrans sont « creux » ou « muets » : les fonctions ne sont pas développées et ne seront pas exécutées.
- Le prototype « vertical » : une seule fonction est complètement développée. Ce type de prototype est utilisé pour tester une fonction ou une propriété du logiciel, comme la fiabilité, la sécurité, ou le temps de réponse.

En pratique, le prototypage est faisable si une équipe de développement est présente et interagit avec la maîtrise d'ouvrage. C'est le cas dans une grande entreprise ou une administration qui développe une application en interne. Immanquablement, il y aura un moment où les représentants des utilisateurs vont se trouver face à face avec les développeurs. C'est l'assistant à la maîtrise d'ouvrage qui doit gérer le choc des cultures. Un talent d'animateur et de modérateur est requis, car il peut y avoir des tensions.

Quel que soit le type de prototype développé, il risque de dériver vers un début de développement. Cela peut être intentionnel (méthodes agiles), mais cela doit être contrôlé. Avant de faire développer un prototype, il faut estimer les risques que comporte cette « incursion » dans la phase de réalisation : spécifications informelles et rampantes, court-circuitage inévitable entre maîtrise d'ouvrage et maîtrise d'œuvre, et illusion, de la part de la maîtrise d'ouvrage, que le développement du produit est presque fini (alors qu'il n'a pas encore commencé).

## Matrices CRUD et RACI

---

Un des moyens de détecter les incohérences et incomplétudes est la matrice CRUD (de l'anglais *Create, Read, Update, Delete*). Une entité doit pouvoir être créée, lue, mise à jour et supprimée du système. Si l'une de ces opérations ne peut être effectuée par le système pour une entité donnée, il faut pousser l'analyse et comprendre pourquoi. Ce n'est pas nécessairement une incomplétude (par exemple, on peut vouloir que certaines entités ne soient jamais supprimées), mais il faut s'en assurer.

Une autre matrice que l'on peut utiliser est celle des responsabilités, ou matrice RACI (*Responsible, Accountable, Consulted, informed*) définissant le rôle des acteurs par rapport aux fonctions. Comme pour la matrice CRUD, il ne s'agit pas de détecter 100 % des incohérences, mais d'affiner l'analyse.

## Évaluer la faisabilité et le coût

---

Il est rare que le demandeur (maîtrise d'ouvrage ou utilisateur) connaisse le coût de ses exigences. D'une part, par ce que ce qui est simple à l'exécution peut être très complexe à implémenter sous forme de logiciel (l'inverse est également vrai : certains utilisateurs n'osent pas demander une fonction en pensant que leur demande est irréaliste, alors qu'elle est très simple à réaliser). D'autre part, par ce que le coût d'une fonction dépend du contexte, de ce qui a été réalisé par ailleurs.

Il est donc important de conseiller le demandeur sur le coût estimé de sa demande. Ce coût n'est pas seulement financier. La réalisation d'une exigence peut nécessiter des délais supplémentaires, une formation des utilisateurs, une nouvelle organisation, ou avoir des répercussions dans d'autres domaines.

Par faisabilité, on entend aussi l'estimation des risques. Une nouvelle fonction peut avoir des conséquences sur le délai de livraison de l'ensemble de l'application, ou sur ses performances, ou sa sécurité.

C'est au maître d'ouvrage de décider de l'opportunité de réaliser une exigence, après avoir été informé des conséquences. Le diagramme de Kano (figure 10-7) permet de chercher le rapport entre le degré de réalisation d'une caractéristique et le degré de satisfaction du client. Le principe est de ranger chaque caractéristique du produit (fonction ou caractéristique non fonctionnelle) dans une des trois catégories :

- **Fonction obligatoire** (indispensable) : son absence crée de l'insatisfaction chez le client, mais sa présence n'apporte pas de satisfaction particulière (souvent considérée comme évidente, donc passée sous silence lors d'une interview).
- **Fonction à satisfaction proportionnelle** : le niveau de satisfaction est proportionnel à la présence de la fonction ou à la performance de la caractéristique (généralement, elles émergent les premières lors d'une interview ou d'une enquête).
- **Fonction attractive** : son absence ne crée pas de frustration, mais sa présence apporte une satisfaction supplémentaire (rarement exprimées lors d'interviews, mais surtout avec des techniques de créativité).

Pour savoir dans quelle catégorie cette fonction doit être rangée (du point de vue de l'utilisateur, bien sûr) il faut lui demander comment il réagirait si ce besoin était comblé et s'il ne l'était pas, puis de lui proposer, pour chaque question, quatre réponses possibles : « j'aime », « c'est normal de l'avoir », « je suis indifférent » et « je n'aime pas ». On consigne ensuite les résultats dans une matrice (tableau) à quatre lignes et quatre colonnes, correspondant à ces quatre réponses aux deux questions. Ils permettent de déterminer, pour chaque besoin exprimé, s'il est indispensable, de base, ou constitue un « plus ».

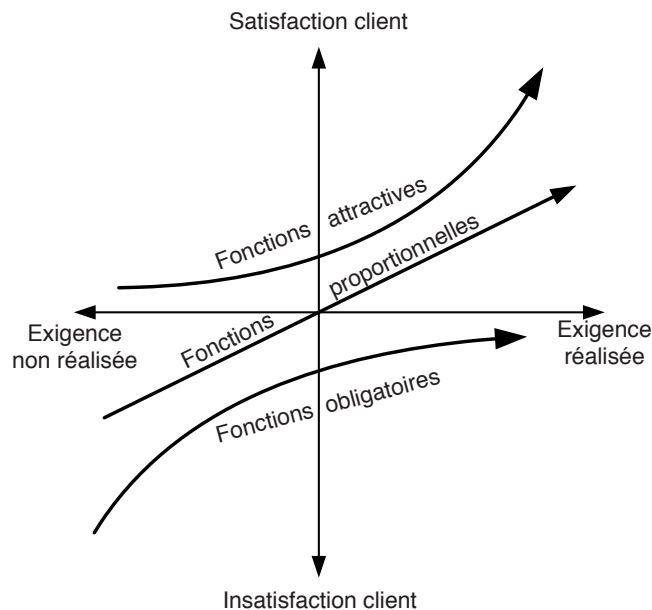


Figure 10-7 : Diagramme de Kano

### Check-list d'analyse

---

Il n'y a pas véritablement de « fin d'étape » d'analyse, il ne s'agit donc pas ici de vérifier qu'un objectif a été atteint. Cette check-list est à utiliser périodiquement pour vérifier que l'on avance dans la bonne direction et que les moyens nécessaires et suffisants sont mis en œuvre pour améliorer la cohérence, la complétude des exigences. Il s'agit aussi d'éviter le phénomène de l'analyse interminable (en anglais *analysis paralysis*). Cette liste peut donc donner l'impression qu'elle fait double emploi avec les check-lists du recueil et des spécifications.

- On a examiné, revu, affiné et si nécessaire modifié le diagramme de contexte.
- Le dictionnaire de données a été créé, et il est régulièrement enrichi.
- Les exigences ont été décomposées de manière suffisamment fine pour refléter les besoins les plus proches du terrain (utilisateurs).
- Les exigences ont été décomposées de manière suffisamment fine pour être compréhensibles sans ambiguïté par le maître d'œuvre.
- On n'a pas décomposé les exigences au-delà du strict nécessaire.
- Les différents acteurs ont activement participé à l'analyse.
- Chaque cas d'utilisation a un acteur défini.
- Chaque fonction a un utilisateur défini.
- Il y a une traçabilité entre les exigences de différents niveaux.
- On a défini des priorités entre exigences.
- On a utilisé des modèles graphiques lorsque cela était nécessaire.
- On a utilisé une matrice CRUD.
- On a utilisé une matrice RACI.



# Les exigences non fonctionnelles

Difficiles à recueillir, à analyser et à spécifier, liées à la fois au matériel, au logiciel et à l'usage qui en est fait, les exigences non fonctionnelles sont souvent négligées. Pourtant, c'est la satisfaction de ces exigences qui fait la qualité, et donc dans une large mesure, la valeur d'un logiciel.

### Les caractéristiques de qualité

Un logiciel ne peut être décrit ni par ses seules fonctions (point de vue du maître d'ouvrage), ni par ses seules caractéristiques techniques (contraintes de la maîtrise d'œuvre). Le cahier des charges doit nécessairement comporter des caractéristiques relatives à la qualité du logiciel, appelées *exigences non fonctionnelles*.

Comme les exigences fonctionnelles, celles-ci doivent être structurées selon leur typologie. Plusieurs typologies existent pour décrire la qualité du logiciel. La plus ancienne est celle de McCall, qui permet de caractériser la qualité du logiciel selon un certain nombre de facteurs et de critères.

Dans les paragraphes qui suivent, nous décrivons les exigences non fonctionnelles en utilisant la structuration par caractéristiques et sous-caractéristiques de qualité selon la norme ISO/CEI 25000. Cette norme a l'avantage de décrire la qualité du logiciel de la manière la plus complète et la moins redondante possible.

Dans la pratique, pour élaborer un cahier des charges, on ne suivra pas nécessairement ce même schéma. Chaque modèle de cahier des charges structure les exigences non fonctionnelles selon un schéma qui lui est propre. ISO 25000 pourra servir de trame et de liste de contrôle lors de l'élaboration d'un cahier des charges ou de l'utilisation d'un modèle de cahier des charges.

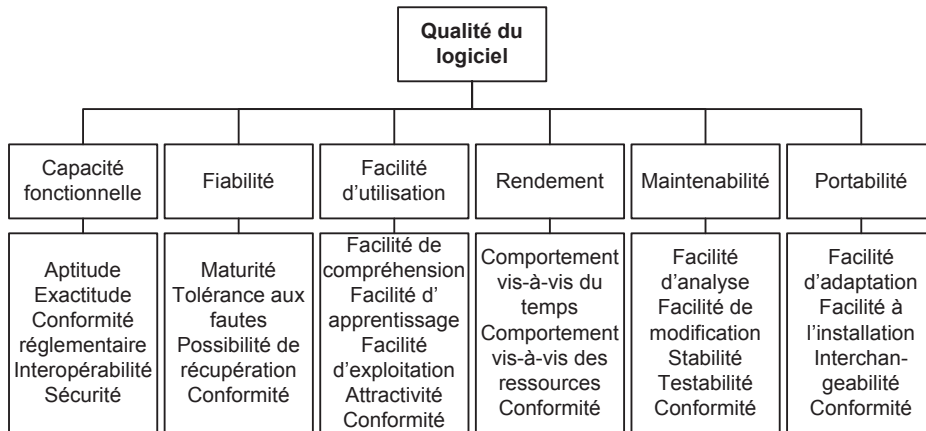


Figure 11-1 : Caractéristiques et sous-caractéristiques ISO 25000

## La norme ISO/CEI 25000

1. La série normative ISO/CEI 25000 résulte de la fusion entre plusieurs normes dont ISO/CEI 9126 et ISO/CEI 14598.

La norme ISO 25000<sup>1</sup>, relative à la qualité du produit logiciel, donne des outils pour, d'une part évaluer la qualité du logiciel, et d'autre part pour *construire* la qualité. Selon cette norme, la qualité du logiciel peut être perçue sur trois niveaux :

- La qualité interne (*internal quality*) correspond au niveau structurel d'analyse, et peut être évaluée par examen du code source, de l'architecture, des spécifications ; c'est la perspective de l'architecte de l'application, du concepteur, du développeur.
- La qualité externe (*external quality*) correspond au niveau comportemental, visible et mesurable en particulier lors de tests ; c'est la qualité vue avec la perspective du chef de projet et des équipes de validation.
- La qualité de fonctionnement (*quality in use*) correspond au point de vue de l'utilisateur. Elle se décompose en quatre caractéristiques : l'*efficacité* (relative à l'atteinte des objectifs de l'utilisateur), la *productivité* (économie de temps, d'efforts) la *sécurité* (risques sur l'utilisateur, le logiciel, l'environnement), et la *satisfaction* de l'utilisateur.

La norme se réfère également à la qualité du processus de développement, sans définir ce processus (il y a d'autres normes pour cela). Ces différents niveaux de qualité du produit et du processus sont interdépendants : la qualité de fonctionnement dépend de la qualité externe, qui dépend de la qualité interne, qui elle-même dépend de la qualité du processus de développement.

Cette représentation étagée de la qualité pourra être mise à profit pour structurer les exigences dans un cahier des charges<sup>2</sup>, par exemple, selon le schéma :

- qualité en utilisation → exigences utilisateur ;
- qualité externe → exigences produit ;
- qualité interne → contraintes techniques ;
- qualité du processus → contraintes projet.

Les contraintes liées au projet et les contraintes techniques seront abordées plus loin. Examinons ici les exigences de qualité interne et externe telles que définies par la norme, c'est-à-dire sous forme de caractéristiques et sous-caractéristiques.

---

2. Voir au chapitre 13 comment formuler les exigences non fonctionnelles, et au chapitre 14 leur place dans les différents modèles de cahiers des charges.

---

### Capacité fonctionnelle

La capacité fonctionnelle (ou fonctionnalité) est *l'ensemble d'attributs portant sur l'existence d'un ensemble de fonctions et leurs propriétés données*. Les propriétés sont celles qui satisfont aux besoins exprimés ou implicites (norme ISO/CEI 25000).

Elle se décompose en cinq sous-caractéristiques :

- Aptitude : attributs du logiciel portant sur la présence et l'adéquation d'une série de fonctions pour des tâches données.
- Exactitude : ensemble des attributs de logiciel portant sur la fourniture de résultats justes ou convenus.
- Conformité réglementaire : respect des normes, conventions et réglementations de droit.
- Interopérabilité : capacité à interagir avec des systèmes donnés.
- Sécurité : aptitude à empêcher tout accès non autorisé (accidentel ou délibéré) au programme ou aux données.

En pratique, dans un cahier des charges, la section exigences fonctionnelles décrira l'aptitude, et rien de plus. La conformité réglementaire sera définie dans la section règles de gestion. Les trois autres sous-caractéristiques (exactitude, conformité réglementaire, sécurité) seront décrites dans des paragraphes correspondants, dans la section exigences non fonctionnelles.

Pour spécifier les exigences d'interopérabilité et de sécurité, un moyen pratique et efficace consiste à faire référence à des normes relatives à ces sous-caractéristiques.

### Fiabilité

La fiabilité est *l'ensemble d'attributs portant sur l'aptitude du logiciel à maintenir son niveau de service dans des conditions précises et pendant une période déterminée*.

Ce critère se décline en :

- Maturité : fréquence de défaillances dues à des défauts du logiciel.
- Tolérance aux fautes : aptitude du logiciel à maintenir un niveau de service acceptable en cas de défaut du logiciel ou de violation de son interface.
- Possibilité de récupération : capacité à rétablir son niveau de service et de restaurer les informations directement affectées en cas de défaillance, et sur le temps et l'effort nécessaires pour le faire.

Parmi les mesures utilisées pour la fiabilité, citons le temps moyen entre deux défaillances (MTBF, *mean time between failures*), la durée moyenne de défaillance (*mean down time*), ou le temps mis par le système pour redémarrer. Nous sommes là à la limite entre les exigences produit (très dépendantes du matériel) et les exigences de service. Formuler ces exigences dans un cahier des charges est tout à fait possible, mais la formulation est très dépendante des fournitures attendues (logiciel seul, matériel et logiciel, services, etc.).

### Facilité d'utilisation (utilisabilité)

La notion d'utilisabilité est proche de celle d'ergonomie. L'ergonomie est l'étude des situations de travail et des relations entre l'être humain et un système. Elle a pour objectif d'améliorer le confort, voire le plaisir, de l'utilisateur et par voie de conséquence l'efficacité dans le travail et dans l'utilisation d'un produit (en l'occurrence, un logiciel). Le terme d'utilisabilité englobe un périmètre plus large, qui comprend l'efficacité. De plus, le mot ergonomie a été largement galvaudé par les informaticiens, qui le mettent un peu à toutes les sauces, et par les utilisateurs, qui ne sont pas suffisamment précis dans leurs exigences.

La norme ISO/CEI 25000 définit la facilité d'utilisation (ou utilisabilité) comme « l'ensemble d'attributs portant sur l'effort nécessaire pour l'utilisation et sur l'évaluation individuelle de cette utilisation par un ensemble défini ou implicite d'utilisateurs ». Elle se décline en :

- Facilité de compréhension : attributs du logiciel portant sur l'effort que doit faire l'utilisateur pour reconnaître la logique et sa mise en œuvre.
- Facilité d'apprentissage : attributs du logiciel portant sur l'effort que doit faire l'utilisateur pour apprendre son application (par exemple, maîtrise de l'exploitation des entrées, des sorties).
- Facilité d'exploitation : attributs du logiciel portant sur l'effort que doit faire l'utilisateur pour exploiter et contrôler son application.
- Attractivité : capacité du produit logiciel à être attrayant pour l'utilisateur.

L'utilisabilité est difficile à définir. Une façon de faire consiste à exiger la conformité à des standards d'utilisabilité ou à une charte d'ergonomie. De tels documents imposent des règles, comme par exemple sur le nombre maximal de boutons par écran, le nombre d'objets sur une liste ou les combinaisons de couleurs autorisées.

Une autre manière de faire consiste à formuler les exigences d'utilisabilité à un plus haut niveau, en l'occurrence au niveau de l'utilisateur. Voici par exemple comment peuvent s'exprimer quatre exigences d'utilisabilité, correspondant aux quatre sous-caractéristiques (facilité d'apprentissage, de compréhension, d'exploitation et d'attractivité) :

Pour 80 % des médecins, une formation initiale de 2 heures sera suffisante à l'appropriation des fonctions de prescription.

Après un mois d'utilisation quotidienne, 80 % des utilisateurs formés à l'outil pourront l'utiliser sans faire appel à une aide extérieure.

Le temps moyen mis par un médecin formé au logiciel pour prescrire un médicament avec le module de prescription ne doit pas excéder de 10 % le temps moyen mis par un médecin pour effectuer cette action manuellement.

Après un mois d'utilisation quotidienne, 80 % des utilisateurs devront se déclarer satisfaits ou très satisfaits par le système.

Remarquons cette proportion de 80 %. On ne peut pas exiger qu'une fonction faisant intervenir des êtres humains soit réalisée à 100 %. Quatre utilisateurs sur cinq efficaces et satisfaits est beaucoup plus réaliste.

### Rendement

La norme ISO/CEI 25000 définit le rendement comme *l'ensemble d'attributs portant sur le niveau de service d'un logiciel et la quantité de ressources utilisées, dans des conditions déterminées*. Celui-ci se décline en :

- comportement vis-à-vis du temps, portant sur les temps de réponse et de traitement, ainsi que sur les débits lors de l'exécution de sa fonction ;
- comportement vis-à-vis des ressources, portant sur la quantité de ressources utilisées et sur la durée de leur utilisation lorsqu'il exécute sa fonction.

Dans un cahier des charges, il est rare que l'on spécifie directement le comportement vis-à-vis des ressources (mémoire ou espace disque). En général, il est plus utile de spécifier l'exigence à un plus haut niveau (nombre d'utilisateurs simultanés). Le comportement vis-à-vis du temps sera également décrit comme une exigence non fonctionnelle de niveau

utilisateur (temps de réponse). Ces deux exigences sont interdépendantes. Cela donne :

Le système pourra être utilisé simultanément par 150 utilisateurs.

À pleine charge, le temps d'affichage d'un écran complet sera inférieur ou égal à 1,5 seconde.

Remarquons que cette la première exigence de rendement est la déclinaison d'une contrainte organisationnelle, voire d'un objectif stratégique. La seconde exigence dérive d'une exigence d'utilisabilité, et correspond à un seuil psychologique. Ces deux exigences sont interdépendantes : avec 150 utilisateurs simultanés, le système répondra en moins d'une seconde et demie.

### Maintenabilité

La maintenabilité est formellement définie « comme l'ensemble d'attributs portant sur l'effort nécessaire pour faire des modifications données » (ISO/IEC 25000). Elle se décompose en :

- Facilité d'analyse : attributs du logiciel portant sur l'effort nécessaire pour diagnostiquer les déficiences ou les causes de défaillance, ou pour identifier les parties à modifier.
- Facilité de modification : attributs du logiciel portant sur l'effort nécessaire pour modifier, remédier aux défauts, ou changer d'environnement.
- Stabilité : attributs du logiciel portant sur le risque des effets inattendus des modifications.
- Testabilité : attributs du logiciel portant sur l'effort nécessaire pour valider le logiciel modifié.

La maintenabilité est une caractéristique essentiellement interne, invisible de l'utilisateur. On peut cependant spécifier des contraintes de service à l'utilisateur. Le modèle Volere (voir chapitre 14) regroupe dans un même paragraphe les exigences de maintenabilité et de support. Il s'agit en réalité d'exigences de maintenance plutôt que de maintenabilité.

### Portabilité

La portabilité est « l'ensemble d'attributs portant sur l'aptitude du logiciel à être transféré d'un environnement à l'autre » (norme ISO/CEI 25000). Elle se décompose en :

- Facilité d'adaptation : elle concerne les attributs du logiciel portant sur la possibilité de son adaptation à différents environnements donnés sans avoir recours à d'autres actions ou moyens que ceux prévus à cet effet pour le logiciel considéré.

- Facilité à l'installation : effort nécessaire pour installer le logiciel dans un environnement donné.
- Conformité relative aux règles de portabilité : attributs du logiciel permettant à celui-ci de se conformer aux normes ou conventions ayant trait à la portabilité.
- Interchangeabilité : aptitude du logiciel à être utilisé à la place d'un autre logiciel dans le même environnement.

La portabilité stricto sensu (facilité d'adaptation) induit une contrainte très forte sur le développement et l'architecture du logiciel. Aussi, on ne l'exigera que si cela s'avère nécessaire, après une éventuelle étude d'opportunité. En revanche, la facilité d'installation est une exigence à forte valeur ajoutée, surtout si le logiciel doit être installé sur des postes individuels. Il faudra la spécifier dans tous les cas.

### Zoom sur l'utilisabilité

---

Il y a au moins trois manières de spécifier l'utilisabilité dans un cahier des charges. La première s'appuie sur les quatre sous-caractéristiques d'utilisabilité de la norme ISO 25000 déjà mentionnée : facilité d'apprentissage, facilité de compréhension, facilité d'exploitation (utilisation au quotidien) et attractivité.

La deuxième façon de structurer les exigences d'utilisabilité s'appuie également sur ISO 25000, cette fois sur la *qualité de fonctionnement* (en anglais, *quality in use*). Celle-ci se définit par quatre critères :

- *L'efficacité* : capacité du logiciel à permettre aux utilisateurs d'atteindre leurs objectifs avec exactitude et exhaustivité.
- *La productivité* : capacité à optimiser le temps et l'effort de l'utilisateur.
- *La sécurité* : capacité du produit logiciel à limiter les risques sur la santé, les activités ou l'environnement des utilisateurs, en particulier suite à une défaillance (ici, le terme *innocuité*, traduction de l'anglais *safety*, serait sans doute plus approprié).
- *La satisfaction* : correspond à une attitude positive de l'utilisateur lors de l'interaction avec le produit.

La troisième manière de structurer les exigences s'appuie sur les critères d'ergonomie définis par la norme AFNOR Z-67-133-1<sup>3</sup>, basée sur les travaux de Bastien et Scapin. Cette norme définit l'ergonomie du logiciel par sept critères (figure 11-2) :

- la compatibilité ;
- le guidage ;
- l'homogénéité ;

---

3. AFNOR Z67-133-1. Définition des critères ergonomiques de conception et évaluation des interfaces utilisateurs, 1991.

---

- la souplesse ;
- le contrôle explicite ;
- la gestion des erreurs ;
- la concision.

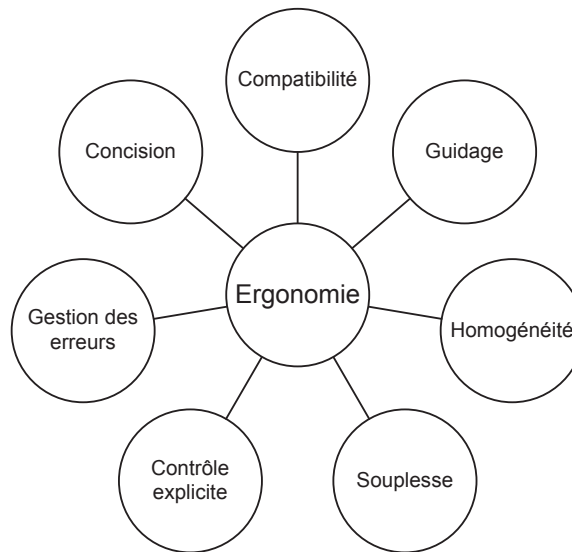


Figure 11-2 : Sept critères d'ergonomie

Examinons chacun de ces critères et la manière de les décliner sous forme d'exigences.

### Compatibilité

La compatibilité est la « capacité à s'intégrer dans l'activité des utilisateurs ». Elle concerne l'accord pouvant exister entre les caractéristiques professionnelles et psychologiques de l'utilisateur (mémoire, perceptions, habitudes, etc.) et l'organisation du dialogue entre l'utilisateur et l'application. Une bonne compatibilité réduit considérablement le temps d'apprentissage d'une application.

On doit exiger que le logiciel s'adapte au métier de chaque profil utilisateur. Le système doit non seulement intégrer les procédures, qui relèvent plutôt d'exigences fonctionnelles, mais s'adapter à la psychologie et la physiologie de l'utilisateur. Un exemple typique est celui de l'utilisation de la souris, exclue pour les utilisateurs qui saisissent des données en masse, mais nécessaire pour les décideurs sur une application intranet. La segmentation des profils utilisateurs et l'analyse de leurs tâches sont donc nécessaires avant toute spécification.



### Guidage

Le guidage est « l'ensemble des moyens mis en œuvre pour conseiller, orienter, informer et conduire l'utilisateur lors de ses interactions avec l'ordinateur ». On distingue le guidage explicite, qui se traduit par l'aide en ligne, les messages d'erreur, d'avertissement ou d'information, et le guidage implicite, qui concerne les formats de saisie des informations, la disposition des informations sur l'écran, etc.

Le guidage est lié à la navigabilité. L'utilisateur doit savoir à tout moment à quel point du processus il se trouve, et comment passer à une autre tâche. Le système doit canaliser le travail de l'utilisateur en lui laissant un maximum de liberté tout en l'empêchant d'entreprendre des actions qui nuisent à sa sécurité, à sa productivité ou à son efficacité.

### Homogénéité

L'homogénéité est la « capacité d'un système à conserver une logique d'usage constante ». Elle se réfère à la façon avec laquelle des choix d'objets de l'interface sont conservés pour des contextes identiques, et des objets différents pour des contextes différents. Elle facilite l'apprentissage et réduit l'effort de mémorisation.

L'homogénéité concerne la localisation d'un objet sur l'écran, son format et sa présentation, ainsi que la syntaxe et le vocabulaire employés. Les écrans de saisie et la logique des actions doivent être analogues dans toute l'application, et même, si possible, d'une application à l'autre.

Un bon moyen d'exiger l'homogénéité consiste à imposer une charte d'ergonomie, préexistante ou adaptée au logiciel. Cependant, des exigences particulières à certaines tâches doivent être élaborées.

### Souplesse

La souplesse est la « capacité de l'interface à s'adapter aux différentes exigences de la tâche, aux diverses stratégies, aux habitudes et niveaux de connaissances des différents utilisateurs ».

Contrairement à la compatibilité, qui revient à adapter d'avance le logiciel aux différents profils utilisateurs, la souplesse permet à un logiciel de s'adapter à ses habitudes, ainsi qu'à son expérience ou son niveau d'expertise.

Exemple de souplesse : les modes expert et débutant. Les utilisateurs débutants auront les commandes présentées sous forme de menu, les expérimentés auront des raccourcis clavier à leur disposition. Autre exemple, la facilité pour l'utilisateur à paramétrer le logiciel pour l'adapter à son métier... tout en respectant l'exigence d'homogénéité, ce qui montre à quel point l'ergonomie n'est pas chose aisée.

### Contrôle explicite

Le contrôle explicite est « l'ensemble des moyens du dialogue qui permettent à l'utilisateur de maîtriser le lancement et le déroulement des opérations exécutées par le système informatique ». C'est ce qui permet à l'utilisateur de maîtriser le système. Ce contrôle se manifeste en particulier par le feedback que fournit le système. En pratique, les réactions du système doivent être prévisibles.

Cette exigence apporte à l'utilisateur l'autonomie qu'il est en droit d'attendre du système, ainsi qu'un confort lié à une sensation de contrôle sur l'application. Cette agréable sensation de « piloter » le produit (plutôt que d'être malmené par lui) facilite l'appropriation de l'outil par l'utilisateur et minimise les erreurs.

### Gestion des erreurs

La gestion des erreurs rassemble « tous les moyens permettant d'une part d'éviter ou de réduire les erreurs, et d'autre part de les corriger lorsqu'elles surviennent ». Elle prévient le stress et augmente la sensation de sécurité (et la sécurité effective). Comme exemples d'exigences, citons :

- la possibilité d'annuler une action (touche « undo ») ;
- la demande de confirmation des actions dont les effets sont irréversibles ;
- les messages d'avertissement.

### Concision

La concision est « l'ensemble des moyens qui, pour l'utilisateur, contribuent à la réduction de ses activités de perception et de mémorisation et concourent à l'augmentation de l'efficacité du dialogue ». Elle a des effets positifs sur la productivité et contribue à diminuer la charge mentale et donc la fatigue et le stress.

Exemple d'exigence :

- minimiser le nombre de menus et de clics pour exécuter une tâche ;
- limiter le nombre de possibilités offertes (tout en respectant les exigences de souplesse et de contrôle explicite).

# Exigences projet et contraintes techniques

Les exigences projet et les contraintes techniques ne sont ni des exigences fonctionnelles, ni des exigences non fonctionnelles. Elles ne concernent pas le produit, mais sa mise en œuvre. Elles occupent un chapitre à part dans le cahier des charges. La difficulté consiste à les décrire sans tomber dans le piège classique de la description d'une solution en lieu et place d'un besoin.

### Contraintes de projet

---

La formulation de ces contraintes dépend beaucoup du type de solution (progiciel, développement spécifique) et du rapport entre client et fournisseur (développement par des équipes internes, ou au contraire appel d'offres public).

### Charges, coûts et délais

Ces exigences imposées à la maîtrise d'œuvre découlent des contraintes de budget et d'urgence au niveau de la maîtrise d'ouvrage. Il faut savoir qu'une diminution du délai a toujours une répercussion importante sur les coûts, voire même sur la faisabilité, (« neuf femmes ne font pas un enfant en un mois » dit-on parfois), et souvent sur la qualité.

L'expression de contraintes de coût peut être utile dans le cadre d'un développement en interne, elle l'est beaucoup moins lorsqu'on s'adresse à un fournisseur externe, et en principe interdite dans le cadre d'un appel d'offres public.

### Installation, mise en exploitation, recette

Les exigences doivent indiquer (ou demander au fournisseur d'indiquer) si le produit est autoinstallable, si l'installation est à la charge du fournisseur

ou du client, la charge d'installation pour le fournisseur et/ou le client, les délais à prévoir, et la disponibilité des différents acteurs par profil.

Le cahier des charges peut détailler les modalités de recette ou demander à la maîtrise d'ouvrage de les proposer.

### Migration

Comme pour l'intégration, la migration peut être plus ou moins complexe et coûteuse, jusqu'à nécessiter une véritable étude. Pour que le fournisseur puisse répondre à cette contrainte, il faut lui indiquer aussi précisément que possible, au minimum le format et le contenu des données échangées, le support physique et/ou logique (par exemple, base de données, disque dur), la structure (fichiers plats, tables de base de données relationnelle...) et le volume de données à reprendre.

Le cahier des charges doit indiquer clairement les engagements mutuels du fournisseur et du client, qui sont très interdépendants. Par exemple, on peut exiger du fournisseur qu'il reprenne les données présentées sous forme de fichier à plat, le client s'engageant à fournir ce fichier après avoir « attaqué » lui-même la base de données de l'ancienne application. Les charges, délais, disponibilités des acteurs sont également des exigences à indiquer.

### Environnement de développement

Imposer l'environnement de développement, en termes d'outils, langages, méthodes, etc., est parfois nécessaire, mais induit des contraintes très fortes sur le maître d'œuvre. Il faudra donc bien peser le pour et le contre avant de formuler cette exigence, éventuellement après discussion ou négociation, ou laisser dans le cahier des charges l'ouverture à la négociation.

## Contraintes d'environnement

---

### Environnement matériel et logiciel

Ces exigences ou contraintes concernent la plate-forme matérielle, le système d'exploitation, les bases de données, les navigateurs sur (ou sous) lesquels la solution devra fonctionner.

La première contrainte concerne le type de fourniture. Trois cas peuvent se présenter :

- Seul le produit logiciel est livré. Le client dispose de la plate-forme. Dans ce cas, c'est au client de décrire l'environnement, qui devient de facto une contrainte. Le fournisseur devra préciser dans sa réponse ses propres contraintes en termes de matériels et logiciels de base.

- Le client demande la fourniture du produit logiciel seul, mais n'a pas encore acquis l'infrastructure nécessaire pour l'héberger. L'exigence consiste à demander au fournisseur de spécifier ses propres exigences en termes d'infrastructure. Le client peut également exiger du fournisseur qu'il maintienne son produit pour tenir compte de l'évolution du matériel ou du logiciel de base.
- Le client demande une solution complète, matériel et logiciel. Il exige du fournisseur de détailler la liste des matériels et logiciels fournis, et de lister ses propres contraintes d'environnement (salle, etc.).

Toutes les nuances et variations sont possibles. Les autres contraintes d'environnement matériel et logiciel dépendent étroitement de cette première contrainte.

### Interfaces

Ce paragraphe détaille les protocoles, formats d'échange de données, appels de procédures entre applications, messages échangés. Si l'on descend dans des détails techniques, ce qui est souvent nécessaire, il faudra faire appel à des experts. En fonction du niveau de détail, il sera utile d'explicitier les exigences sous forme de diagrammes de séquence ou de diagrammes d'état.

Chaque fois que cela sera possible, on s'appuiera sur des normes et standards d'interopérabilité existants. La référence à des standards est infiniment plus sûre et économique en efforts de spécification que la description détaillée d'interfaces. Elle a des conséquences très positives sur le coût et la qualité de la solution.

### Services d'accompagnement

---

Les services d'accompagnement doivent être spécifiés avec beaucoup de soin, clarté et précision, car c'est souvent sur ces services que le fournisseur « fait sa marge ». Une spécification imprécise peut donner lieu à des litiges, voire des conflits entre client et fournisseur, avec des risques financiers.

Il y a deux manières de formuler les exigences de service : imposer des contraintes ou demander au fournisseur de préciser sa réponse (proposition technique et commerciale).

Lorsque le fournisseur est une société externe et que le client doit passer par un appel d'offres public (État, collectivité), c'est souvent la deuxième option qui est choisie, voire nécessaire.

### Exigences d'intégration

L'intégration du produit à l'environnement du client est un point très délicat. Pour que le fournisseur puisse répondre aux exigences d'intégration, il doit bien connaître l'environnement du client, en particulier les applications « partenaires » avec lesquelles le produit fourni devra communiquer. Souvent, une véritable étude d'intégrabilité doit avoir lieu. Elle peut être faite par le client, le fournisseur, et le plus souvent par une collaboration des deux parties. L'exercice est d'autant plus difficile que, lorsque le client établit son cahier des charges, il ne connaît pas encore la solution proposée par le fournisseur. C'est la situation dans laquelle se trouve, en particulier, l'Administration lorsqu'elle établit un appel d'offres public pour choisir un progiciel du marché.

### Support utilisateurs, support technique

La qualité du support aux utilisateurs est souvent une des clés de la réussite d'un projet. De plus, le support est un gros consommateur de ressources. Il est donc indispensable de spécifier précisément les conditions de support ou de demander au fournisseur de les spécifier.

Les différents niveaux de support doivent être décrits, ainsi que le profil des intervenants à chaque niveau de support, et les ressources dédiées. Par exemple, le support premier niveau pourra être fait par le client, et le deuxième niveau par le fournisseur.

### Maintenance

Les exigences de maintenance ne doivent pas être confondues avec les exigences de maintenabilité. La maintenabilité est une propriété du logiciel, alors que la maintenance est un service rendu au client.

En dehors de cas très rares, un logiciel installé devra être maintenu. Le client ne pourra pas se passer de ce service, qui lui sera facturé, parfois très cher. De plus en plus fréquemment, la maintenance coûte plus cher que l'achat du produit (cette tendance s'accélère avec l'arrivée de produits *open source*). De plus, les opérations de maintenance peuvent elles-mêmes induire des contraintes pour le client et l'utilisateur, avec des arrêts d'exploitation pour montée de version ou pour correction de bogues.

Un client a donc intérêt à bien spécifier les conditions et les modalités de la maintenance. Les exigences dépendent bien sûr du domaine d'application, du type d'application, de l'environnement informatique et métier. Voici quelques exemples d'exigences de maintenance :

- fréquence des installations de versions majeures ;
- modalités d'installation (par le client, par le fournisseur, etc.) ;

- processus de maintenance, tout particulièrement pour la maintenance corrective ;
- obligations respectives du client et du fournisseur concernant la maintenance de l'infrastructure sous-jacente (serveurs, base de données, etc.) ;
- compatibilité ascendante entre versions (peut aussi être considérée comme une exigence de maintenabilité).

### Documentation

Rappelons que la documentation fait partie intégrante du produit. Les contraintes générales s'appliquant au produit s'appliquent donc ipso facto à sa documentation. C'est en particulier le cas de la livraison d'une documentation suite à un changement de version du produit. Cela va sans dire, mais cela va mieux en le disant, et encore mieux en le stipulant par écrit.

Les exigences de documentation peuvent être extrêmement variées. Elles peuvent concerner le volume de la documentation, sa forme physique (papier, média, documentation en ligne...), la fréquence de sa mise à jour, son contenu, la facilité de lecture et d'apprentissage ou son mode de distribution.

### Formation des utilisateurs

La formation des utilisateurs est très importante pour la réussite du projet. La plupart des fournisseurs ont, parallèlement à leur offre produit, des offres de formation standard. Le prix de ces formations peut être très élevé.

Il est tout à fait possible, même dans le cadre d'appels d'offres publics, de « négocier » ces prix au moyen du cahier des charges. En particulier, on sait que certains utilisateurs n'utilisent qu'un très petit nombre de fonctions. La formation standard « généraliste » peut être « tronçonnée » en fonction du profil utilisateur. Rien n'empêche de stipuler que telle catégorie d'utilisateurs doit pouvoir être formée en deux heures (en lieu et place d'une formation générale de trois jours, par exemple). En fonction de la manière dont elle est formulée, cette exigence peut aussi trouver sa place dans la rubrique « facilité d'utilisation » (chapitre des exigences non fonctionnelles).

### Environnement physique d'utilisation

Il ne s'agit pas ici d'environnement informatique, mais bien de l'environnement physique au sens large. Par exemple, une solution (matériel et logiciel) peut nécessiter d'être utilisée en milieu tropical, stérile (bloc opératoire), humide, bruyant, etc. Décrire cet environnement est important, car il induit des contraintes sur le matériel informatique, mais aussi

sur l'ergonomie du produit. En indiquant, par exemple, que le logiciel sera utilisé au service des urgences d'un hôpital, on donne une information très importante au fournisseur, en lui laissant la liberté (et la responsabilité) d'imaginer la solution qui convient le mieux (affichage très sobre) ou de faire une étude d'ergonomie.

Le fait de décrire l'environnement d'utilisation et d'exiger du fournisseur que son produit respecte cette contrainte d'environnement reporte l'effort de recherche et d'étude (en particulier d'ergonomie) sur le fournisseur. Dans le cas d'un logiciel critique, cela risque d'être plus coûteux pour le fournisseur, mais aussi plus stimulant. On devra donc étudier l'opportunité de formuler les exigences, soit sous forme de contraintes physiques d'utilisation, soit sous forme de règles d'ergonomie, ou même d'exigences fonctionnelles. L'exercice n'est pas facile.



# L'étape de spécification

### Le processus de spécification

Le schéma ci-dessous présente une vision simplifiée du travail de spécification.

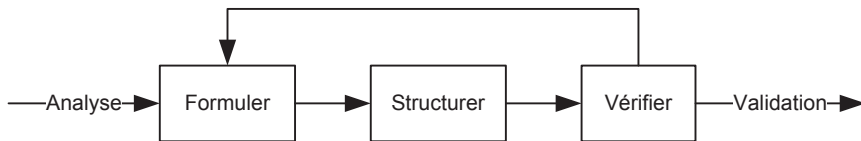


Figure 13-1 : Le processus de spécification

Les trois tâches consistant à formuler, structurer et vérifier les exigences sont très interdépendantes et le processus est naturellement itératif.

### Formuler

Le comportement (exigences fonctionnelles) et les propriétés (exigences non fonctionnelles) doivent être formulés avec beaucoup de soin et de précision, de manière à être compris par tous ceux qui liront le cahier des charges. Une bonne maîtrise de la langue écrite est nécessaire, mais non suffisante. Une bonne formulation sera facilitée si les étapes précédentes ont été respectées.

La spécification graphique est un cas particulier de formulation. Un graphique peut faciliter la compréhension, mais si l'on veut qu'il soit compréhensible de tous les acteurs, il est encore plus difficile à « écrire » qu'un texte.

### Structurer

Structurer le cahier des charges consiste à ranger les exigences sous forme arborescente, afin de rendre la lecture, la compréhension et la maintenance du document la plus aisée possible. Le modèle de cahier des charges nous donne un premier niveau de structuration, mais certains sous-chapitres devront être subdivisés, parfois très finement. C'est en particulier le cas des exigences fonctionnelles.

Pour un logiciel d'une grande richesse fonctionnelle, la structuration est loin d'être évidente. Dans un tel cas, on présente en général les exigences de deux niveaux différents dans deux chapitres différents :

- exigences utilisateur (par exemple, sous forme de cas d'utilisation) ;
- exigences produit (sous forme d'exigences élémentaires).

### Vérifier

Il s'agit ici de vérifier, en particulier au moyen de check-lists, que les exigences qui viennent d'être rédigées répondent à un certain nombre de critères. Elles doivent, en particulier, être alignées sur les objectifs et les exigences de plus haut niveau, correspondre au besoin des utilisateurs, être réalistes et bien formulées.

Les activités de structuration, de formulation et de vérification sont itératives.

L'activité de vérification fait partie intégrante de la spécification et est distincte de l'étape ultérieure de validation, qui s'intéresse autant au contenu qu'à la structure ou à la formulation.

## La formulation

---

Appliquer des règles de bonne formulation peut apparaître contraignant, mais est très avantageux à terme. Un cahier des charges bien formulé est beaucoup plus maintenable. N'oublions pas que sa maintenance commence très tôt, toute exigence étant susceptible d'être modifiée à tout moment (mais pas n'importe comment).

Nous donnons dans les paragraphes qui suivent un ensemble de principes et de règles de bonne formulation. La liste peut paraître longue, mais avec l'expérience, la formulation correcte devient une seconde nature.

### La structure grammaticale

Une exigence élémentaire correctement formulée répond aux critères suivants :

- Elle est grammaticalement correcte.

- Elle est rédigée à la forme active : sujet, verbe, complément.
- Le sujet est nécessairement un utilisateur, un système ou un attribut du système.
- Un même terme a la même signification pour tout lecteur potentiel.
- Les termes ambigus ont été définis dans un glossaire.

Il est nécessaire d'utiliser des phrases du type « le logiciel doit... ». Par exemple : « Le logiciel doit afficher le coût des médicaments prescrits » plutôt que : « Affichage du coût ».

Pour exprimer une exigence, il est préférable d'utiliser une phrase courte et claire. On évitera les phrases comportant des propositions relatives, ou des conjonctions de coordination. L'idéal (pas toujours réalisable) est que chaque exigence tienne sur une seule phrase et que chaque phrase exprime une seule exigence.

### Check-list de bonne formulation

La liste suivante servira de check-list à appliquer à chaque exigence. Une exigence bien formulée doit être :

- **Élémentaire** (en anglais, *atomic*). Elle ne doit comporter qu'un élément insécable. Pour vérifier si une exigence est élémentaire, essayez de la couper en deux ( en général au niveau des conjonctions de coordination).
- **Nécessaire**. L'exigence doit servir au moins à un profil utilisateur ou une partie prenante. Un moyen de vérifier cette règle est de se poser la question : que se passera-t-il si cette exigence est supprimée ?
- **Mesurable, ou du moins vérifiable**. Dans le cas contraire, le maître d'œuvre peut faire ce qu'il veut, c'est un vœu pieux. Les adjectifs et adverbes non mesurables (tels que rapide, performant, fiable, nombreux) rendent l'exigence non mesurable. Il en est de même pour les phrases à la voix passive, l'emploi du « etc. » et des points de suspension.
- **Concis**. Plus la formulation est courte, plus l'exigence sera robuste. Méthode : éliminer systématiquement tous les mots qui ne servent à rien.
- **Traçable**, vis-à-vis d'exigences de plus haut niveau ou de plus bas niveau. Au-delà d'un certain volume de spécifications, l'utilisation d'outils spécifiques est nécessaire pour que ce critère soit respecté.
- **Réalisable et modifiable**. Dans le cas contraire, c'est un vœu pieux. C'est ici que la compétence technique de l'analyste peut être utile. Il peut avertir son client de la difficulté ou de l'impossibilité de mettre en œuvre une exigence, et aider à la négociation.
- **Autosuffisante**. Dans la mesure du possible, elle doit être lue et comprise indépendamment des autres. Dans une exigence, un pronom per-

sonnel comme « il » ou « elle » se rapporte généralement à une exigence précédente. Il faut remplacer le pronom personnel par le sujet, quitte à se répéter (un cahier des charges n'est pas une œuvre littéraire).

- **Indépendante de la solution.** Rappelons qu'une exigence doit décrire un besoin ou une contrainte, et non une solution. De plus, certaines règles régissent les relations entre exigences. Un ensemble d'exigences doit être :
  - **Complet.** L'ensemble doit décrire tous les cas possibles. Par exemple, si on décrit le comportement du système en cas d'acceptation, on doit aussi décrire son comportement en cas de refus.
  - **Cohérent.** Une exigence ne doit pas entrer en conflit avec les autres. Une forme plus subtile d'incohérence advient lorsqu'un ensemble d'exigences entre en contradiction avec un autre ensemble.
  - **Non redondant.** Deux exigences ne doivent pas se recouvrir ou se répéter, même partiellement.

### La « règle des 5 C »

La règle dite « des 5 C » est une check-list condensée, un moyen mnémotechnique simple qui permet de vérifier rapidement qu'une exigence est bien formulée. Cette règle s'applique également au document d'exigences dans sa totalité.

Toute exigence doit être :

- **Correcte** : elle respecte les règles de la grammaire, les lois, les règlements, les bonnes pratiques de la profession.
- **Complète** : elle définit l'acteur, décrit l'action, et précise si nécessaire les conditions de l'action.
- **Claire** : elle ne comporte pas de flou, pas d'ambiguïté, pas de termes à sens multiple ; tout lecteur la comprend d'emblée, sans explication supplémentaire.
- **Concise** : elle est formulée avec le moins de mots possibles.
- **Cohérente** : elle n'entre pas en conflit avec d'autres exigences.

## La langue de bois : quand et comment l'éliminer ?

Qui n'a jamais usé de la langue de bois ? Quant aux sous-entendus, il est impossible de s'en passer dans la vie courante, ne serait-ce que pour la brièveté du discours. Le flou est généralement largement compensé par la communication non verbale.

Par ailleurs, le langage flou a sa place lors de l'étape de recueil. Nous laissons du flou dans nos questions pour permettre à l'interlocuteur de s'exprimer. C'est d'ailleurs une technique d'interview souvent utilisée, consciemment ou non.

Lors de la spécification, la langue de bois, et de manière générale le langage flou, est à proscrire. L'analyste l'éliminera sans pitié, soit en faisant

la correction lui-même (à valider plus tard), soit en demandant des précisions complémentaires. Le flou peut arriver :

Pendant le recueil des besoins (interview) : c'est normal, la précision doit venir progressivement. Reformuler en douceur.

Pendant l'analyse : c'est le moment de chercher à l'éliminer. Un bon schéma peut mettre au jour une incohérence du discours.

Lors des spécifications (rédaction du cahier des charges) : la tâche sera plus difficile ; il faudra chercher à préciser l'exigence, la corriger, et vérifier la cohérence d'ensemble.

Lors de la validation (revue de spécifications) : c'est un indicateur de non-qualité lors des étapes précédentes. Prévoir une charge de travail supplémentaire, et chercher à s'améliorer.

### Améliorer la formulation : exemples

Voici quelques exigences mal formulées, suivies d'une explication et de la reformulation améliorée. Elles proviennent d'un cahier des charges, en cours d'élaboration, pour un progiciel de gestion du circuit du médicament dans un établissement de santé.

#### **Exemple 1 : à la fois trop et pas assez**

Voici comment une exigence était initialement formulée dans un cahier des charges pour l'informatisation du circuit du médicament d'un hôpital :

Les stocks seront gérés au niveau du code produit, du produit et du numéro de lot, dans tous les lieux de stockage : la pharmacie et l'armoire à pharmacie des unités de soins.

Un analyste expérimenté détecte tout de suite qu'il ne s'agit pas d'une exigence. N'importe quel lecteur muni de la check-list de formulation correcte s'en apercevra également en essayant de faire passer la phrase de la voix passive à la voix active : on ne sait pas qui est l'acteur et que fait le système.

D'autre part, le texte apporte une surabondance d'informations concernant les lieux de stockage. Ils doivent être décrits, soit en intention : *quel que soit le lieu de stockage*, soit en extension : à la pharmacie et dans les unités de soins.

L'exigence élémentaire devient alors :

Le système doit permettre au pharmacien de suivre les stocks d'un médicament donné, par son code produit et par son numéro de lot.

Le système doit permettre le suivi de médicaments en pharmacie.

Le système doit permettre le suivi des médicaments en unités de soins.

On obtient donc, non pas une, mais trois exigences élémentaires. Cela peut sembler lourd, mais c'est beaucoup plus clair.

### **Exemple 2 : améliorer la clarté d'une exigence**

Voici une exigence extraite d'un cahier des charges pour un logiciel hospitalier :

Le progiciel doit comporter un préparamétrage de ces données (au niveau de l'hôpital ou d'une spécialité), préparamétrage modifiable par le prescripteur.

On comprend de quoi il s'agit, mais l'exigence peut être grandement améliorée. L'expression « comporter un paramétrage... » ne précise pas l'acteur. L'expression « au niveau de... » est ambiguë : on ne sait pas qui, de l'hôpital ou de la spécialité, est l'acteur de ce paramétrage, ou le bénéficiaire, ou les deux. « Ces données » fait allusion à une exigence précédente, spécifiant les informations qu'un prescripteur peut saisir. L'exigence améliorée devient :

Le progiciel doit permettre à l'administrateur du progiciel de paramétrer, pour l'ensemble de l'hôpital, les informations qu'un prescripteur peut saisir, et le caractère obligatoire de cette saisie.

Le progiciel doit permettre à l'administrateur du progiciel de paramétrer, pour chaque spécialité, les informations qu'un prescripteur peut saisir, et le caractère obligatoire de cette saisie.

Le progiciel doit permettre à un prescripteur de paramétrer, pour toutes les prescriptions à venir, les informations qu'il pourra saisir.

Cette formulation en trois exigences élémentaires est évidemment plus lourde, mais elle ne laisse plus aucune ambiguïté quant aux actions autorisées pour chaque acteur.

### **Exemple 3 : beaucoup de mots pour rien**

L'énoncé de l'exigence est le suivant :

Le progiciel doit pouvoir supporter les différentes organisations existantes dans l'établissement au niveau des unités de soin.

Le verbe « supporter », en l'occurrence un anglicisme, est ici ambigu. Grâce au contexte, le lecteur pourra comprendre qu'il faut l'interpréter dans le sens de « répondre aux besoins de ». L'expression « les différentes organisations... au niveau des unités de soins » signifie ici les différents types d'organisation que l'on peut rencontrer dans un établissement. L'exigence devient donc :

Le progiciel doit répondre aux besoins des différents modes d'organisation d'une unité de soin.

La formulation est un peu plus claire, cependant, « répondre aux besoins » est imprécis : on ne sait pas à quel besoin le progiciel doit répondre. Si le besoin est différent d'un mode d'organisation à l'autre, il faudra rédiger une exigence pour chacun d'eux. Sinon, l'exigence devient :

Le progiciel doit être utilisable dans toute unité de soins de l'établissement, quel que soit son mode d'organisation.

On s'aperçoit maintenant que l'imprécision vient du mode d'organisation. Il faudrait donc lister et expliciter les différents modes d'organisation de l'établissement, soit en intention (référence à une description de tous les modes d'organisation) soit en extension (faire la liste des modes d'organisation).

Or, ce qui est plus simple et plus efficace, c'est de décrire les différents modes organisationnels dans un chapitre préliminaire, décrivant le contexte. L'exigence devient alors :

Le progiciel doit être utilisable dans toute unité de soins.

Mais c'est là une tautologie. Cette exigence peut donc être purement et simplement supprimée... à moins d'en déduire une exigence non fonctionnelle d'utilisabilité : le progiciel doit être adaptable (ou déjà adapté ?) aux différents modes d'organisation de l'établissement.

## La structuration

---

### Structuration : adopter et adapter les modèles

Il existe plusieurs modèles de cahiers des charges<sup>1</sup>, comme X50-151, IEEE 830, ou le modèle Volere. Ces modèles sont extrêmement utiles : ils servent de check-lists et évitent d'oublier certains points indispensables. Ils ont une structure cohérente, descendante (*top-down*) et complète (toutes les catégories d'exigences et autres rubriques sont représentées). Ils constituent donc un bon départ pour la spécification des exigences, et même plus que la spécification : ils servent de check-lists pour le recueil.

Une organisation pourrait choisir un de ces modèles et l'adapter à sa situation particulière. Par exemple, une entreprise soumise au code des marchés publics peut y ajouter un chapitre sur les aspects juridiques. Si ce nouveau modèle ne correspond pas à toutes les situations d'une organisation, il pourra être raffiné au niveau d'une direction ou d'un service, et ainsi de suite.

---

1. Voir au chapitre suivant les différents modèles de cahiers des charges.

---

Rappelons que le modèle de développement et de gestion des exigences, tel que décrit dans cet ouvrage, peut (et souvent doit) être adapté au cas particulier de chaque entreprise. Il n'y a pas de modèle parfait.

### Réutiliser autant que possible

Une exigence bien formulée coûte cher à obtenir. Elle doit passer par le cycle complet de recueil, analyse, spécification et validation. À long terme, il est donc beaucoup plus efficace de rédiger des exigences réutilisables d'un projet à l'autre que de les réinventer et de les reformuler. C'est en particulier le cas des exigences non fonctionnelles, telle la sécurité.

Rendre les exigences réutilisables ou, mieux encore, élaborer du premier coup des exigences réutilisables, exige un bon niveau d'organisation. Il faut retrouver des exigences existantes, les adapter, et éventuellement les réinjecter dans un « pot commun » et, ce qui est plus difficile, conserver la cohérence d'ensemble. Au-delà d'une certaine complexité, les outils de gestion des exigences deviennent quasiment indispensables.

En pratique, cela nécessite :

- d'avoir une base d'exigences, bien tenue, c'est-à-dire contenant des exigences correctes (cohérentes, non ambiguës, etc.) ;
- d'avoir le moyen de les retrouver rapidement en fonction d'un certain nombre de critères ;
- une fois l'exigence réutilisée, c'est-à-dire utilisée dans plus d'une spécification, d'avoir les moyens de suivre son évolution et de répercuter les modifications sur plusieurs occurrences ;
- de pouvoir gérer la traçabilité sur chaque occurrence.

### Choisir la forme de description

Les exigences peuvent être décrites sous plusieurs formes : cas d'utilisation, modèles graphiques (traitements, données, flux, processus...) et exigences élémentaires.

Le choix de la forme (textuelle ou graphique), du niveau de détail (la « granularité » d'une exigence) en fonction des acteurs et du domaine d'application est un art plutôt qu'une science.

Il n'y a pas de dogme en ce qui concerne les modes de représentation. Par exemple, pour certains auteurs (comme Alistair Cockburn), les cas d'utilisation se suffisent à eux-mêmes dans 90 % des cas. Pour d'autres, ils ne sont qu'un moyen de découvrir les exigences fonctionnelles et n'ont pas à apparaître dans un cahier des charges. Pour d'autres enfin, cas d'utilisation et exigences fonctionnelles peuvent coexister au sein d'un même document.



En tout état de cause, le type incontournable d'exigence est *l'exigence élémentaire*.

## Spécifier les exigences élémentaires

Les exigences élémentaires constituent le cœur de la plupart des cahiers des charges. Par exigence élémentaire (en anglais, *atomic requirement*) on entend toute exigence formulée en langue naturelle sous forme de phrase qui se suffit à elle-même. Il peut s'agir là d'une exigence fonctionnelle ou non fonctionnelle, ou d'une contrainte produit ou projet. Par exemple :

- Le système présentera la liste des couleurs disponibles.
- Le temps maximum d'indisponibilité du système sera de 20 minutes par jour.
- Le système sera accessible au moyen d'un navigateur HTML 4.0.

Une exigence élémentaire peut découler directement d'un objectif, d'une règle de gestion, d'un cas d'utilisation, ou de toute exigence de plus haut niveau. Les exigences élémentaires sont regroupées sous forme arborescente.

## Cas des exigences non fonctionnelles

Les exigences non fonctionnelles sont particulièrement difficiles à formuler. En fait, chaque caractéristique non fonctionnelle (fiabilité, maintenabilité...) présente des difficultés de description particulières. En toute rigueur, il est nécessaire de spécifier (par exemple, ici, pour la fiabilité) :

- la définition de la mesure (par exemple, la définition du temps d'indisponibilité ou du temps de réponse) ;
- l'unité de mesure (par exemple, minutes) ;
- la valeur minimale ou maximale (par exemple : 20 minutes d'interruption maximum) ;
- la méthode de mesure (par exemple, chronomètre) ;
- la valeur optimale ;
- la valeur nominale et la tolérance ;
- les valeurs admises par exception (par exemple, le dimanche, une interruption de 30 minutes est autorisée).

Certains auteurs préconisent des modèles de structuration des exigences. Tom Gilb<sup>2</sup> propose un langage spécial de spécification des exigences non fonctionnelles, appelé *planguage* (*planning language*) qui permet de les décrire avec précision. Elle peut cependant être vue comme trop complexe et trop formelle pour la plupart des projets. Rien n'empêche de la simplifier, ou d'utiliser une description moins formelle.

---

2. Tom Gilb, *Competitive Engineering*, Butterworth-Heinemann, 2005.

---

Dans de nombreux cas, on peut se contenter d'une description beaucoup moins formelle, comme :

L'application doit pouvoir fonctionner 24 heures sur 24 et 7 jours sur 7 avec un taux d'indisponibilité maximal de :

- 15 minutes par jour dans la période comprise entre 7 heures et 23 heures ;
- 30 minutes par jour dans la période comprise entre 23 heures et 7 heures ;
- 3 heures par an pour maintenance.

Rien n'empêche, d'ailleurs, de s'inspirer de la norme ISO 25000 (plus précisément, les normes ISO/CEI 25020 à 25024, qui reprend les éléments de la norme ISO/CEI 9126-4). Cette norme donne une définition très précise d'un grand nombre d'indicateurs de qualité et de leurs métriques. Voici un exemple :

- Nom : temps moyen d'indisponibilité.
- But : temps moyen pendant lequel le système demeure indisponible entre une défaillance et la remise en ordre de marche graduelle.
- Formule de mesure :  $X = T/N$ , où T est le temps total d'indisponibilité et N le nombre de défaillances observées.
- Interprétation de la valeur mesurée :  $0 < X$  ; la plus petite valeur mesurée est la meilleure.
- Type d'échelle : ratio (rapport de valeurs).
- Type de mesure : T = temps, N = nombre, X = temps/nombre.
- Origine de la mesure : rapport de test, rapport d'intervention.
- Référence à une description de processus (ISO/IEC 12207) : 5.3 intégration, 5.3 tests de qualification, 5.4 opération, 6.5 validation.

La norme décrit ainsi des centaines de métriques sur presque cent pages, pour chacune des vingt-sept sous-caractéristiques ISO 25000. Elle pourra servir de check-list pour vérifier que toutes les exigences non fonctionnelles ont été intégrées. Elle apportera également des idées sur la manière de formuler une exigence et d'en fixer les critères de satisfaction.

## Check-list de spécification d'une exigence

---

La check-list qui suit concerne chaque exigence individuellement.

- L'exigence est alignée sur un objectif énoncé, une exigence de plus haut niveau, ou un besoin d'un type d'utilisateur.
- L'exigence décrit un besoin ou une contrainte, et non un élément de solution.

- L'exigence doit satisfaire au moins un besoin d'au moins un type d'utilisateur.
- L'exigence est spécifiée au niveau adéquat, ni trop détaillé, ni trop général, pour être comprise par toutes les parties prenantes.
- L'exigence émane d'une source reconnue et légitime, partie prenante ou texte de référence.
- L'exigence est spécifiée avec le formalisme (textuel, graphique) adéquat pour être comprise par tous les lecteurs.
- Si l'exigence est spécifiée textuellement, elle l'est sous forme active : sujet (acteur) verbe (action) complément (précisions).
- Les termes ambigus sont définis dans le glossaire.
- L'exigence est traçable. Il est possible, à partir d'une exigence donnée, de remonter à une exigence de plus haut niveau ou de plus bas niveau.
- L'exigence est complète. Elle représente une fonction élémentaire.
- L'exigence élémentaire est « atomique » : elle représente une seule fonction élémentaire insécable.
- L'exigence est grammaticalement correcte.
- L'exigence est formulée selon le modèle en vigueur.
- L'exigence n'entre pas en conflit avec une exigence de plus haut niveau.
- L'exigence est utile. Elle a été validée par au moins un représentant des utilisateurs.
- L'exigence est réalisable. Elle a été validée par un expert technique, concepteur ou développeur.
- L'exigence est non ambiguë. Elle a une seule interprétation possible, quel que soit le lecteur.
- L'exigence est concise. Elle est écrite de la manière la plus simple possible, avec le moins de mots possibles.
- L'exigence est vérifiable. Une fois le produit développé, il sera possible de vérifier objectivement et sans équivoque qu'elle est satisfaite.

### Check-list d'étape de spécification

---

Cette check-list est relative aux différentes activités de spécification : elle liste les conditions à remplir avant de passer à la prochaine étape (validation). Pour la formulation des exigences elles-mêmes, voir la check-list précédente, ainsi que la check-list relative au cahier des charges.

- Le modèle de cahier des charges a été accepté par tous les acteurs.
- Le modèle de cahier des charges a été adapté à l'organisation.

- Le modèle de cahier des charges a été adapté au projet. Les modèles graphiques utilisés sont compréhensibles par tous.
- Chaque rubrique du modèle de cahier des charges a été remplie, ou laissée vide avec justification.
- La formulation de chaque exigence a été vérifiée à l'aide de la check-list adéquate.
- Telles que formulées, les exigences seront utiles à la réalisation.

# Structure du cahier des charges

Choisir, adapter et utiliser un modèle de cahier des charges est une des clés de l'efficacité de son élaboration et de la qualité du livrable. En effet, un modèle est à la fois un guide pour agir, une check-list pour valider et un support pour communiquer sur les exigences. On trouve dans la littérature et sur le Web des modèles de cahier des charges prêts à utiliser, que l'on peut éventuellement adapter. Visite commentée.

## Le modèle de cahier des charges

### Les éléments du contenu

Un bon modèle de cahier des charges contient toutes les rubriques qui seront nécessaires à la description et à la compréhension des exigences, et rien de plus. On peut adapter le modèle aux contraintes particulières du domaine d'application et des technologies mises en œuvre. Cette opération n'est pas triviale. Malgré leur apparente simplicité, les modèles de cahiers des charges présentés dans ce chapitre résultent d'une longue réflexion, d'un travail collaboratif entre experts, et d'une expérimentation sur le terrain, parfois sur des centaines de projets.

### L'arborescence des paragraphes

La plupart des modèles de cahiers des charges (normes ou standards) comportent cinq à dix parties, découpées en dix à trente paragraphes.

L'ordre d'apparition des paragraphes est important. Il doit correspondre idéalement à l'ordre de lecture, et aussi, dans la mesure du possible, à la séquence des étapes du processus d'élaboration du cahier des charges.

### Les différents modèles existants

Nous présentons dans les paragraphes qui suivent différents modèles de cahiers des charges, et leurs domaines d'application possibles. Pour chacun d'eux, nous exposons sa structure, son contenu, ses avantages et inconvénients. La plupart sont téléchargeables, souvent gratuitement ou moyennant une somme modeste. Il est important d'investir du temps à bien choisir, bien comprendre et si nécessaire adapter un modèle. Cet investissement correspond à une fraction minime de l'effort qui sera par la suite consacré à l'élaboration du cahier des charges.

## Le modèle IEEE 830

---

### Orientation

---

1. IEEE Std 830-1998. *IEEE Recommended Practice for Software Requirements Specifications, Software Engineering Standards Committee, of the IEEE Computer Society.*

---

La norme de spécification des exigences IEEE 830 (1998)<sup>1</sup> donne un modèle généraliste, plutôt orienté vers le développement spécifique et assez technique.

Le modèle ne fournit pas une structure figée de cahier des charges, mais propose un ensemble de recommandations (emploi du verbe *should*) sur la manière de structurer le cahier des charges, et les différentes parties dont il doit se composer. Il fournit un squelette de cahier des charges. Enfin, une annexe informative donne huit manières de structurer les exigences, en particulier fonctionnelles.

### Structure du modèle

Le tableau 14-1 donne les éléments du document avec leur traduction en français.

**Tableau 14-1 – Modèle IEEE 830**

1. Introduction	1. Introduction
1.1 Purpose	1.1 Objectif
1.2 Scope	1.2 Périmètre (portée)
1.3 Definitions, acronyms, and abbreviations	1.3 Définitions, acronymes, et abréviations
1.4 References	1.4 Références
1.5 Overview	1.5 Vue d'ensemble
2. Overall description	2. Description générale
2.1 Product perspective	2.1 Perspective du produit
2.2 Product functions	2.2 Fonctions du produit
2.3 User characteristics	2.3 Caractéristiques utilisateurs
2.4 Constraints	2.4 Contraintes
2.5 Assumptions and dependencies	2.5 Suppositions et dépendances
3. Specific requirements	3. Exigences spécifiques
Appendixes	Annexes
Index	Index

## Contenu du document

Le modèle IEEE est un grand classique et contient tous les éléments que l'on s'attend à trouver dans un cahier des charges, du moins sur le plan technique. La norme précise que les exigences concernant la solution peuvent exceptionnellement y figurer, sous forme restrictive (contraintes).

La norme précise que les exigences fonctionnelles (paragraphe 2.2) peuvent contenir des diagrammes, ainsi que le paragraphe de perspective (2.1) dans le cas où le produit serait connecté à d'autres (ce qui, de nos jours, est presque toujours le cas). Un diagramme de contexte y trouverait sa place.

Le paragraphe 3 (exigences spécifiques) constitue le corps du document. La partie fonctionnelle doit contenir toutes les entrées et sorties du système (stimulus et réponses), et la description des fonctions en réponse à une entrée ou sortie. Les exigences non fonctionnelles (attributs) sont organisées en fiabilité, disponibilité, sécurité, maintenabilité, et portabilité.

La norme donne de nombreuses explications sur les différentes manières de structurer les exigences fonctionnelles : par mode, par classes ou

entités, par caractéristiques utilisateur, par stimulus et réponse, par hiérarchie fonctionnelle, ou par une combinatoire de ces types de structuration.

### Avantages et inconvénients de la norme

Le modèle a une forte orientation « arborescence fonctionnelle », les autres éléments (modèle de données, exigences non fonctionnelles) étant traités rapidement, voire passés sous silence. L'ensemble est donc assez technique, avec une coloration « génie logiciel ». Il fait d'ailleurs explicitement référence à la norme IEEE 12207 (cycle de vie du logiciel) avec laquelle il est compatible.

Le point fort de la norme est plutôt dans son annexe informative sur les différentes manières de structurer la décomposition des fonctions, qui donne des pistes de réflexion. Dans le cas d'un système de grande taille fonctionnelle (nombre important de fonctions élémentaires) le problème est loin d'être trivial, et les autres normes, plus récentes, le passent sous silence.

## Le modèle AFNOR X50-151

---

### Orientation de la norme

La norme AFNOR X50-151 donne un modèle de cahier des charges très généraliste, qui n'est pas spécifique à l'informatique et aux systèmes d'information. Il fait partie d'une série normative sur l'analyse de la valeur et l'analyse fonctionnelle. Il donne donc une place importante à l'ouverture sur le dialogue entre client et fournisseur.

### Structure du modèle

Le tableau 14-2 donne le plan d'un cahier des charges selon la norme AFNOR X50-151.



**Tableau 14-2 – Modèle AFNOR X50-151**

1. Présentation du problème
  - 1.1 Le produit et son marché
  - 1.2. Contexte du projet, les objectifs
2. Énoncé fonctionnel du besoin
  - 2.1. Cycle d'utilisation du produit et identification de son environnement
  - 2.2 Énoncé des fonctions de service et des contraintes
    - Fonctions de service attendues
    - Contraintes
    - Classement ou notation des fonctions
  - 2.3. Caractérisation des fonctions de service et des contraintes
    - Critères d'appréciation
    - Niveaux des critères d'appréciation
    - Flexibilité des niveaux : classe de flexibilité, limites d'acceptation
    - Taux d'échange
3. Appel à variantes
4. Cadre de réponse

### Contenu du document de spécifications

Les notions de niveau, de flexibilité et de taux d'échange sont des concepts d'analyse de la valeur. Le principe est que pour permettre au fournisseur d'optimiser sa solution (apporter le maximum de satisfaction au coût le plus faible), il est nécessaire d'indiquer la priorité entre exigences, car les exigences formulées par le client dans le cahier des charges n'ont pas toutes la même importance ou la même urgence.

- La classe de flexibilité (niveau de flexibilité de chaque exigence : nulle, faible, bonne, forte), indique dans quelle limite une contrainte est négociable.
- La limite d'acceptation définit, pour chacune des caractéristiques fondamentales du produit, une valeur minimale de la mesure du critère d'appréciation. En deçà de cette valeur, le besoin sera déclaré insatisfait.
- Le taux d'échange indique les compensations fixées a priori au cas où certains critères d'acceptation, jugés essentiels par le demandeur, ne seraient pas satisfaits (par exemple, les pénalités de retard, dans le cas où le fournisseur ne respecterait pas ses délais de livraison).

Enfin, l'appel à des variantes permet de stimuler la créativité du fournisseur. Le client peut solliciter des propositions d'amélioration qui, dans le cas d'un logiciel, seront fonctionnelles (création, modification ou suppression de fonctions) ou non fonctionnelles (performances, ergonomie). Les variantes peuvent être soit des suppléments, soit des alternatives au scénario préconisé.

Enfin, la norme prévoit explicitement un cadre de réponse, utile dans le cadre des marchés publics.

### Avantages et inconvénients

Avec les notions de niveau, de flexibilité, de limite d'acceptation et de taux d'échange, la norme X50-151 nous donne des pistes de réflexion intéressantes. Ces concepts sont assez difficiles à utiliser en pratique dans le contexte des systèmes d'information, mais on peut s'en inspirer pour construire un modèle adapté.

Le modèle n'est pas adapté aux spécificités du logiciel et des systèmes d'information. La structure est incomplète et le vocabulaire utilisé n'est pas celui du génie logiciel. Si on décide de l'utiliser, il faudra nécessairement le compléter avec des éléments d'autres modèles de cahiers des charges.

## Le modèle de Wiegers

---

### Orientation

Le modèle préconisé par Karl Wiegers est généraliste, simple et pragmatique, orienté vers le découpage fonctionnel. Il n'est pas très détaillé. Il pourra être utilisé pour spécifier des exigences pour un développement spécifique, mais également pour un progiciel.

### Structure du modèle

Remarquons un certain nombre de particularités concernant ce modèle<sup>2</sup> de document :

- Les interfaces avec le matériel, le logiciel et l'utilisateur sont regroupées dans un même chapitre. La spécification est centrée sur le produit, qui interagit avec des acteurs externes.
- Les modèles d'analyse (diagrammes de flux, de classes, etc.) sont mis en annexe. Les diagrammes ne sont pas un but mais un moyen d'arriver aux spécifications sous forme textuelle.

---

2. Le modèle peut être téléchargé sur le site de Karl Wiegers <http://www.processimpact.com/goodies.shtml>.

---

**Tableau 14-3 – Le modèle proposé par Karl Wiegiers**

<p>1. Introduction</p> <p>1.1 Purpose</p> <p>1.2 Document Conventions</p> <p>1.3 Intended Audience and Reading Suggestions</p> <p>1.4 Project Scope</p> <p>1.5 References</p> <p>2. Overall Description</p> <p>2.1 Product Perspective</p> <p>2.2 Product Features</p> <p>2.3 User Classes and Characteristics</p> <p>2.4 Operating Environment</p> <p>2.5 Design and Implementation Constraints</p> <p>2.6 User Documentation</p> <p>2.7 Assumptions and Dependencies</p> <p>3. System Features</p> <p>3.1 System Feature 1</p> <p>3.2 System Feature 2 (and so on)</p> <p>4. External Interface Requirements</p> <p>4.1 User Interfaces</p> <p>4.2 Hardware Interfaces</p> <p>4.3 Software Interfaces</p> <p>4.4 Communications Interfaces</p> <p>5. Other Nonfunctional Requirements</p> <p>5.1 Performance Requirements</p> <p>5.2 Safety Requirements</p> <p>5.3 Security Requirements</p> <p>5.4 Software Quality Attributes</p> <p>6. Other Requirements</p> <p>Appendix A : Glossary</p> <p>Appendix B : Analysis models</p> <p>Appendix C : Issues lists</p>	<p>1. Introduction</p> <p>1.1 Objectif</p> <p>1.2 Conventions documentaires</p> <p>1.3 À qui s'adresse ce document ; guide de lecture</p> <p>1.4 Périmètre du projet</p> <p>1.5 Références</p> <p>2. Description générale</p> <p>2.1 Perspective du produit</p> <p>2.2 Caractéristiques du produit</p> <p>2.3 Profils utilisateurs et caractéristiques</p> <p>2.4 Environnement d'opération</p> <p>2.5 Contraintes de conception et de réalisation</p> <p>2.6 Documentation utilisateur</p> <p>2.7 Hypothèses et dépendances</p> <p>3. Caractéristiques du système</p> <p>3.1 Caractéristique 1</p> <p>3.2 Caractéristique 2 (etc.)</p> <p>4. Exigences d'interface externe</p> <p>4.1 Interfaces utilisateur</p> <p>4.2 Interfaces avec le matériel</p> <p>4.3 Interfaces avec le logiciel</p> <p>4.4 Interfaces de communication</p> <p>5. Autres exigences non fonctionnelles</p> <p>5.1 Exigences de performance</p> <p>5.2 Exigences de sécurité (innocuité)</p> <p>5.3 Exigences de sécurité</p> <p>5.4 Attributs de qualité du logiciel</p> <p>6. Autres exigences</p> <p>Annexe A : Glossaire</p> <p>Annexe B : Modèles d'analyse</p> <p>Annexe C : Liste de problèmes</p>
---	--

### Contenu du document de spécifications

Dans ses ouvrages, Wiegers insiste sur la nécessité de décrire précisément la vision et la portée du projet (*vision and scope*) dans un document particulier, lors de la phase de préparation. S'il ne détaille pas ce point dans son modèle de document, c'est par ce qu'il préconise de se référer au document en question, ou de le recopier.

Chaque rubrique (chapitres et paragraphes) est commentée en quelques lignes. Ces différents points sont repris avec plus de détails dans l'ouvrage *Software Requirements*, une lecture utile à tout analyste des exigences.

### Avantages et inconvénients

De par sa simplicité, ce modèle pourra être utilisé par tout analyste ou consultant chargé d'élaborer un cahier des charges, même débutant. Ce dernier pourra enrichir le modèle pour traiter des cas plus complexes.

Sans négliger l'essentiel, le modèle est un peu pauvre en ce qui concerne les exigences non fonctionnelles. On pourra enrichir le paragraphe sur les attributs de qualité, par exemple en reprenant des éléments de la norme ISO 25000 déjà décrite.

## Le modèle Volere (Robertson & Robertson)

---

### Orientation

Utilisé sur de très nombreux projets informatiques (plus de deux mille, d'après leurs auteurs), le modèle Volere<sup>3</sup> est fortement orienté vers le développement de logiciels spécifiques. C'est néanmoins un modèle suffisamment complet pour être utilisé sur quasiment tout type de projet (choix de progiciel et mise en œuvre, développement spécifique) quel que soit le domaine d'application.

### Structure du modèle

Le modèle est structuré de manière « descendante », les paragraphes correspondent à l'ordre de lecture et, dans une très large mesure, à l'ordre d'élaboration du cahier des charges, depuis la définition des objectifs jusqu'aux détails les plus fins.

Le tableau 14-4 donne la structure du modèle et sa traduction en français.

---

3. Le modèle Volere est téléchargeable sur [www.volere.co.uk](http://www.volere.co.uk).

---

**Tableau 14-4 – Le modèle Volere**

<p><b>PROJECT DRIVERS</b></p> <ol style="list-style-type: none"> <li>1. The Purpose of the Project</li> <li>2. Client, Customer and other Stakeholders</li> <li>3. Users of the Product</li> </ol> <p><b>PROJECT CONSTRAINTS</b></p> <ol style="list-style-type: none"> <li>4. Mandated Constraints</li> <li>5. Naming Conventions and Definitions</li> <li>6. Relevant Facts and Assumptions</li> </ol> <p><b>FUNCTIONAL REQUIREMENTS</b></p> <ol style="list-style-type: none"> <li>7. The Scope of the Work</li> <li>8. The Scope of the Product</li> <li>9. Functional and Data Requirements</li> </ol> <p><b>NON-FUNCTIONAL REQUIREMENTS</b></p> <ol style="list-style-type: none"> <li>10. Look and Feel Requirements</li> <li>11. Usability and Humanity Requirements</li> <li>12. Performance Requirements</li> <li>13. Operational Requirements</li> <li>14. Maintainability &amp; Support Requirements</li> <li>15. Security Requirements</li> <li>16. Cultural and Political Requirements</li> <li>17. Legal Requirements</li> </ol> <p><b>PROJECT ISSUES</b></p> <ol style="list-style-type: none"> <li>18. Open Issues</li> <li>19. Off-the-Shelf Solutions</li> <li>20. New Problems</li> <li>21. Tasks</li> <li>22. Cutover</li> <li>23. Risks</li> <li>24. Costs</li> <li>25. User Documentation and Training</li> <li>26. Waiting Room</li> <li>27. Ideas for Solutions</li> </ol>	<p><b>MOTIVATIONS DU PROJET</b></p> <ol style="list-style-type: none"> <li>1. Objet du projet</li> <li>2. Clients et autres parties prenantes</li> <li>3. Les utilisateurs du produit</li> </ol> <p><b>CONTRAINTES DU PROJET</b></p> <ol style="list-style-type: none"> <li>4. Contraintes obligatoires</li> <li>5. Conventions de noms et définitions</li> <li>6. Faits et hypothèses déterminants</li> </ol> <p><b>EXIGENCES FONCTIONNELLES</b></p> <ol style="list-style-type: none"> <li>7. Périmètre de l'œuvre</li> <li>8. Périmètre de l'ouvrage</li> <li>9. Exigences sur les fonctions et données</li> </ol> <p><b>EXIGENCES NON FONCTIONNELLES</b></p> <ol style="list-style-type: none"> <li>10. Exigences d'interface utilisateur</li> <li>11. Exigences d'utilisabilité</li> <li>12. Exigences de performance</li> <li>13. Exigences opérationnelles</li> <li>14. Exigences de maintenabilité et support</li> <li>15. Exigences de sécurité</li> <li>16. Exigences culturelles et politiques</li> <li>17. Exigences légales</li> </ol> <p><b>QUESTIONS SUR LE PROJET</b></p> <ol style="list-style-type: none"> <li>18. Questions ouvertes</li> <li>19. Solutions sur étagère</li> <li>20. Problèmes nouveaux</li> <li>21. Tâches</li> <li>22. Finalisation</li> <li>23. Risques</li> <li>24. Coûts</li> <li>25. Documentation utilisateur et formation</li> <li>26. Questions mises en attente</li> <li>27. Idées de solutions</li> </ol>
---	---

Le modèle Volere est régulièrement revu et amélioré par ses auteurs, avec la contribution des analystes qui l'utilisent sur les projets. La dernière version est téléchargeable sur le site de Volere.

Le téléchargement du document en version anglaise est payant, et son utilisation sur un ou plusieurs projets est soumise au paiement d'une redevance. On trouve sur le site plusieurs traductions, dont une traduction partielle en français (téléchargement gratuit).

### Contenu du document de spécifications

Le modèle de cahier des charges de Volere contient tous les éléments indispensables à la spécification des exigences pour un logiciel, qu'il s'agisse d'une application spécifique ou du choix d'un progiciel. Dans le modèle fourni sur le site, chaque élément de la structure est abondamment commenté et justifié.

Notons que les contraintes, au sens large du terme, précèdent les exigences fonctionnelles. C'est une manière de baliser le terrain et de rappeler que les exigences fonctionnelles sont limitées par des contraintes inamovibles. Autre particularité : les exigences non fonctionnelles, ailleurs traitées comme le parent pauvre, occupent huit grands paragraphes.

### Avantages et inconvénients

Testé sur plus de deux mille projets, régulièrement mis à jour, le modèle est d'une solidité à toute épreuve... ou presque. Son principal défaut (qui n'en est pas un, bien sûr) est d'être trop riche pour la plupart des projets. Pour l'adapter, il suffit la plupart du temps de tailler cette arborescence un peu trop touffue.

De par sa complexité et sa complétude, les différentes rubriques sont parfois difficiles à interpréter. L'utilité de certains paragraphes ne saute pas immédiatement aux yeux. Plutôt que de les supprimer, nous recommandons de les laisser « sans objet », du moins en phase d'expérimentation, car ils pourraient se révéler très utiles par la suite.

Les aspects juridiques sont volontairement exclus par ses auteurs, et devront être rajoutés pour faire du document un CCTP (cahier des clauses techniques particulières) utilisable dans le cadre des marchés publics.

### Construire son propre modèle

---

À moins de travailler dans un laboratoire de recherche, inutile de « réinventer la poudre ». Construire son propre modèle de cahier des charges consiste à adapter un modèle, essentiellement par élimination de paragraphes inutiles, éventuellement par un apport venu d'un autre modèle.

La méthode est donc la suivante :

1. Examiner les différents modèles parmi les plus « classiques ».
2. Pour chaque modèle, cocher les rubriques qui seront utiles dans le cadre du (ou des) projets à traiter.
3. Choisir le modèle le plus proche.
4. Éliminer les paragraphes inutiles.
5. Si cela s'avère nécessaire, introduire avec prudence de nouveaux paragraphes, de préférence à partir de paragraphes repris sur d'autres modèles.
6. Vérifier et assurer la cohérence de l'ensemble.
7. Expérimenter le modèle, et éventuellement le modifier si nécessaire, en faisant valider les modifications.

Rappelons-le, un modèle de document est un investissement à long terme, particulièrement lorsqu'il s'agit d'un document comme un cahier des charges, et plus particulièrement si l'on a l'intention d'élaborer plusieurs cahiers des charges dans des domaines proches. Il ne faut donc pas hésiter à investir quelques heures à quelques jours sur cette adaptation.

### Check-list : cahier des charges

---

Contrairement à la check-list du chapitre précédent, qui concernait une exigence individuelle, cette check-list s'applique au document de spécification des exigences (cahier des charges) dans sa totalité.

- **Complétude.** Le document est complet. L'ensemble des exigences qui le composent couvre la totalité du problème.
- **Cohérence.** Il n'y a pas de conflit entre deux exigences, ni de redondance.
- **Traçabilité.** Toute exigence peut être tracée vis-à-vis d'une exigence de plus haut niveau ou d'un objectif.
- **Maintenabilité.** Le document est maintenable. Un historique des modifications, ou tout mécanisme équivalent, permet de réécrire une exigence sans perte de cohérence.
- **Priorisation.** Entre deux exigences de même niveau, il est possible d'indiquer laquelle des deux est prioritaire.
- **Monosémie.** Tout terme utilisé a la même signification pour tous les lecteurs dans tout le document.
- **Fonctions de persistance (CRUD).** Toute entité du système peut être créée (C = *create*), consultée (R = *read*), mise à jour (U = *update*) et supprimée (D = *delete*) (ou l'absence de cette fonction justifiée).





## Chapitre 15

# L'étape de validation

La validation des exigences permet de s'assurer que le document d'exigences est complet, correct et cohérent, que les exigences correspondent à des objectifs du maître d'ouvrage ou à des besoins des utilisateurs, et que ces exigences sont correctement formulées. Concrètement, la validation consiste en des revues de document, des inspections et des check-lists. Le principe est simple et la méthode efficace. Le reste est une question de discipline.

### Intérêt de la validation

---

La validation des exigences n'est pas qu'une formalité bureaucratique. Lorsqu'elle est menée avec sérieux par des gens de terrain, elle multiplie considérablement l'efficacité du processus d'exigences. C'est un investissement extrêmement rentable.

Les raisons sont faciles à comprendre :

- Les exigences sont systématiquement relues par d'autres que ceux qui les ont écrites, d'où une meilleure détection des incohérences.
- La maîtrise d'ouvrage (ou le marketing) ne perd pas de temps à interpréter des exigences floues et se concentre sur le contenu.
- Le processus d'élaboration est mieux géré, car il fournit des indicateurs d'avancement (un pourcentage d'exigences validées est plus parlant que le nombre total d'exigences produit).
- Le niveau de qualité des exigences devient prévisible, ce qui permet de planifier plus précisément le développement et les tests.
- Après validation, les exigences reflètent, dans une large mesure, les vrais besoins (on n'aura pas à recommencer le développement).

Il est rare que la validation soit la « tasse de thé » d'un groupe de travail ou d'un analyste. Cette activité en rebute plus d'un, car la plupart des personnes préfèrent aller de l'avant et voient donc les sessions de validation comme du temps perdu. Pour éviter de tomber dans ce piège, il faut planifier et organiser les sessions de validation et les inclure dans le processus et dans les plannings.

### Le processus de validation

---

La validation (figure 15-1) est essentiellement faite de vérifications par check-lists et de revues et inspections.

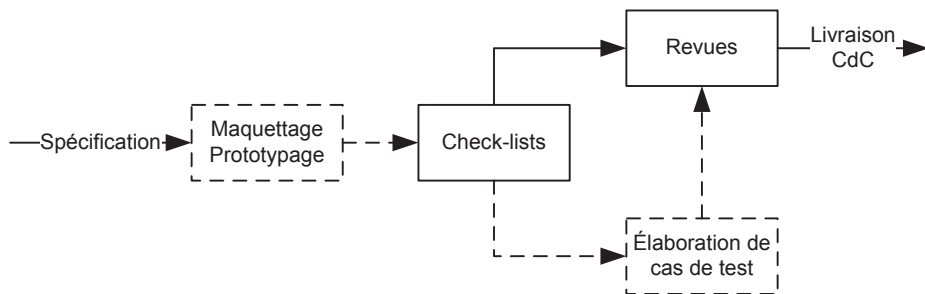


Figure 15-1 : Le processus de validation

Une validation formelle doit avoir lieu pour valider la totalité des spécifications (cahier des charges). Cependant, des validations partielles peuvent avoir lieu à divers moments, par exemple pour un ensemble de fonctions. On peut prévoir des minivalidations suite à une session d'un groupe de travail, soit par les membres du groupe lui-même, soit en organisant des validations croisées.

Des formes simples et rapides de validation (check-lists) peuvent avoir lieu quasiment à tout moment. Les différentes formes de reformulation d'une exigence lors d'une interview sont en quelque sorte des microvalidations.

### Les techniques

---

Les techniques de validation des exigences sont :

- la revue de document, plus ou moins formelle, qui est la technique de validation à proprement parler ;

- le maquettage et le prototypage, déjà mentionnés, qui sont également des techniques de recueil et d'analyse ;
- l'élaboration de cas de test à partir des exigences, qui constitue leur « épreuve du feu ».

Ces trois familles de techniques peuvent, bien entendu, être combinées. Dans ce chapitre, nous décrirons dans les détails les différentes formes de revue, depuis la liste de contrôle jusqu'à la revue formelle.

## Vérification par check-lists

---

### Check-lists de propriétés

Le moyen le plus simple de valider les exigences consiste à les confronter à une liste de contrôle (check-list). C'est aussi la première étape de la validation, qui permet de dégrossir la vérification. Les check-lists peuvent être utilisées par l'auteur du document à valider (première vérification) et doivent être utilisées par tous les autres.

On trouve dans la littérature des dizaines de variantes de check-lists, mais le principe est toujours le même. Il s'agit de vérifier les exigences individuellement (atomicité, non-ambiguïté, concision...) et le cahier des charges en tant qu'ensemble d'exigences (complétude, cohérence...). Le lecteur pourra utiliser les différentes check-lists déjà présentées dans cet ouvrage.

### Check-lists de contenu

Ces check-lists permettent de vérifier que le document de spécifications (cahier des charges) contient toutes les rubriques nécessaires. Ce sont des check-lists arborescentes qui reprennent la structure du cahier des charges. Par exemple : le document contient-il la liste des parties prenantes ? A-t-on spécifié la fiabilité ? Etc.

Nous conseillons de se servir du modèle de cahier des charges lui-même comme check-list. Si on a bien choisi et bien adapté le modèle de cahier des charges, on n'a besoin de rien d'autre. Il suffit de vérifier que tous les paragraphes ont été correctement remplis et qu'il ne reste pas de paragraphe vide. De plus, la plupart des modèles de cahier des charges contiennent non seulement l'ensemble des rubriques utiles, mais également des instructions sur *comment* les remplir. Cette autodocumentation dispense de toute check-list supplémentaire.

## Relecture simple

---

C'est, comme son nom l'indique, la forme la plus simple de contrôle d'un document. Elle fait intervenir deux personnes : l'auteur et le lecteur. Les étapes sont les suivantes :

1. L'auteur fournit le document au lecteur.
2. Le lecteur parcourt le document, y cherche les erreurs ou non-conformités, et note ses remarques (le plus souvent, sur le document lui-même).
3. Le lecteur retourne le document annoté à l'auteur.
4. L'auteur prend en compte les remarques.

Ce type de relecture relève plus de la vérification que de la validation. Il est utile pour passer rapidement en revue un ensemble de spécifications assez court (une dizaine de pages de cahier des charges est un maximum). L'action doit être brève, le cycle complet ne devrait pas dépasser 48 heures.

La revue simple est efficace si l'auteur et le lecteur sont du même niveau hiérarchique. Il ne s'agit ni de se faire approuver par son supérieur, ni de sous-traiter une tâche de relecture à son subordonné. La lecture elle-même devrait être rapide (cinq minutes par page, pour fixer les idées). Les annotations peuvent parfaitement se faire sur papier.

De telles revues se font souvent spontanément, basées sur le donnant-donnant (je relis ton document en espérant que tu reliras le mien). La difficulté n'est pas de les faire, mais de les institutionnaliser et de les budgéter. Autrement, elles dépendent d'une initiative individuelle et risquent de se faire rares lorsque les projets ou les collègues sont sous pression.

## Relecture croisée

---

La relecture croisée est un peu plus formelle que la lecture simple et fait intervenir un plus grand nombre d'acteurs : un rédacteur et plusieurs relecteurs. La procédure est la suivante :

1. Lors d'une réunion, l'auteur fait une présentation générale du document à plusieurs lecteurs et leur fournit le document.
2. Chaque lecteur lit le document isolément, y cherche les erreurs ou non-conformités, et note ses remarques.
3. Les participants se réunissent. Le document est parcouru pas à pas. Chaque participant indique les points qui lui semblent critiques. Un des participants note la liste des modifications à apporter.
4. L'auteur prend en compte les remarques et apporte les corrections.

Cette procédure est plus longue, plus coûteuse et plus efficace. Le cycle complet ne devrait pas dépasser deux semaines. L'équipe de relecture (auteur et relecteurs) doit être composée de trois à six personnes.

## Revue formelle et inspection

Une revue formelle est plus complexe, plus coûteuse et plus longue qu'une simple relecture du document. Cela demande une certaine discipline et un investissement important en temps, mais le retour sur investissement est beaucoup plus conséquent.

Une revue doit nécessairement avoir lieu à toute nouvelle version du cahier des charges, avant sa publication ou sa diffusion. On pourra aussi organiser des revues intermédiaires, sur un extrait du document, à la fin de certaines étapes.

On indique ici la procédure formelle, sachant que celle-ci peut être simplifiée en fonction des méthodes de travail, du budget-temps imparti et de la criticité du système.

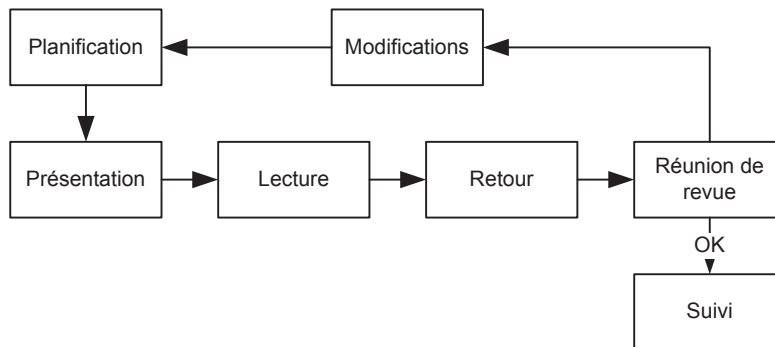


Figure 15-2 : Le processus de revue

La revue des spécifications d'exigences est analogue à toute revue de document. Les étapes d'une inspection sont : la planification, la réunion de présentation, la préparation, la réunion de revue, les modifications et le suivi.

Les participants à une revue sont :

- l'animateur (ou modérateur) de la revue, dans l'idéal différent de l'auteur du document ;
- l'auteur du document ;
- les relecteurs ;
- le secrétaire, de préférence différent de l'animateur.

L'objectif d'une revue est d'obtenir un consensus sur le document, et non à corriger des erreurs ou des non-conformités sur lesquelles tout le monde s'accorde. Si on veut éviter que les participants gaspillent leur temps et leur énergie sur des détails, il faut préalablement « toiletter » le document, selon la check-list :

- Le document est conforme au modèle.
- Les fautes d'orthographe les plus grossières ont été corrigées.
- La formulation est conforme aux règles de bonne présentation (voir check-list).
- On a préparé une liste des points à discuter en séance.
- Le document a été relu par une autre personne que l'auteur, qui n'y a pas trouvé de défauts majeurs.

La revue comprend les étapes suivantes :

- **Planification.** Choix des participants, du temps à consacrer par chaque participant, des dates de réunion, du nombre de réunions. La revue est une démarche itérative que l'on pourrait affiner à l'infini, il est préférable de fixer des limites par avance.
- **Réunion de présentation.** L'auteur du document explique aux participants le contenu du document et ce qu'il attend de la revue. Cette étape peut être supprimée si les participants ont l'habitude de travailler ensemble.
- **Lecture.** C'est l'étape la plus importante ; chaque participant passe en revue le document, à la recherche de défauts ; il peut s'aider d'une check-list et/ou s'appuyer sur son expérience. Il remplit une fiche de revue de document ou coche une check-list.
- **Retour.** Les participants renvoient leur fiche à l'auteur, qui prendra en compte les remarques.
- **Réunion de revue.** Les participants se réunissent pour discuter des points obscurs ou de désaccord. Chaque paragraphe est lu, et tout participant peut faire ses remarques. Les participants décident si une nouvelle réunion doit être programmée.
- **Modifications.** L'auteur du document prend en compte les remarques et fait les modifications convenues.
- **Suivi.** Un participant vérifie que toutes les remarques et points obscurs ont été pris en compte.

On peut boucler sur ces étapes jusqu'à obtenir un niveau de qualité satisfaisant. Pour décider s'il faut planifier une nouvelle tournée de revue ou s'arrêter, on pourra utiliser la check-list suivante :

- Toutes les remarques ont été prises en compte :
  - soit le document a été modifié ;
  - soit la remarque a été rejetée avec une justification.

- Le document modifié est correct sur la forme.
- Tous les points obscurs ont été levés.
- Le document est géré (nom et version, stockage, etc.).

Longue et coûteuse, mais utile et rentable, l'inspection formelle doit être institutionnalisée à haut niveau, d'autant plus que les participants peuvent faire partie d'entités très différentes et ne sont pas toujours des parties prenantes actives (il peut s'agir d'experts extérieurs). La difficulté est ici de retenir l'intérêt de relecteurs qui ne participent pas à l'élaboration du document.

## Élaboration de cas de test

---

Élaborer des cas de test correspondant à des exigences, c'est déjà mettre ces exigences à l'épreuve. Pour élaborer un cas de test, il faut imaginer le comportement réciproque de l'utilisateur et du système, ce qui fait ressortir toutes les incohérences qui étaient auparavant passées inaperçues. C'est donc un excellent moyen de faire émerger les défauts dans un document de spécification.

Les cas de test qui sont élaborés à ce niveau concernent évidemment des tests de validation et non des tests de vérification. Comme leur pendant au niveau des spécifications, ils décrivent le *quoi*, et non le comment.

Il est utile d'élaborer les cas de test le plus en amont possible. Plutôt que d'attendre d'avoir un cahier des charges entièrement bouclé, il est plus intéressant d'anticiper et de commencer l'élaboration des cas de test dès que l'on dispose d'un ensemble suffisamment autonome (c'est-à-dire cohérent et complet) d'exigences. Détecter des failles rapidement est très avantageux : cela évite de refaire les mêmes erreurs sur d'autres parties du cahier des charges.

Avant d'élaborer ces cas de test, nous conseillons de passer les check-lists concernant toutes les autres propriétés des spécifications (complétude, cohérence, etc.). L'élaboration de cas de test constitue pour ainsi dire l'épreuve du feu pour un ensemble d'exigences. Elle démontre leur *testabilité*, qui suppose que les autres propriétés (complétude, cohérence, non-ambiguïté, etc.) sont vérifiées.

Une variante de cette technique consiste à élaborer, à partir des exigences, la documentation du logiciel. L'élaboration de cette documentation peut être confiée à une autre équipe que celle responsable de l'élaboration du cahier des charges, ce qui permet de tester qu'elle est compréhensible par un lecteur extérieur. On procédera par la suite à une relecture croisée.

## Contrôle qualité des exigences

---

---

I. V. S. & J. Robertson, *Mastering the Requirements Process*, Addison-Wesley.

---

L'idée est d'avoir un ou plusieurs points de passage obligés par lesquels toute exigence passe systématiquement et sans exception. Robertson<sup>1</sup> parle de *quality gateway* que l'on pourrait traduire par « passage à niveau de la qualité ».

On peut décider que toute exigence peut se trouver dans l'un des deux états suivants : non validée ou validée. Le point de bascule d'un état à l'autre dépend bien sûr du niveau de rigueur du processus d'élaboration, et en particulier de validation.

Les variantes sont évidemment possibles. On peut imaginer un système à trois états ou plus, par exemple :

0. Proposée.
1. Vérifiée (check-list minimale).
2. Formellement acceptée (par le comité de validation).

La gestion de ces changements d'état est facilitée avec les outils de gestion des exigences. Avec de tels outils, « état » est une des propriétés de l'exigence. Certains outils permettent de définir une *baseline*, ou version de référence d'un ensemble d'exigences, où toutes les exigences sont validées.

## Impliquer les personnes concernées

---

Ces actions de validation ne serviraient à rien si elles n'étaient faites par les personnes adéquates au moment adéquat. De plus, les différentes parties prenantes doivent s'impliquer véritablement. C'est tout l'art de la validation. Le reste n'est que procédure.

C'est là que l'analyse des parties prenantes, qui a été faite lors de l'étape préalable, va s'avérer utile. Par exemple, le cahier des charges en version finale sera signé par le directeur général, après que de nombreuses validations partielles ont eu lieu, par d'autres analystes puis par les représentants des utilisateurs. Le maître d'œuvre devra s'engager à réaliser, ce qui est aussi une forme de validation. Auparavant, des experts auront vérifié que les exigences sont effectivement réalisables.

Il s'agit là d'un cas parmi d'autres. Réussir à impliquer tous les acteurs au bon niveau au bon moment demande un minimum d'organisation :

- avoir défini une carte des acteurs, c'est-à-dire un schéma ou un graphe des parties prenantes ;
- dès que possible, procéder à des validations informelles de spécifications partielles par les personnes du terrain (représentants des utilisateurs) ;



- au fur et à mesure que les spécifications se consolident, procéder à des validations de plus en plus formelles ;
- tenir informées les parties prenantes de plus haut niveau de responsabilité sans leur demander une validation formelle ;
- lorsque le document est finalisé, revu par toutes les parties prenantes, faire signer au plus haut niveau (par exemple, directeur général).

Cette manière de procéder ne garantit pas le succès, mais elle y contribue grandement. La pire des choses est d'obtenir un accord purement formel, une signature « les yeux fermés ».

### **Champagne !**

A contrario de la signature de pure forme sans réelle implication, l'approbation du cahier des charges au plus haut niveau de responsabilité peut légitimement apporter une grande satisfaction professionnelle à l'analyste. Cette satisfaction est d'autant plus importante que le décideur de haut niveau était, au départ, neutre, voire réticent. Un directeur va signer s'il a compris que le cahier des charges qui vient d'être produit reflète sa stratégie, et la décline sur le plan opérationnel, en termes plus ou moins techniques, mais compréhensibles par tous.

Pour arriver à ce résultat, il n'y a pas de miracle, pas de secret. L'analyste doit suivre, de bout en bout, un processus structuré de développement des exigences. Ainsi, chaque acteur y gagne en termes de coûts, de délais, de qualité. Rien de magique. Cela s'appelle l'ingénierie des exigences.

## Check-list : validation

---

- Le rédacteur a passé la check-list de bonne formulation.
- Le rédacteur a passé la check-list de structuration.
- Une personne extérieure a relu les spécifications.
- Les différentes parties prenantes ont participé à la validation.
- Une revue au moins a été effectuée.
- Un consensus sur les exigences a été obtenu.
- Les exigences sont aptes à être communiquées pour réalisation.



## Chapitre 16

# Un modèle de processus

Jusqu'ici, nous avons décrit très globalement la démarche générale de développement des exigences puis, plus en détail, les étapes qui la composent et les techniques que l'analyste peut mettre en œuvre à chaque étape. Nous proposons ici un processus, que chacun pourra adapter à son environnement.

### Un processus-type

---

Nous donnons donc ici un modèle générique, simple, facile à utiliser. Il devra être adapté et complété en fonction des particularités du projet et du produit, en particulier :

- les objectifs à atteindre ;
- le type de produit (spécifique, adaptation ou choix de progiciel) ;
- le mode de développement en interne ou en externe ;
- la taille et complexité du produit attendu ;
- la structure des parties prenantes ;
- l'organisation de la maîtrise d'ouvrage et de la maîtrise d'œuvre ;
- les procédures déjà existantes dans l'organisation ;
- les risques.

On fera particulièrement attention à adapter le vocabulaire, très différent d'un domaine d'application à l'autre. Des expressions comme comité de pilotage, comité directeur, ou plan projet peuvent avoir des significations sensiblement différentes selon les organisations. En cas de divergences entre le vocabulaire de la maîtrise d'ouvrage et celui de la maîtrise d'œuvre, c'est généralement celui de la maîtrise d'ouvrage (votre client) qui prime.

### Attention : la carte n'est pas le terrain

Rappelons la phrase du célèbre statisticien, George Box : « Tous les modèles sont faux, certains modèles sont utiles ». La carte est utile pour comprendre le terrain, mais la carte n'est pas le terrain. Une élaboration réelle de cahier des charges ne comportera probablement pas ces sept étapes. Elles sont là pour baliser la route. Chaque analyste responsable de l'élaboration d'un cahier des charges doit donc interpréter, et le plus souvent adapter, ce processus. En aucun cas, ce modèle de processus ne devra être suivi à la lettre et en l'état. Sous sa forme actuelle, c'est une check-list et une aide à la réflexion, et non une procédure.

Ce chapitre donne une carte de processus de développement des exigences. Le lecteur doit adapter cette carte en ajoutant, en modifiant ou en supprimant des étapes. Au niveau de chaque étape, il devra adapter les paragraphes pour en faire des fiches de procédure.

La figure 16-1 donne la carte du processus modèle.

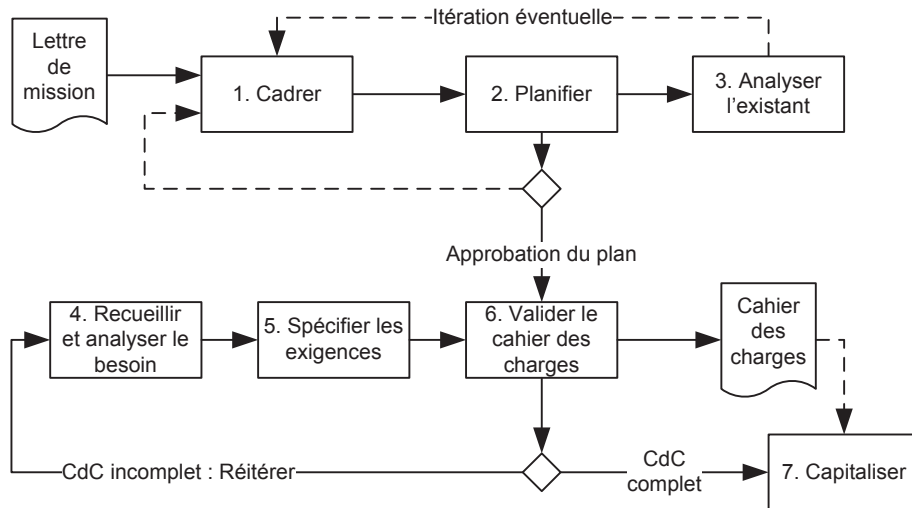


Figure 16-1 : Modèle de processus

## Étape 1 : cadrer

### Entrée

Le seul élément écrit en entrée est éventuellement une lettre de mission. Ce document n'est pas nécessairement une « lettre ». Mais en tout état de cause, c'est un document signé par le donneur d'ordres (maître d'ouvrage stratégique) et adressé à l'assistant à la maîtrise d'ouvrage (maître d'ouvrage délégué).

L'approbation formelle du donneur d'ordres est parfois difficile à obtenir. Si elle ne peut être obtenue à cette étape, elle devra l'être au plus tard dans la note de cadrage.

### Actions

Il s'agit d'avancer autant que possible sur le triangle objectifs-parties prenantes-périmètre. Les actions sont :

- **Recueillir les objectifs**, par interview des parties prenantes côté client (maître d'ouvrage) au plus haut niveau de responsabilité.
- **Analyser les parties prenantes**, du côté du client et du fournisseur, directes (utilisateurs et développeurs) et indirectes (impactées par le produit), stratégiques et opérationnelles.
- **Définir le périmètre** du projet et du produit, au moyen de réunions, dont une réunion formelle. Au plus tôt, il faudra entamer l'élaboration du diagramme de contexte de l'existant, et éventuellement de la cible. On peut à ce stade élaborer les en-têtes des cas d'utilisation métier.

### Sortie

Le document en sortie de cette étape est la note de cadrage. Il contient les éléments suivants :

- objectifs du projet ;
- structure des parties prenantes ;
- diagramme de contexte ;
- liste des événements métier ;
- cas d'utilisation métier ;
- description des cinq à dix exigences de haut niveau ;
- chiffrage grossier de la réalisation du produit.

### Conditions de sortie

La note de cadrage ne doit pas nécessairement faire l'objet d'une validation formelle. On peut attendre la sortie de l'étape 2 (planification) pour faire approuver ce cadrage par le donneur d'ordres. Il est cependant utile de le formaliser pour en discuter entre analystes et le soumettre au donneur d'ordres pour avis.

## Étape 2 : planifier

---

### Entrées

- Lettre de mission.
- Note de cadrage.

### Actions

- Choisir et adapter le processus de développement des exigences.
- Choisir et adapter le modèle de cahier des charges.
- Choisir et adapter les techniques.
- Choisir les outils.
- Déterminer les profils utilisateur.
- Planifier les jalons importants (comités de pilotage).
- Planifier les plages de travail pour les interviews.
- Planifier les groupes de travail.

### Sortie : le plan projet

- Profils utilisateurs types.
- Sources d'exigences.
- Tableau des charges, délais et planning.
- Fiches de risques.

### Conditions de sortie

À ce stade, le donneur d'ordres devra avoir approuvé la note de cadrage et le plan projet. Ces deux documents peuvent d'ailleurs être réunis en un seul.

## Étape 3 : analyser l'existant

---

### Entrées

- Liste des documents existants.
- Tableau des parties prenantes.
- Documentation de l'organisation existante.
- Documentation du système existant.

### Actions

- Rassembler la documentation sur l'existant.
- Analyser le cadre organisationnel.
- Examiner les services impactés par le fonctionnement du système.
- Si nécessaire, enrichir le diagramme de contexte de l'existant.
- Si nécessaire, enrichir le tableau des parties prenantes.
- Analyser les processus actuels.

- Dresser un diagramme de flux de processus présentant le circuit des informations entre services.
- Décrire les fonctions dans leur état actuel.
- Dresser le bilan de la situation actuelle :
  - inefficience des processus et leurs causes : circuits trop complexes, documents inutiles, règles de gestion obsolètes, moyens inadéquats ;
  - fonctions inadaptées, manquantes, inutilisables ;
  - difficulté d'utilisation ;
  - performances du système : temps de réponse trop longs, consommations ;
  - faiblesse des attributs non fonctionnels : fiabilité, portabilité, maintenabilité ;
  - support aux utilisateurs insuffisant, faible efficacité de la maintenance.
- Mentionner par sous-système ou domaine :
  - les principes ou les méthodes à remettre en cause ;
  - les processus ou procédures à revoir ;
  - les facteurs de qualité à revoir ;
  - les informations à préciser.
- Examiner, pour chacun des points précédents :
  - ce qui est à conserver, voire à amplifier ;
  - ce qui est à supprimer ou à remplacer ;
  - ce qui est à modifier.
- Analyser l'existant en termes d'offre sur le marché.
- Si nécessaire, étudier les scénarios d'évolution (redéveloppement, migration, choix de progiciel...).

### Sorties

- Panorama de l'état actuel du système.
- Organigramme des services concernés.
- Liste des fonctions, en leur état actuel.
- Diagrammes de processus, de séquence, d'états.
- Vision organisationnelle (postes de travail, nature des traitements, types de traitements).
- Solution technique actuelle : procédures, matériels et logiciels de base, logiciels applicatifs, progiciels.

- Bilan de la situation actuelle : principaux axes d'amélioration nécessaires au système actuel.
- Axes d'évolution (si une étude d'évolution a été menée) :
  - liste (non détaillée) des différents types de solutions fonctionnelles envisageables ;
  - brève présentation des scénarios étudiés ;
  - justification du scénario retenu ;
  - mesures envisagées pour conduire le changement ;
  - mesures envisagées pour faciliter l'appropriation du nouveau système par les utilisateurs.
- Comptes-rendus d'interview ou note de synthèse.

### Conditions de sortie

On peut passer à l'étape suivante lorsque les documents de sortie sont jugés suffisamment complets et lorsqu'un consensus s'est fait autour des scénarios d'évolution.

L'analyse de l'existant peut remettre en cause l'objectif ou le périmètre, avec éventuellement des répercussions sur les charges ou le planning. Si tel est le cas, il faudra obtenir l'approbation formelle du donneur d'ordres, après passage éventuel par les étapes 1 et/ou 2.

## Étape 4 : recueillir et analyser les besoins

---

### Entrées

Les entrées de cette étape sont les éléments de sortie des trois étapes précédentes, en particulier :

- lettre de mission ;
- note de cadrage ;
- plan projet ;
- étude de l'existant.

À ce stade, l'absence d'une lettre de mission donnant formellement à l'analyste les moyens et l'autonomie de mener sa mission est un facteur de risque important.

### Actions

- Planifier le recueil.
- Assembler la documentation.



- Recueillir les besoins organisationnels :
  - inventorer les règles de gestion ;
  - inventorer les contraintes.
- Recueillir les besoins utilisateurs :
  - mener les interviews ;
  - animer les groupes de travail ;
  - lire les documents.
- Synthétiser et classer les informations recueillies :
  - exigences métier ;
  - cas d'utilisation ;
  - exigences fonctionnelles ;
  - exigences non fonctionnelles ;
  - contraintes produit ;
  - contraintes projet ;
  - faits et hypothèses pertinents.
- Reformuler les exigences recueillies (cf. check-list de formulation des exigences).
- Distribuer les exigences recueillies aux participants aux groupes de travail.
- Analyser les exigences en groupe de travail :
  - discuter, en réunion de groupe, les informations recueillies ;
  - enrichir, si nécessaire, le diagramme de contexte ;
  - élaborer tout diagramme à même de faciliter la compréhension commune ;
  - réaliser, si nécessaire, des maquettes papier ;
  - réaliser, si nécessaire, un prototype.
- Vérifier les exigences (check-list de recueil et d'analyse).

### Sorties

- Comptes-rendus d'interview ou note de synthèse.
- Exigences reformulées.
- Diagrammes.

### Conditions de sortie

Lorsque les exigences recueillies forment un tout cohérent, les inclure dans le cahier des charges en cours d'élaboration (étape 5).

## Étape 5 : spécifier les exigences

---

### Entrées

- Modèle de cahier des charges.
- Exigences formulées.

### Actions

- Reformuler si nécessaire les exigences recueillies à l'étape précédente.
- Passer la check-list de formulation des exigences.
- Enrichir le cahier des charges avec les exigences recueillies à l'étape précédente.
- Passer la check-list de l'étape de spécification.

### Sorties

- Cahier des charges enrichi.
- Plan projet réactualisé.

### Conditions de sortie

Après passage réussi des check-lists, passer à l'étape suivante.

## Étape 6 : valider les exigences

---

### Entrées

Lors de la première itération, c'est uniquement le squelette de cahier des charges qui devra être validé. Pour les itérations suivantes, on devra valider le contenu, sauf si le squelette a dû être modifié. Les éléments d'entrée sont donc :

- cahier des charges ;
- note indiquant quelles parties du cahier des charges sont à l'état « à valider » ;
- modèle de fiche de lecture, vierge.

Les indications sur les parties du cahier des charges à valider peuvent se trouver dans le cahier des charges lui-même, par exemple sous forme de texte en couleurs, surligné, ou en utilisant le mode révision d'un outil de traitement de texte.

### Actions

Appliquer la procédure de validation, selon le protocole convenu :

- revue ou inspection de document ;

- maquette, prototype ;
- élaboration de cas de test.

### Sorties

- Exigences validées.
- Fiches de relecture remplies.

### Conditions de sortie

Les validations à chaque itération peuvent être formelles ou informelles. La dernière itération donnera lieu à une validation formelle.

S'il reste des exigences à recueillir, repasser à l'étape 4.

Si le cahier des charges est exhaustif, mais que les parties prenantes ont fait des remarques, repasser à l'étape 5.

Si la totalité du cahier des charges est validée par la totalité des parties prenantes concernées (donneur d'ordres, maîtrise d'ouvrage), le processus d'élaboration du cahier des charges est terminé. L'étape suivante (capitalisation) permettra d'améliorer le processus d'élaboration lui-même.

## Étape 7 : capitaliser

---

### Entrées

Tous les documents élaborés lors des étapes précédentes.

### Actions

- Lors d'une réunion, solliciter un feedback des parties prenantes.
- Modifier, si nécessaire, le modèle de cahier des charges.
- Modifier ou enrichir les check-lists.
- Modifier, si nécessaire, les procédures d'élaboration du cahier des charges.
- Remercier les participants.
- Célébrer la fin de la mission.

### Sorties

- Nouveau modèle de cahier des charges.
- Procédures modifiées ou enrichies.
- Rapport de mission.



## Chapitre 17

# Améliorer le processus

Dans les chapitres précédents, nous avons vu que le processus de développement des exigences doit être adapté à l'organisation, aux objectifs et au type de projet. Et comme tout processus, il y a des moyens pratiques de l'améliorer. Pour rendre le processus plus efficace, il faut chercher où se trouvent les gisements d'efficacité. Après une explication des facteurs d'amélioration, nous donnons quelques conseils pratiques<sup>1</sup>.

---

1. La lecture de ce chapitre n'est pas indispensable à la compréhension des chapitres qui suivent.

---

## Pourquoi le processus peut être amélioré ?

---

Quelle que soit la démarche choisie, le processus comporte, dans ses grandes lignes, toujours les mêmes étapes : recueil, analyse, spécification et validation. Peuvent s'y ajouter, selon les auteurs, l'étape préalable de cadrage ou préparation, et l'étape supplémentaire de capitalisation.

Les ouvrages sur l'ingénierie des exigences nous disent que les quatre (ou cinq, ou six...) étapes du processus sont « imbriquées ». Et tout praticien de la méthode sait que ce processus n'est ni linéaire ni fermé. Comprendre comment et pourquoi ces étapes sont « imbriquées » va nous permettre d'améliorer le processus. Or, il y a trois raisons pour lesquelles les étapes sont interdépendantes :

- Le processus n'est pas linéaire : il contient des boucles de rétroaction.
- Le processus est récursif : chaque étape contient les autres.
- Le processus est itératif : on travaille par couches successives.

Nous allons décrire ces trois mécanismes et voir comment ils peuvent influencer, positivement ou négativement, l'efficacité du processus. Après cette analyse, nous aurons en main les moyens d'améliorer cette efficacité.

### Rétroactions

**2. Retravail (*rework*).**  
Formellement, c'est l'étape de prise en compte des remarques suite à une inspection ou une revue formelle.

Le premier mécanisme est facile à comprendre : les retours à une étape précédente<sup>2</sup> sont inévitables. Ainsi, si au cours d'une étape, il manque des informations (incomplétude), on remontera à l'étape de recueil. Si l'on découvre une incohérence, on remontera à l'étape d'analyse. Cela n'a rien de dramatique, bien sûr, mais occasionne des pertes de temps et d'énergie. Rappeler une personne au téléphone pour lui demander des précisions est plus coûteux en temps et en effort que, par exemple, reformuler ce qu'il vient de dire. Ainsi, pour optimiser le processus, il faudra faire en sorte que les retours aux étapes précédentes soient les moins fréquents possibles.

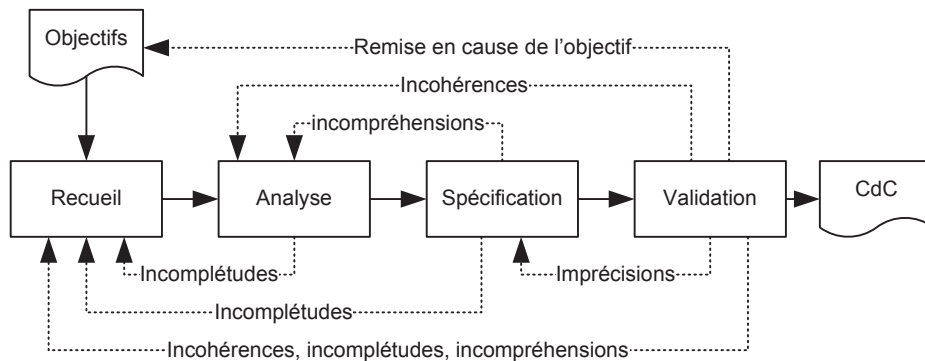


Figure 17-1 : Rétroactions

### Récursions

La deuxième raison pour laquelle les quatre étapes sont imbriquées tient à la nature récursive du processus. Dit plus simplement, chaque étape contient des éléments des autres étapes. Par exemple, lorsque l'on recueille les exigences, on ne va pas se contenter d'écouter et de transcrire. On va aussi analyser, voire spécifier, cette exigence. On va pouvoir anticiper sur les étapes à suivre. Cette anticipation peut accélérer le processus, mais aussi le dégrader.

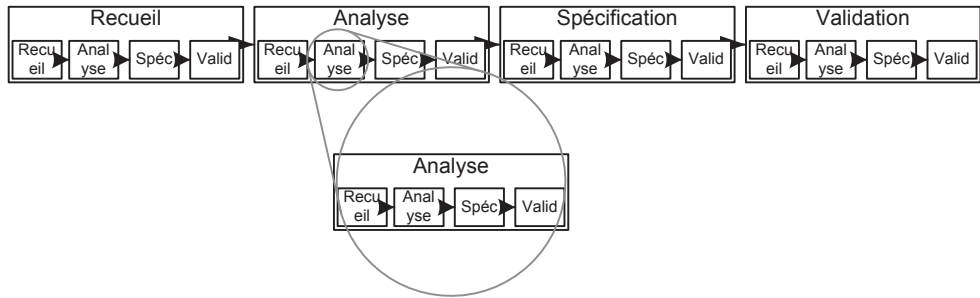


Figure 17-2 : Récursivité du processus

## Itérations

La troisième raison, c'est que le processus est itératif : les spécifications sont produites par couches successives. Examinons, par exemple, ce qui se passe lors de la phase de cadrage. Nous allons recueillir les objectifs, les analyser, les spécifier et les faire valider. L'étape préalable n'est ni plus ni moins qu'un minicycle de la phase d'exigences. Il est possible de systématiser cette approche et de fournir, par itérations, des cahiers des charges de plus en plus précis. Ici aussi, cette façon de travailler peut accélérer le processus ou le dégrader.

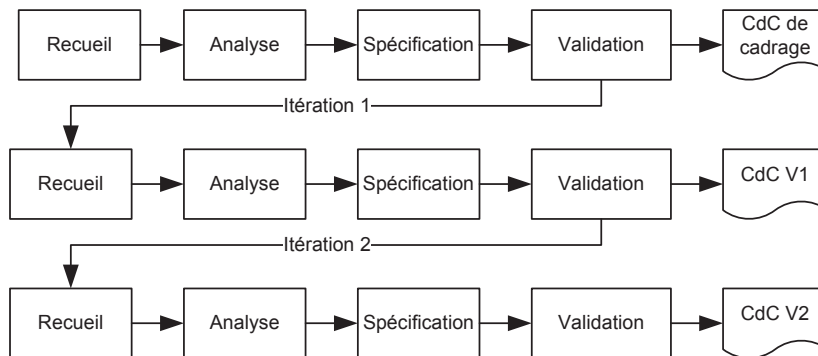


Figure 17-3 : Processus itératif

## Comment le processus peut être amélioré

En résumé, trois moyens sont à notre disposition pour optimiser le processus de développement des exigences :

- minimiser les retours en arrière chaque fois que cela est possible ;

- anticiper sur les étapes suivantes, lorsque cela est utile ;
- considérer l'option de travailler de manière itérative, par couches successives.

Voyons comment, de manière pratique, ces moyens peuvent être mis en œuvre.

### Minimiser le retravail (rework)

Nous avons introduit, tout au long de cet ouvrage, un outil simple et efficace pour minimiser les retours en arrière : la check-list. Un simple contrôle nous permet de savoir si nous avons fait tout ce qui devait l'être à une étape donnée. Le présent ouvrage contient plusieurs check-lists, à plusieurs niveaux, mais d'autres check-lists peuvent être élaborées pour enrichir le référentiel méthodologique d'une équipe d'analystes.

Remarquons que les documents types sont également des check-lists d'une forme plus élaborée. C'est en particulier le cas du modèle de cahier des charges.

### La technique de l'anticipation

Anticiper ne signifie pas, bien sûr, brûler les étapes (cela aurait l'effet contraire à celui escompté), mais bien préparer chaque sous-étape ou micro-étape pour analyser, spécifier ou faire valider au plus tôt ce qui peut l'être.

Pour anticiper sur les étapes suivantes, nous avons déjà donné quelques « trucs » dans les différents chapitres. Cette technique d'anticipation peut être systématisée.

Anticiper sur les étapes à venir a des avantages et des inconvénients. Parfois, il est intéressant d'apporter de la précision le plus vite possible, et de sortir d'une interview avec des exigences déjà clairement formulées. À d'autres moments, il sera inutile de spécifier avec précision des exigences qui seront remises en cause à la prochaine interview.

Prenons l'exemple d'un outil de recueil bien connu : l'interview. Suite à l'interview proprement dite, l'analyste peut mener les actions suivantes :

- mini-analyse : confronter les notes d'interview avec les autres éléments recueillis et chercher à lever les ambiguïtés et les incohérences ;
- minispécification : les notes d'interview peuvent être structurées et bien formulées au lieu d'être laissées telles quelles ;
- mininvalidation : l'analyste envoie à la (ou aux) personne(s) interviewée(s) ces spécifications partielles pour validation.

Ces opérations peuvent être menées pour la plupart des activités de recueil : réunion de groupe de travail, brainstorming, et même analyse de la documentation.



Cette démarche de préanalyse, préspecification et prévalidation peut se faire au niveau des microétapes. Par exemple, lors d'une interview sur le terrain, suite à une réponse apportée par l'interviewé à une question, l'analyste peut, si les circonstances le permettent, faire en temps réel une :

- microanalyse : confronter la réponse donnée à d'autres exigences ; utiliser des diagrammes et en discuter avec son interlocuteur ;
- microspécification : reformuler le plus précisément possible la réponse de l'interviewé, sous une forme claire et cohérente, prête à être insérée dans le cahier des charges ;
- microvalidation : l'exigence ainsi spécifiée par écrit est montrée séance tenante aux interlocuteurs pour validation.

Plus rapidement on éliminera les erreurs, et plus on pourra s'appuyer sur un terrain solide pour continuer. Cet ouvrage donne plusieurs exemples de mini-anticipations, comme les check-lists, et de microanticipations, comme la reformulation. Cependant, ces microanticipations ne sont pas toujours réalisables, ni même souhaitables. C'est l'expérience qui permet de savoir s'il est opportun de les mettre en œuvre. L'analyste doit mettre en balance le bénéfice apporté par une analyse ou une formalisation immédiate, et sa possible remise en question par la suite (lors d'une autre interview, par exemple).

### Le travail itératif

Travailler par « couches horizontales » peut être très intéressant. Mais là aussi, il n'y a pas de règle absolue. Dans son ouvrage sur le travail collaboratif sur les exigences<sup>3</sup>, Ellen Gottesdiener consacre un chapitre entier aux différentes manières de naviguer entre les exigences avec les groupes de travail, soit horizontalement, par couches successives, soit verticalement, en descendant dans les détails au cours d'une réunion, soit même en zigzag.

La première couche horizontale, celle de la définition des objectifs et du cadre, sera généralement bien séparée des autres. Nous l'avons décrite en détail au chapitre 6. Elle donne lieu à un livrable particulier. Rien n'empêche d'ailleurs d'utiliser ce livrable comme premier chapitre du cahier des charges.

---

3. Ellen Gottesdiener, *Requirements by collaboration*, Addison-Wesley, 2002.

---

## La méthode du document navette

---

### Un principe simple

La technique consiste à mener les travaux, du premier recueil des objectifs jusqu'à la validation finale, autour de la création et de la tenue à jour

d'un document unique, le cahier des charges. Cette démarche simple et efficace présente plusieurs avantages :

- Le cahier des charges devient le fil directeur du projet. Il est à la fois le matériau sur lequel on travaille et l'objectif à atteindre.
- La progression est visible de tous. L'enrichissement progressif d'un document partagé permet à chacun de constater les progrès en temps réel.
- Les acteurs ont un seul et unique document à gérer. Ce document central va devenir un véritable outil de communication entre les parties prenantes.

Avant de partager le document, il faudra figer, autant que possible, le modèle de cahier des charges, qui va couvrir les différents aspects du projet et faciliter la structuration des idées et des activités. Le sommaire du document va constituer un guide pour le recueil et l'analyse des besoins, la définition fonctionnelle et les orientations de solution. Toute modification ultérieure devra faire l'objet de négociations. Cette étape est sous la responsabilité de l'analyste, qui est le garant de la forme et de la méthode.

En s'étoffant, le document va permettre de contrôler la couverture et la progression de tous les aspects, au fur et à mesure de l'avancement du projet. Bien entendu, cela n'empêche pas de prévoir des documents complémentaires (comptes-rendus de réunions du comité de pilotage, par exemple). Mais c'est toujours le document navette qui fait foi.

De manière à optimiser ses travaux d'animation et de conseil à la maîtrise d'ouvrage, l'analyste va enrichir le cahier des charges de façon continue au cours des étapes de recueil, d'analyse et de spécification, ce qui réduira l'effort de rédaction et de validation. Le document changera de version au cours du projet, en passant d'un état exploratoire à un état provisoire, pour se figer, après validation par les parties prenantes, à l'état de référence.

Cette manière de procéder permet le partage du contrôle de la qualité des livrables pour ainsi dire en temps réel, et un dialogue plus fluide entre maîtrise d'ouvrage et maîtrise d'œuvre. Le modèle de cahier des charges, puis les versions successives de celui-ci serviront de support à ce dialogue. Cette coconstruction permet de déterminer précisément les priorités en termes d'exigences non fonctionnelles, car maîtrise d'ouvrage et maîtrise d'œuvre construisent ensemble un vocabulaire commun et non ambigu.

### Une pratique simple et efficace

La technique du document unique ne résout pas tous les problèmes. Elle minimise le nombre de documents et demande beaucoup de rigueur dans la gestion des versions du cahier des charges. Plutôt que de gérer une multitude de documents (généralement non versionnés), on devra gérer les versions multiples, et généralement arborescentes, d'un document

central. C'est tout aussi complexe, mais plus efficace lorsque l'on maîtrise la technique. Par exemple, le cahier des charges est à mi-parcours :

- Lors de la réunion d'un groupe de travail, l'animateur note les exigences au fur et à mesure du recueil, soit sur papier, soit directement sur le document lui-même. Il peut utiliser une couleur différente pour les différentes modifications apportées (vert pour un nouveau texte, rouge pour un texte à supprimer, violet pour un texte à revoir, etc.). Il peut également utiliser le mode révision du traitement de texte (attention, certains ont beaucoup de mal avec le mode révision, d'autres ne le supportent tout simplement pas).
- À la fin de la réunion, ou dans les deux jours qui suivent, l'animateur envoie le document aux différents acteurs, pour validation.
- Les remarques sont prises en compte et une nouvelle version du document est créée.

Cette manière de procéder peut également être mise en œuvre si l'on utilise des outils de gestion des exigences. En fonction des circonstances et de la typologie des acteurs, l'outil sera soit directement utilisé par les groupes de travail, soit uniquement utilisé par le consultant ou les animateurs. La plupart des outils permettent de générer un document de spécifications (cahier des charges en devenir) qui est le reflet de la base d'exigences. Ici, il n'y a pas de méthode unique, elle est très dépendante des outils utilisés et de la culture de l'entreprise.



# PARTIE 3

---

## Faire vivre les exigences

Il est bien sûr illusoire de penser que les exigences, une fois formulées, vont rester gravées dans le marbre. En réalité, les exigences évoluent tout au long du cycle de vie du logiciel, c'est-à-dire avant, pendant, et après la phase d'exigences. Les changements dans les exigences vont donc impacter non seulement le cahier des charges, mais le produit en cours d'écriture ou déjà écrit. La gestion des exigences est donc très liée à la gestion des versions et de la configuration du logiciel. Dans ce cadre, les outils de gestion des exigences peuvent améliorer l'efficacité et la sécurité de ce processus.



## Chapitre 18

# La gestion des exigences

Dans cet ouvrage, il a été essentiellement question de ce qu'il est convenu d'appeler le développement des exigences, processus qui permet, en partant d'un objectif, d'aboutir à un cahier des charges. En réalité, les demandes et contraintes évoluent tout au long du cycle de vie du logiciel : pendant la phase d'exigences, mais également pendant les phases ultérieures : conception, réalisation, intégration et maintenance. Il est illusoire de penser que les exigences sont un jour stabilisées. Ce chapitre est consacré à la gestion des demandes de changement, activité appelée gestion des exigences.

## La gestion des changements

La gestion des changements des exigences est à la frontière de plusieurs activités d'ingénierie du logiciel, en particulier la gestion des versions, la gestion de la configuration logicielle, et la gestion de projet. En effet, tout changement dans les exigences aura un impact sur :

- la configuration du logiciel<sup>1</sup> ;
- les coûts et les délais de livraison des différentes versions ;
- les ressources engagées pour opérer les modifications.

L'étude de ces disciplines (gestion de configuration et gestion de projet) dépasse le cadre de cet ouvrage. Nous n'en aborderons ici que les aspects en relation directe avec l'évolution des exigences.

Le changement en lui-même est inévitable. Il doit être contrôlé de manière à :

- conserver la cohérence entre demandes émanant d'utilisateurs ou d'acteurs différents et non concertés ;

---

**1. Gestion de configuration** : discipline qui consiste à gérer les divers composants d'un système, ainsi que les modifications apportées au cours de son évolution.

---

- maîtriser le coût induit par une modification (une modification en apparence légère, pourra induire un impact très important en termes de coûts, de délais, d'architecture du produit) ;
- éviter les « spécifications rampantes », suite à une demande initiale souvent mal formulée et peu formalisée, avec des dérives sur les délais, les charges et la qualité du produit ;
- contrôler le circuit de décision, éviter de court-circuiter les parties prenantes qui ont la légitimité pour décider ou conseiller ;
- tracer les modifications demandées et les modifications faites.

### Les activités

Cette gestion du changement est donc essentielle, et évite que l'effort initial de recueil, d'analyse et de structuration des exigences soit perdu. Les activités de gestion consistent à :

- évaluer l'impact d'une exigence sur les exigences existantes, et l'impact de sa mise en œuvre sur le produit développé ou en cours de développement ;
- estimer l'importance, l'urgence et la difficulté de réalisation d'une modification, qui peuvent être très différentes lorsqu'elles sont estimées par le demandeur (qui a une vision partielle de la situation) ou par les développeurs (qui voient surtout les contraintes techniques) ;
- négocier les modifications et gérer les exigences selon les priorités ainsi définies, après obtention d'un consensus ;
- suivre les modifications, et mettre à jour les plannings ;
- enregistrer les demandes et les modifications ;
- mettre à jour les documents (cahier des charges, spécifications, documentation du produit).

### La commission de contrôle des changements (CCB)

Afin que les changements soient gérés plutôt qu'imposés ou subis, la bonne pratique consiste à mettre en place une commission de contrôle des changements (en anglais, CCB, *change control board*). Cette commission devra être un passage incontournable pour toute demande de modification des exigences<sup>2</sup>. Toute demande de changement, même minime, devra être formalisée au moyen d'un formulaire ad hoc, puis passer par cette commission. La commission décide de la suite à donner, en fonction des priorités de l'entreprise, des impacts sur les coûts et les délais, des modalités de prise en compte d'un changement dans les exigences.

Plusieurs modèles d'organisation sont possibles, en fonction de la taille de l'organisation et du nombre de projets en cours. L'organisation la plus

---

2. Le plus difficile : constituer une commission de contrôle des changements qui soit légitime et institutionnaliser le processus de contrôle de manière à le rendre incontournable.

---



classique consiste à mettre en place une commission par projet, présidée par le chef de projet maîtrise d'œuvre.

La commission est formée au minimum d'un représentant de la maîtrise d'œuvre et d'un représentant de la maîtrise d'ouvrage, ainsi que d'un expert et d'un gestionnaire de configuration. Elle se réunit à dates fixes, ou en cas de besoin. Ses prérogatives sont :

- la revue des demandes de changement ;
- l'analyse d'impact (déléguée à un développeur ou expert) ;
- la priorisation des demandes ;
- la décision de faire, de ne pas faire, ou de reporter le changement, en fonction des impacts sur les coûts et les délais ;
- le suivi de la mise en œuvre du changement ;
- en cas de difficulté, l'escalade du problème au plus haut niveau de décision.

### La réunion de la commission

La gestion des changements sur les exigences est un processus rigoureux, illustré par la figure 18-1.

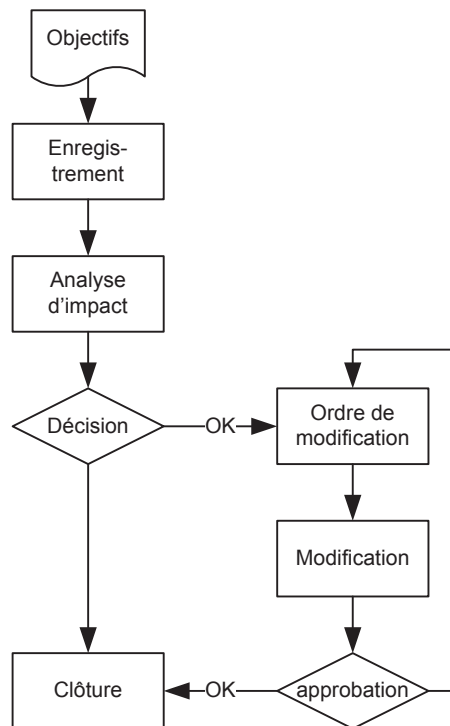


Figure 18-1 : Le processus de gestion des changements

Chaque demande de changement est enregistrée, puis examinée par la *commission de contrôle des changements* qui, en fonction de la nature, de l'urgence et de l'importance du changement à apporter, la transmettra à un expert (concepteur, développeur, souvent plusieurs personnes) qui va analyser l'impact de la modification. Suite à cette analyse, la commission de contrôle des changements peut, soit rejeter la demande, soit décider de demander une ou plusieurs modifications. Cette modification peut être planifiée pour la version en cours de développement, ou pour une version ultérieure. Lorsque toutes les modifications relatives à une demande de changement ont été effectuées et approuvées, la demande est close.

### L'analyse d'impact

L'analyse d'impact peut être effectuée par un ou plusieurs experts (assistant maîtrise d'ouvrage, développeur, testeur), mais elle est sous la responsabilité d'une seule personne, nommément désignée. Elle se concrétise par une fiche (le tableau 18-1 est un exemple de fiche d'analyse d'impact).

### Les attributs des exigences

Afin de pouvoir être gérée, chaque exigence doit comporter un certain nombre d'attributs :

- date de création ;
- numéro de version ;
- demandeur (partie prenante à l'origine de la demande) ;
- auteur (personne qui a formalisé la demande) ;
- son statut (provisoire, mise en œuvre, supprimée, ajournée) ;
- priorité ;
- degré de stabilité ;
- version du logiciel impacté par le changement ;
- et composant du logiciel où l'exigence est mise en œuvre.

Pour gérer tous ces attributs avec rigueur, mais sans surcharge excessive de travail, les outils de gestion des exigences deviennent vite indispensables.

**Tableau 18-1 – Un modèle de fiche d'analyse d'impact**

Rapport d'analyse d'impact	
Numéro de demande de changement	
Intitulé de la demande	
Description précise de l'exigence	Pris en charge par : Le :
Risques encourus si le changement est mis en œuvre	Risques encourus si le changement n'est pas mis en œuvre
Estimation du coût du changement (jours*hommes)	Estimation de l'effort perdu (coût de la suppression d'une fonction déjà existante + coût d'un développement qui sera abandonné)
Impact sur le planning global du projet	Impact du changement sur les charges totales
Impact sur la qualité finale des livrables (performances, maintenabilité... )	
Autres exigences impactées	
Tâches affectées si le changement est mis en œuvre	
Problèmes d'intégration estimés	
Liste des documents qui seront touchés par une modification	
Liste des modules de code qui seront touchés par une modification	
Liste de tous les autres éléments qui seront touchés par une modification	

## L'évolution des demandes de changement

La figure 18-2 présente le diagramme d'états des demandes de changement et d'ordres de modification.

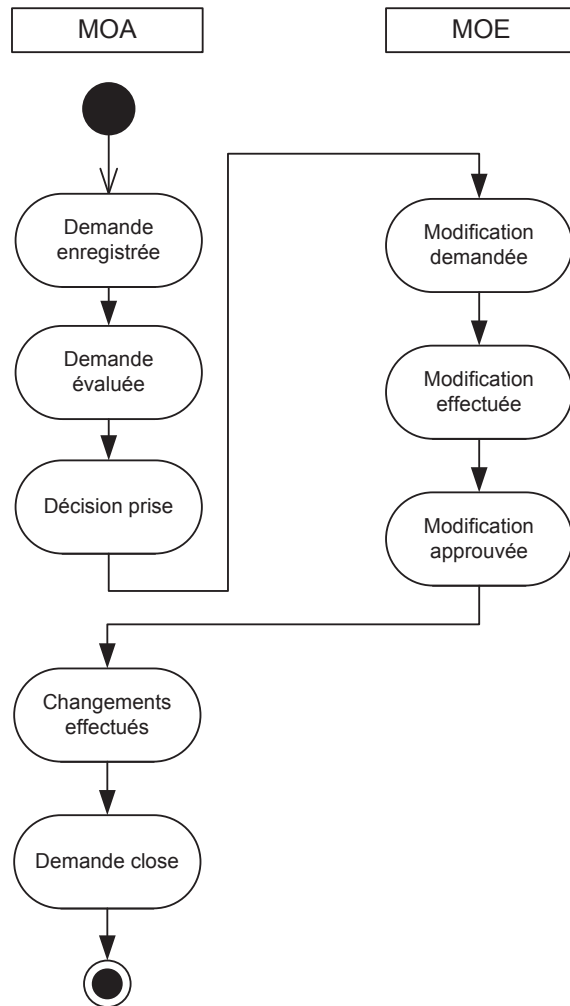


Figure 18-2 : Diagramme d'état des demandes de changement

Une demande de changement pourra donner lieu à plusieurs ordres de modification (en effet, une demande peut impacter plusieurs composants du logiciel et faire appel à des expertises diverses).

Toute demande de changement va passer par des états successifs :

- proposée : enregistrée, soit sur une fiche prévue à cet effet, soit au moyen d'un outil de gestion des changements ou d'un outil de gestion des exigences ;
- approuvée par la commission de contrôle des changements, elle a donné un ou plusieurs ordres de modification ;

- modifiée : les modifications correspondant à la demande ont été prises en compte ;
- vérifiée : les tests correspondant à la demande de changement ont été effectués ;
- supprimée ou rejetée : la demande peut avoir été retirée par le demandeur ou rejetée (avec justification) par la commission.

### Une discipline nécessaire et rentable

---

Le processus de gestion des changements des exigences que nous venons de décrire est complexe et demande de la rigueur et de la discipline : procédures, commission, fiches à remplir, analyses d'impact, suivi et traçabilité. Ces opérations ne sont pas si consommatrices de temps que l'on pourrait penser. Et contrairement à ce que l'on pense souvent, elles ne sont pas un frein à la créativité des développeurs, elles ne servent qu'à canaliser cette créativité.

Mettre en place cette « bureaucratie de projet » nécessite un changement culturel. La mise en place et l'institutionnalisation d'un tel processus peuvent prendre plusieurs mois. Il faudra surtout éviter de s'embarquer dans des processus « usines à gaz », disproportionnés à la taille des projets et au type de produit développé. En revanche, cette discipline, même légère, devra être incontournable.

Cette discipline permet de préserver une grande partie de l'effort consenti en amont, en phase de développement des exigences. La cohérence des exigences, et donc du produit livré, est à ce prix. Remarquons que cet effort de gestion sera de toute façon nécessaire aux autres activités de développement, en particulier :

- évaluation précise des charges ;
- gestion optimale des ressources (développeurs, testeurs) ;
- alignement entre produit, spécification et documentation.



## Chapitre 19

# Les outils

À partir d'une certaine complexité, le besoin de mettre en œuvre des outils de définition ou de gestion des exigences se fait sentir. Utilisés à bon escient par des personnes compétentes, ils vont contribuer efficacement au processus. Mais attention, il y a aussi de fausses raisons de les utiliser et des pièges à éviter.

### **Check-list : votre projet en a-t-il besoin ?**

Comme nous l'avons vu, pour gérer les exigences et leurs évolutions, il est nécessaire de tenir à jour un certain nombre d'attributs pour chacune d'elles. Le besoin est fonction de l'importance du projet, c'est-à-dire du périmètre fonctionnel, de la taille des équipes et des besoins de traçabilité exigés. Sur un projet d'importance, la gestion d'informations telles que la date de création ou de modification d'une exigence et ses liens de traçabilité deviennent un vrai casse-tête. Il est difficile, voire impossible, de gérer toutes ces informations à la main.

L'utilité des outils de gestion des exigences est fonction, entre autres :

- du nombre d'exigences à traiter, qui dépend du périmètre fonctionnel et du niveau de détail ;
- du nombre d'attributs à gérer pour chaque exigence, qui dépend de la rigueur demandée au processus ;
- du nombre de personnes en charge de recueillir, analyser, spécifier, modifier les exigences ;
- du temps d'historisation prévu pour ces données, de la pérennité dans le temps des exigences ;
- du degré de réutilisation des exigences d'un projet à l'autre ;
- de la volatilité des exigences (nécessité de les modifier souvent) ;

- du nombre de règles de gestion dont dépendent ces exigences ;
- de la possibilité d'investir du temps à se former aux outils.

Cette liste peut servir de check-list pour estimer grossièrement (mais rapidement) le besoin d'outiller.

## Les bonnes raisons d'utiliser les outils

---

La première raison d'acquérir un outil est liée au processus de définition et de gestion des exigences elles-mêmes. Un bon outil facilite l'application des bonnes pratiques de développement (utilisation d'un modèle, mise en forme du texte) et de gestion des exigences (traçabilité des changements, gestion du cycle de vie des modifications).

La seconde raison d'utiliser les outils concerne la gestion des changements dans leur ensemble. Dans un monde où tout va très vite et où les changements sont constants, les exigences sont rarement stables. Cela est vrai qu'il s'agisse des besoins (émanant des clients) ou des contraintes (liées aux technologies), et même des règles de gestion. Tout changement dans les exigences aura des répercussions sur l'ensemble des autres activités : conception, réalisation, tests, maintenance.

La troisième raison est la conformité. Les changements eux-mêmes devront être tracés. En effet, de nos jours, les entreprises sont soumises à des contrôles de plus en plus nombreux : audits, certifications (ISO ou autres), évaluation des pratiques professionnelles, accréditations, normes et standards de conformité, de traçabilité, de sécurité... le logiciel lui-même n'échappe pas, bien sûr, à ces obligations : sécurité, interopérabilité, conformité à des règles de gestion de toute sorte font partie du lot quotidien des analystes et développeurs. On se dit que bientôt, la forêt de la conformité va cacher l'arbre de la fonctionnalité. Quoi qu'il en soit, il faudra faire avec.

## Fonctions de base et fonctions avancées

---

### Fonctions de référentiel

Un outil de gestion des exigences est avant tout un référentiel. Il permet de stocker, et de retrouver facilement, l'ensemble des exigences, leurs attributs et leurs liens de traçabilité. Il arrive avec un certain nombre de fonctions de création, de retrait, de modification, et de suppression des exigences ou de leurs attributs. Il permet de contrôler l'accès à ces informations, d'où son utilité pour le travail de groupe.



- **Stockage.** Les exigences sont stockées, généralement dans une structure arborescente, sous une forme indépendante du document final de spécifications (cahier des charges). Le référentiel stocke, en plus des attributs, l'état de chaque exigence (proposée, en cours, validée, etc.).
- **Filtrage.** Il est possible d'utiliser des filtres pour visualiser une partie seulement de l'arborescence.
- **Traçabilité des accès.** L'outil trace tous les accès à une exigence ou à un attribut.
- **Gestion des liens entre exigences,** et possibilité de suivre les liens de traçabilité horizontale ou verticale.

## Traçabilité horizontale et verticale

Les exigences sont reliées entre elles de plusieurs façons. On dénombre au moins deux types de traçabilité :

- La traçabilité *verticale* lie les exigences et leur évolution au long du cycle de vie du logiciel, depuis les exigences de haut niveau jusqu'aux composants du logiciel.
- La traçabilité *horizontale* lie les versions successives d'une même exigence.

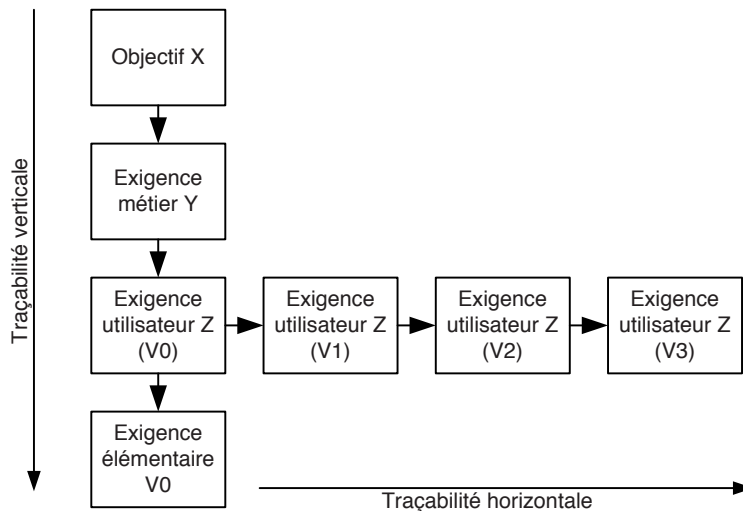


Figure 19-1 : Traçabilité horizontale et verticale

Ces deux types de lien de traçabilité sont indispensables si l'on veut contrôler et gérer à long terme les demandes de changements. Or, au-delà d'une certaine taille de projet, le maintien de ces liens de traçabilité est quasiment impossible sans la mise en œuvre d'outils adéquats.

### Gestion des attributs

Seul un outil permettra de renseigner facilement tous ces attributs, dont certains (date, auteur, etc.) de manière automatique. Avec un document texte, cet exercice est un véritable casse-tête. L'utilisation d'un tableur est une solution de contournement, mais ne permet pas de stocker (ou très difficilement) les exigences qui se présentent sous forme de tableau ou de graphique. La plupart des outils permettent à un administrateur de paramétrer les attributs et de créer des types d'attributs différents en fonction du type d'exigence. Tout utilisateur pourra visualiser les attributs de chaque exigence, et certains utilisateurs pourront modifier des attributs en fonction de leur habilitation.

#### Une multitude d'attributs

Un très grand nombre d'attributs peuvent être accolés à chaque exigence. En voici quelques-uns :

- date de création ;
- version actuelle ;
- auteur (le rédacteur) ;
- responsable de la mise en œuvre ;
- demandeur (pas nécessairement utilisateur du produit) ;
- bénéficiaire (en général, un profil utilisateur) ;
- priorité ;
- justification de l'exigence ;
- version du produit où l'exigence sera mise en œuvre ;
- statut (provisoire, en cours d'étude, validée) ;
- volatilité (exigence provisoire ou pérenne) ;
- tests à faire pour la vérifier ;
- seuils de tolérance (pour les exigences non fonctionnelles) ;
- traçabilité horizontale et verticale.

S'y ajoutent les attributs concernant la modification apportée à une exigence : date, auteur, demandeur, justification, etc.

### Travail d'équipe

Ces fonctions favorisent un travail d'équipe sûr et efficace, qui est pénible et risqué (voire impossible) lorsque l'on partage un document sous traitement de texte.

- **Traçabilité.** L'outil garde la trace de toutes les modifications faites sur une exigence ou un attribut, ainsi que de la date et l'auteur de chaque modification.

- **Contrôle d'accès.** L'outil gère les accès aux exigences et aux différents attributs, en fonction de droits d'accès paramétrables.
- **Sécurisation des modifications.** L'outil bloque l'accès en écriture à des exigences en cours de modification, n'autorisant que les accès en lecture.
- **Communication de projet.** L'outil envoie un message aux différents utilisateurs lorsqu'une exigence ou un ensemble d'exigences viennent d'être modifiées. Il garde la trace d'une discussion concernant une exigence particulière.
- **Gestion de projet.** L'outil conserve l'état de chaque exigence (en cours, à valider, à modifier...). Cela facilite le suivi.

## Gestion des versions et des changements

Ces fonctions rendent ce type d'outil proche des outils de gestion des versions et de la configuration (ils peuvent d'ailleurs s'interfacer avec certains d'entre eux).

- **Vernionnement.** L'outil garde la trace des changements faits sur une exigence et incrémente son numéro de version.
- **Mise en référence (*baselining*).** L'outil permet de figer une configuration de référence (*baseline*) et de créer de nouvelles versions d'une spécification d'exigences en s'appuyant sur une configuration de référence existante.
- **Automatisation du processus.** Certains outils facilitent le processus de gestion des changements d'exigences en liaison avec la gestion des versions des exigences.

## Rédaction et documentation

On constate donc qu'un outil de gestion des exigences est beaucoup plus intéressant qu'un traitement de texte. Il n'empêche qu'une large partie du travail consiste à rédiger et que le livrable final est souvent un texte avec toutes les contraintes de rédaction et de typographie en vigueur. L'outil doit donc être capable de faciliter la rédaction et la présentation documentaire. Entre autres fonctions, on trouve les suivantes :

- **Respect du modèle de spécifications.** Il est possible de paramétrer la structure des exigences selon un modèle prédéfini. On force ainsi le respect d'un modèle de cahier des charges.
- **Glossaire.** L'outil permet d'alimenter et maintenir un glossaire des termes, et chercher les occurrences d'un terme du glossaire.
- **Respect des règles d'écriture.** Outre le classique correcteur orthographique, l'outil possède des fonctions de recherche contextuelle de termes flous, de phrases ambiguës, de mots inconnus, de certaines incohérences.

- **Génération automatique de documents.** À partir d'un modèle de document prédéfini, du contenu du référentiel et d'un ensemble de filtres, l'outil génère automatiquement le cahier des charges ou tout autre document de spécifications.
- **Intégration de graphiques.** L'outil permet de créer ou d'intégrer des exigences sous forme graphique : scénarios, modèles de données, etc.
- **Import de documents.** L'outil permet d'intégrer des exigences contenues dans un cahier des charges dans le référentiel, en fonction de règles paramétrables ; par exemple, niveau de titre ou style de paragraphe correspondant à un niveau ou type d'exigence.

### Attention aux mirages

---

On le voit, les avantages de ces outils sont certains. Prenons garde, néanmoins, à ne pas tomber dans le piège des outils. À lui seul, aucun outil ne se substituera à une bonne organisation et à une discipline rigoureuse. Si les exigences sont mal gérées, un outil n'y fera rien. Il pourra même avoir un effet contre-productif.

Citons quelques-unes des fausses bonnes raisons d'acquérir un outil :

- « Il gère mieux les spécifications et la documentation des exigences ». C'est faux. La rédaction du cahier des charges se fait par des personnes compétentes. Un traitement de texte pourra les aider, bien mieux qu'un outil de gestion des exigences.
- « Il gère mieux les changements des spécifications et de la documentation ». C'est vrai, mais partiellement. La gestion des changements est une affaire de processus et de personnes. En termes d'outils, un tableur pourra faire l'affaire. Ce n'est que sur la combinatoire gestion des changements *et* gestion de la documentation qu'un outil apportera sa pleine puissance, car il permet de lier automatiquement les deux.
- « Il facilite le travail de vérification et de validation ». Ce n'est que très partiellement vrai. Effectivement, certains outils disposent de fonctions de recherche en plein texte et de détection de phrases douteuses, mais le gros du travail est intellectuel et ne peut être pris en charge par quelque outil que ce soit.

### Quand et comment choisir un outil ?

---

Inutile de choisir un outil si l'on n'a pas, déjà, un modèle de document à peu près stable et un processus rigoureux de définition et de gestion des exigences. Comme nous l'avons dit plus haut, un bon outil ne peut pallier

un processus défaillant, il ne peut qu'empirer la situation. De plus, il vaut mieux choisir un outil en fonction du processus que l'inverse.

Un outil de gestion des exigences doit être sélectionné comme tout logiciel :

- en analysant l'existant et le contexte de l'organisation ;
- en précisant les objectifs (par exemple, améliorer la productivité, sécuriser les accès, automatiser la production de documents, réutiliser les exigences) ;
- en recueillant les besoins, en les analysant, en les spécifiant dans un cahier des charges ;
- en préparant une grille de sélection efficace (on s'inspirera des fonctions décrites plus haut) ;
- en étant rigoureux jusqu'au bout.

Pour un analyste des exigences, ces actions semblent aller de soi, mais l'expérience montre qu'il est plus difficile d'être rigoureux quand on travaille pour sa propre organisation (pour savoir pourquoi, analysons les parties prenantes : celui qui finance mon outil est aussi celui qui me paye à la fin du mois).

Une fois l'outil sélectionné, acquis, installé, il faudra le traiter comme tout autre logiciel. En particulier, il faudra se former à l'outil, être conseillé par l'éditeur (attention : la conduite du changement coûte généralement beaucoup plus cher que l'outil lui-même), et travailler sur le paramétrage.

D'une manière générale, il faut éviter de se précipiter, et avoir un processus bien rôdé avant de « l'embarquer » dans l'outil. Ce que l'on ne peut pas accomplir avec un papier et un crayon, inutile de tenter de le faire avec l'outil, car sa complexité peut perturber. A contrario, si l'on s'est « pris la tête » à de nombreuses reprises, avec un traitement de texte, l'arrivée de l'outil sera vécue comme un soulagement et ses bénéfices vite appréciés.



# Au-delà des exigences

Le travail de l'analyste consiste à spécifier les exigences, mais aussi à évaluer les solutions proposées, et leur adéquation aux exigences spécifiées. Pour y parvenir, nous pouvons utiliser diverses techniques, en complément ou à la place de l'élaboration d'un cahier des charges. Nous sommes ici aux confins de la phase d'exigences, et pénétrons parfois dans la zone floue entre exigences et conception.

### Étude de choix

---

Pour une étude comparative ou une étude de choix, la démarche parfois appelée « en entonnoir » (figure 20-1) est la plus classique : elle permet d'aller du général au particulier et du global au détaillé.

#### Les étapes

Les étapes de la démarche sont les suivantes :

- **Screening.** Une dizaine de fournisseurs de logiciels, dont les produits correspondent, dans les grandes lignes, aux besoins du client, sont identifiés, et leurs produits sont passés en revue. On peut inclure dans cet ensemble des logiciels qui ne présentent pas toutes les caractéristiques requises, mais dont l'examen peut apporter des informations intéressantes, ou révéler de nouveaux critères de choix.
- **Présélection.** En collaboration avec le client, nous examinons les cinq logiciels qui correspondent au mieux aux critères définis par le client ; parallèlement, la liste des critères est affinée. On détermine les critères obligatoires et les critères éliminatoires.
- **Short-list.** Toujours en collaboration avec le client, une liste restreinte (généralement trois logiciels) est fixée ; ces logiciels sont examinés de

près ; parallèlement, et indépendamment, les critères sont pondérés lors d'une réunion avec les responsables, chez le client.

- **Approfondissement.** Les trois logiciels « short-listés » sont examinés de près. Une note est attribuée à chaque critère. On aboutit, selon la demande du client, soit à la sélection d'un produit unique, soit à un tableau comparatif permettant une prise de décision efficace.

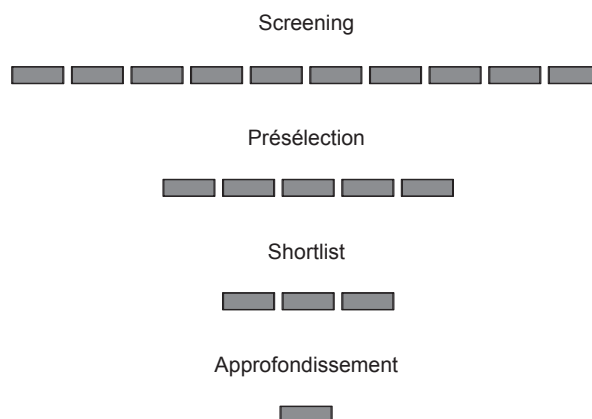


Figure 20-1 : La méthode de l'entonnoir

## Étude comparative complexe

Il y a plusieurs façons de traiter les réponses à un cahier des charges. On peut procéder à une analyse comparative tenant compte de contraintes multiples, non seulement fonctionnelles, techniques et organisationnelles, mais également stratégiques ou politiques. Une fois l'étude faite, le challenge est de donner à un décideur de haut niveau, non spécialiste, une vision très claire des potentialités des différentes solutions en présence, et ce, souvent en moins de cinq minutes.

L'étude se fera comme précédemment, et inclura des éléments supplémentaires d'expertise sur des points particuliers (intégrabilité, déploiement, conduite du changement, etc.). La présentation des résultats devra être « percutante ». Pour cela, on devra fournir aux décideurs des tableaux et des schémas extrêmement simples, contenant le minimum d'informations. L'information plus détaillée sera jointe si nécessaire en annexe.

### Tableau de B.O.R.D.

Le tableau de B.O.R.D. (Bénéfices, Opportunités, Risques, Dangers) est une variante du tableau SWOT (*Strengths, Weaknesses, Opportunities, Threats*,



c'est-à-dire forces, faiblesses, opportunités et menaces) en plus contrasté. L'idée est de présenter un tableau très synthétique, à l'usage des décideurs. Ce tableau indique, pour chaque solution présentée, les :

- **Bénéfices** : avantages procurés de façon certaine ou très probable par l'acquisition du produit, à court ou moyen terme, sans investissement, ou avec un investissement faible.
- **Opportunités** : avantages apportés par le produit, au prix d'un effort financier ou organisationnel raisonnable.
- **Risques** : difficultés pouvant surgir, avec une probabilité faible ou moyenne, suite à la mise en place du produit, et pouvant être surmontées avec un investissement financier ou organisationnel raisonnable.
- **Dangers** : difficultés lourdes de conséquences, ayant une forte probabilité de surgir suite à la mise en place du produit, et ne pouvant être jugulées que par un effort financier ou organisationnel important.

Dans l'exemple<sup>1</sup> donné au tableau 20-1, la solution Y a été choisie, car elle apporte un véritable avantage concurrentiel sans risques insurmontables. La solution X semble moyennement bien placée, la solution Z risquée. Cependant, le choix revient au décideur, qui aurait pu en décider autrement.

1. L'exemple présenté ici correspond à un cas réel (outil de *workflow* pour le système d'information administratif dans l'industrie automobile).

**Tableau 20-1 – Tableau de B.O.R.D**

	<b>Solution X</b>	<b>Solution Y</b>	<b>Solution Z</b>
<b>B</b>		Expérience dans au moins un projet d'envergure et de nature similaires. Excellente capacité de conseil et de gestion de projet.	
<b>O</b>	Possibilité pour le client de développer sa propre solution.	Solution ouverte, en particulier extensible ( <i>workflow</i> ). Possibilité de tirer profit de l'expérience chez un autre client.	Possibilité de tirer profit de l'expérience de l'entreprise « W ».
<b>R</b>	Manque d'expérience du fournisseur sur des projets d'envergure similaire.	Produit complexe, exigeant une bonne formation et un bon suivi de projet.	Solution fermée, difficilement extensible ; négociations techniques difficiles.
<b>D</b>			Solution très contraignante pour l'utilisateur final

## Graphique de synthèse

Le graphique de la figure 20-2 fait la synthèse des éléments mentionnés dans le texte du rapport et les tableaux, et permet de comparer les différentes solutions proposées en mettant en balance les bénéfices apportés et les difficultés pressenties. Il permet de comparer :

- les bénéfices, qui se décomposent en :
  - réponse de l'offre aux exigences du cahier des charges,
  - potentiel d'évolution de la solution proposée,
  - capacité de service du fournisseur : conseil, support, et assistance technique ;
- et les coûts pour le client, qui se décomposent en :
  - prix d'achat et coût de la mise en œuvre,
  - coûts de la phase transitoire (baisse initiale de la productivité du personnel),
  - conduite du changement (formation, information et accompagnement).

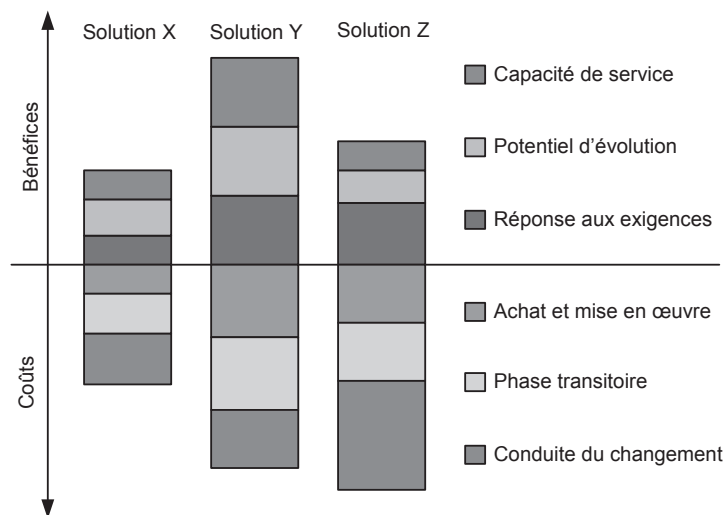


Figure 20-2 : Graphique comparatif de synthèse coûts/bénéfices

## Étude d'opportunité complexe

### Un exemple d'étude

Une étude d'opportunité peut être très complexe et faire appel à des compétences et des savoir-faire très divers. Dans l'exemple qui suit, nous

allons décrire une étude qu'un cabinet de conseil a effectuée pour une grande ville française. Il s'agissait d'étudier l'opportunité de remplacer toute la bureautique existante par du logiciel libre. L'étude a mobilisé deux consultants seniors (un consultant maîtrise d'ouvrage et un expert technique) pendant trois mois, avec une charge totale de 50 jours-hommes.

- Étape 1 : lancement.

Il s'agit ici de définir le périmètre de la mission, d'identifier les acteurs (très nombreux), et de figer le plan détaillé des livrables

- Étape 2 : étude de la stratégie d'évolution.

L'objectif est de définir l'ordre et la stratégie de mise en place d'un projet de passage au logiciel libre dans l'administration de la ville.

- Étape 3 : état des lieux fonctionnel et technique sur l'existant.

Elle consiste en une consultation de l'Administration (France) au moyen d'interviews des responsables informatiques (développement, maintenance et exploitation).

- Étape 4 : étude de l'expérience dans une autre ville européenne.

L'objectif de cette étape est d'avoir un retour d'expérience dans une administration analogue, dans un autre pays européen, et de connaître les avantages, inconvénients, coûts, contraintes, difficultés rencontrées suite au passage au logiciel libre.

- Étape 5 : étude des besoins des décideurs et utilisateurs.

Ce n'est qu'à cette étape qu'intervient l'étude des besoins, à très grosses mailles. L'objectif n'est pas de définir des besoins fonctionnels, mais d'avoir un avis des décideurs et futurs utilisateurs. Une majorité d'entretiens a été faite par téléphone.

- Étape 6 : étude des produits disponibles et de leur utilisation.

L'objectif est ici de connaître les contraintes techniques inhérentes au passage au logiciel libre, les avantages, inconvénients, le potentiel de ces outils.

L'étape a consisté à examiner les différents types de produits logiciels libres sur le marché (réseaux, systèmes d'exploitation, applicatifs...). On a essentiellement examiné l'utilisation qui en était faite. Cela a nécessité de rencontrer à la fois les utilisateurs et les industriels. À cette étape, les aspects techniques étaient secondaires.

- Étape 7 : étude des contraintes techniques.

On a examiné les contraintes techniques sur le matériel et le logiciel, et analysé l'impact lié au renouvellement du parc informatique (mises à niveau, migration ou portage des applications, impacts sur la sécurité).

L'analyse a permis de mettre en regard les contraintes et les produits disponibles.

- Étape 8 : étude des aspects financiers.

On a examiné les aspects financiers sur plusieurs critères : utilisation, mise en œuvre, conversion de fichiers, coûts des émulateurs (pour les applications qui n'ont pas d'équivalent dans le monde du logiciel libre), mise en œuvre de la sécurité, migration des applicatifs et formation.

- Étape 9 : synthèse tenant compte des aspects techniques, organisationnels et financiers.

Cette synthèse comprend : les scénarios cibles de passage, l'analyse technique, l'analyse économique, l'analyse des risques. Elle est formalisée par un tableau de synthèse pour l'estimation du coût de la migration, et un tableau de bord avantages et inconvénients pour chaque scénario.

### Une mission à la limite

Une mission de conseil de ce type est à la frontière entre la définition des besoins et une étude préalable à un schéma directeur. Dans cette mission, l'étude des besoins proprement dite, au sens de l'ingénierie des exigences, n'occupe qu'une place modeste.

L'étude a été menée par un consultant senior et un expert technique. Les très forts enjeux politiques ont constitué une difficulté supplémentaire. Cependant, si le consultant sénior avait eu le réflexe de mener une analyse formalisée des parties prenantes, certains écueils auraient probablement été évités.

Une autre difficulté provient de la dichotomie opérée entre les aspects techniques et les autres, accentuée par le fait que les consultants étaient de profils très différents. Tout au long de cet ouvrage, nous avons pu voir que les exigences pouvaient être structurées en « couches », depuis les exigences stratégiques (en l'occurrence, politiques) jusqu'aux exigences fonctionnelles et techniques. Les consultants ont examiné ces deux aspects extrêmes, sans consacrer le temps nécessaire à l'examen des exigences de niveaux intermédiaires (exigences métier et exigences utilisateurs).

### Étude d'intégrabilité et design

---

Tout au long de cet ouvrage, nous avons répété cette recommandation à la manière d'un dogme : il faut spécifier des besoins, et non une solution. Cela reste toujours vrai lorsque l'on élabore un cahier des charges.

Ce n'est qu'à la phase suivante, la phase de conception, que l'on spécifie la solution.

Or, la frontière entre définition des besoins et conception de la solution n'est pas toujours étanche. Prenons un cas fréquent : nous avons élaboré un cahier des charges pour une application de gestion complexe. Plus précisément, nous avons défini les exigences pour un système d'information (sans même parler d'application, puisque nous cherchons une solution). Ce cahier des charges, nous l'avons élaboré dans les règles de l'art, dans le plus strict respect de la méthodologie. Puis nous avons élaboré une grille de dépouillement qui reprend point par point chaque exigence.

Cependant, les offres des fournisseurs ressemblent à ce que l'on peut voir dans la figure 20-3 : malgré leurs qualités respectives, aucun produit ne correspond parfaitement à la demande. Cependant, le besoin peut être couvert par une intégration entre deux des logiciels, ou les trois logiciels.

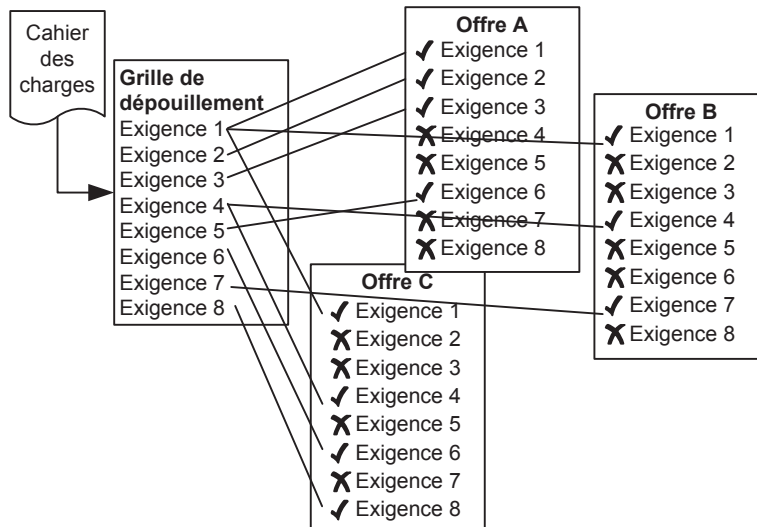


Figure 20-3 : Des exigences à l'architecture de la solution

Nous venons de franchir la frontière qui sépare les exigences de la conception. Nous sommes face à un problème de design, d'architecture du système d'information. Nous devons faire un choix entre trois scénarios :

- choisir un produit parmi les trois et faire développer en spécifique les fonctions manquantes ;
- faire développer une interface entre les deux produits ;
- faire intégrer les trois produits.

L'étude de l'intégration entre différentes « briques » du système d'information doit se faire non seulement sur le plan technique, mais aussi sur le plan économique, et aussi sur le plan de l'ergonomie. Deux produits peuvent très bien communiquer entre eux sur le plan technique et aboutir à une catastrophe en ce qui concerne l'utilisabilité, la fiabilité, et bien sûr la maintenabilité de la solution.

Après étude, nous devons reformuler notre demande vis-à-vis d'un, de deux ou de trois fournisseurs, et donc élaborer un nouveau cahier des charges. Et nous voilà de nouveau passés de l'autre côté de la frontière, à nouveau dans la phase d'exigences. Voilà pourquoi on dit que la frontière entre la phase d'exigences et la phase de conception est souvent floue.

## Chapitre 21

# Neuf conseils

Nous avons décrit le processus, la démarche d'élaboration, les étapes, techniques et outils. Nous avons montré pourquoi et comment le processus peut être adapté aux différents contextes, et pourquoi et comment il peut être amélioré. Voici quelques conseils pour réussir encore mieux, plus efficacement.

### 1. Ayez toujours l'objectif en tête

En principe, vous connaissez l'objectif de votre client, et si ce n'est pas le cas, votre première tâche consiste à le découvrir et le définir avec précision. Votre objectif est aligné sur celui de votre client, mais il n'est pas identique à celui du client.

Votre objectif à vous, analyste des exigences, est de livrer à votre client, à l'heure convenue avec lui, un cahier des charges élaboré selon les règles de l'art. Vous pouvez affiner cette formulation et la mettre par écrit. Il faudra la garder en tête et avancer. Vous devez tenir le cap contre vents et marées.

Il est parfaitement normal que le client lui-même retarde le projet, change d'avis, en demande trop ou au contraire n'exprime rien. Il est normal que les idées divergent d'un acteur à l'autre, que la motivation passe par des hauts et des bas, que l'analyste devienne l'exutoire de tous les conflits. Pour filer la métaphore de la navigation, c'est lorsque le bateau tangue qu'il est utile de savoir clairement où l'on va. Vous devez garder le cap malgré les dérives.

Cette vision claire de l'objectif vous permettra de planifier l'élaboration du cahier des charges, et guidera vos décisions lors de vos interventions sur le terrain.

## 2. Analysez et validez au plus tôt

---

Dans les chapitres précédents, nous avons indiqué plusieurs raisons d'analyser et de valider au plus tôt, et nous avons donné quelques techniques pour y parvenir. Mais il y a aussi des raisons psychologiques, qui sont complémentaires aux raisons techniques. L'analyse au plus tôt apporte une visibilité qui permet de mieux maîtriser la situation et encourage les différents participants à travailler ensemble. La validation au plus tôt donne un feedback rapide et une vision claire qui encouragent à aller de l'avant.

## 3. Lancez-vous un défi et donnez-vous les moyens de le réussir

---

La définition des besoins ne devrait jamais être une routine, car deux clients n'ont jamais le même objectif, les mêmes besoins, le même passé, les mêmes acteurs. Élaborer un cahier des charges est un travail difficile, même pour les plus doués et les plus expérimentés d'entre nous.

Certains consultants font du travail à la chaîne, en brûlant les étapes. Ils « vendent » à leur client un cahier des charges préfabriqué, en spécifiant de manière approximative des exigences validées en vitesse par une minorité d'acteurs. Malgré les apparences, la valeur ajoutée d'un tel document est faible. La satisfaction professionnelle, elle aussi, est faible.

D'autres baissent les bras devant les difficultés. Ils pensaient que la tâche serait facile. Elle ne l'est qu'en apparence. Les vraies difficultés sont humaines et concernent la résolution de conflits et la gestion des priorités. La tentation est grande alors de faire de l'analyse des besoins une tâche purement technique, et de court-circuiter certaines parties prenantes. Au premier abord, la qualité du livrable semble satisfaisante. L'ensemble est en apparence correct et complet. Mais le véritable besoin n'a pas été cerné.

Le challenge est de concilier la rigueur et la souplesse, et de fournir un livrable dans un temps court et au moindre coût, sans que cela se fasse au détriment de la qualité. Pour y arriver, il faut utiliser la bonne démarche et les bonnes techniques, s'y entraîner encore et encore, et aussi célébrer les réussites.



## 4. Conciliez concepts et action de terrain

---

Supposons que vous ayez lu un ou plusieurs ouvrages sur l'ingénierie des exigences ou l'élaboration du cahier des charges pour le logiciel. Vous avez appris une démarche, des concepts, sans rien mettre en pratique. Supposons maintenant qu'après cela, vous partez sur le terrain avec un consultant aguerri pour élaborer un cahier des charges. Il est fort probable que vous constatiez un fort décalage entre ce que vous avez lu et ce que vous voyez et entendez. Vous commencez à avoir des doutes. Ce que vous avez lu est-il pure théorie, ou bien est-ce le consultant qui s'y prend mal ?

La vérité doit se situer entre les deux. Sans doute le consultant utilise sa méthode, qui a fait ses preuves, sans chercher à l'optimiser outre mesure. Peut-être a-t-il pris quelques mauvais plis en même temps qu'il a pris de l'assurance vis-à-vis de ses clients. Mais le plus probable est qu'il utilise de nombreuses techniques décrites dans cet ouvrage de manière très souple. Pour un petit projet, les objectifs peuvent être décrits en quelques lignes et apparaître dans le compte-rendu d'une réunion de lancement. De même, l'analyse des parties prenantes peut être faite rapidement et son résultat tenir en quelques lignes.

Lorsqu'on est débutant, il vaut mieux aller lentement, formaliser et tracer dans la mesure du possible, pour des raisons pédagogiques et pour sécuriser leur relation avec le client. Lorsque les bonnes pratiques sont devenues des réflexes, on peut prendre de l'assurance et de la vitesse. Dans le feu de l'action, quand le processus est optimisé, les concepts et la pratique sont une seule et même chose.

## 5. Concentrez-vous sur votre livrable

---

Le désir d'atteindre un livrable de qualité est le meilleur moteur de l'action. En pratique, cela signifie que les livrables intermédiaires, c'est-à-dire les versions intermédiaires du cahier des charges, même « en chantier » doivent, toutes proportions gardées, être irréprochables. Un document navette n'est pas un document poubelle. Les parties à compléter, à retravailler, à valider doivent être clairement signalées au moyen de conventions connues de tous. Si des « paquets d'exigences », en provenance de sources diverses, doivent être intégrés au document intermédiaire, cela doit se faire selon des règles strictes, mais simples à mettre en œuvre.

Avec cette manière de procéder, l'équipe d'analystes et leurs clients ont à leur disposition un document toujours présentable, sur lequel on peut discuter, qui demandera le moins de remaniement possible. Dans le cas contraire, l'équipe passera son temps à bricoler et à rafistoler, ce qui consomme beaucoup de temps et d'énergie.

Sur le plan psychologique, cette méthode présente aussi de nombreux avantages. On a plaisir à travailler sur un document bien clair, on voit le travail avancer, et cela tire tous les participants dans une spirale vertueuse. Dans le cas contraire, c'est l'énervement permanent.

### 6. Sachez réussir presque à coup sûr

---

Savoir que l'on ne court aucun risque, ou presque, de rater un cahier des charges, est très sécurisant et permet de travailler dans de bonnes conditions. Encore faut-il savoir s'y prendre.

Il y a un secret pour réussir de cette manière : la mission d'élaboration d'un cahier des charges doit être structurée de telle sorte que, quoi qu'il arrive, elle ne soit jamais un échec complet, et toujours une réussite ou une quasi-réussite. Mais comment faire ?

D'après la « loi » de Pareto, 20 % de l'effort d'élaboration permettra d'atteindre 80 % du résultat escompté. En d'autres termes, si tout se met à aller mal alors que vous n'en êtes qu'à un cinquième du parcours, le cahier des charges, lui, sera prêt aux quatre cinquièmes.

La loi de Pareto se vérifie si l'on choisit le bon modèle de document, les bonnes méthodes, le bon rythme, et qu'on travaille de manière méthodique pour faire les choses dans le bon ordre. Il n'y a pas de règle absolue pour savoir quel est le bon ordre d'élaboration, mais le plus souvent, il vaut mieux travailler de manière descendante, par raffinements successifs, depuis les objectifs jusqu'aux détails les plus fins.

Dans le cas contraire, où nous sommes face à un existant, (par exemple une spécification existante), le bon ordre consiste à consolider cet existant en le (re)formalisant, puis à remonter jusqu'aux objectifs, puis à redescendre à nouveau dans les détails de plus en plus fins.

La loi de Pareto s'applique également en ce qui concerne le champ de l'étude, les objectifs et les parties prenantes. D'où l'intérêt de bien les connaître dès le début, et de connaître les priorités. Si on ne peut pas tout faire ou satisfaire tout le monde, il vaut mieux savoir qui satisfaire en premier et quels sont ses objectifs prioritaires.

### 7. Mettez en avant votre client

---

C'est votre client, la maîtrise d'ouvrage, qui élabore le cahier des charges, pas vous. Effacez-vous devant lui et valorisez son travail. Vous êtes animateur, facilitateur, expert, mais c'est votre client qui exprime un besoin, et il l'exprime à sa maîtrise d'œuvre. Se mettre en avant n'est ni fair-play, ni efficace. Lorsque vous parlez au maître d'œuvre, vous êtes le porte-parole

du maître d'ouvrage, vous ne faites que traduire ses besoins. Lorsque vous faites une présentation brillante en comité de pilotage, rappelez-vous : votre présentation est une *représentation* des besoins de votre client.

Lorsque le cahier des charges sera livré, si tout se passe bien, si votre client est généreux et bon joueur, il reconnaîtra peut-être que vous avez fait du bon travail et vous remerciera pour le service que vous lui aurez apporté. Cela fait très plaisir et encourage à faire encore mieux la prochaine fois. Mais au cours de l'élaboration, votre ego doit rester au vestiaire.

## 8. Perdez un peu de temps pour en gagner beaucoup

---

Le temps « perdu » n'est pas toujours celui que l'on croit. Faire l'impasse sur le recueil d'un objectif, laisser de côté une partie prenante pour, soi-disant, gagner du temps, est purement et simplement un manque de professionnalisme. Bien sûr, votre hiérarchie, votre client, le maître d'œuvre ou toute autre partie prenante peut faire pression pour vous pousser à aller plus vite. Il est de votre responsabilité d'expliquer à vos interlocuteurs qu'il faut consolider les fondations avant de s'attaquer aux murs, et que le temps investi sera récupéré au décuple.

Il ne faut donc pas hésiter à consacrer du temps aux étapes en amont de l'élaboration : définition de l'objectif, des parties prenantes et du champ de l'étude. Si un de ces points est mal défini, il faudra après coup défaire le travail et le refaire. Et parfois, tout reprendre du début. Cela occasionne beaucoup de temps perdu, de fatigue inutile, de stress et de découragement.

Ce qui est vrai pour le macroprocessus d'élaboration l'est aussi au niveau des microprocessus. Le temps passé à préparer une interview ou une réunion n'est pas du temps perdu. Parfois, il vaut mieux passer beaucoup de temps à discuter autour d'un schéma que d'avancer. C'est l'expérience qui permet de distinguer l'investissement en temps, de la perte de temps pure et simple. Aussi, est-il important d'observer sa propre manière de procéder et d'essayer de s'améliorer en permanence.

## 9. Faites de la définition des besoins un projet en soi

---

Définir les besoins est un vrai métier, avec ses méthodes, ses techniques, ses outils, avec des experts et un savoir-faire propre. Les analystes ont leur propre rythme de travail, leurs règles et leur code de bonne conduite,

qui diffèrent de ceux des réalisateurs ou des testeurs. Voilà pourquoi il faut considérer l'élaboration d'un cahier des charges comme un projet en soi, avec son livrable bien défini. De plus, avoir une vision globale de la mission de définition des besoins est beaucoup plus valorisant et pousse à l'action.

Considérer la définition des besoins seulement comme un sous-projet, et non un projet en soi, n'est pas très motivant. Vous avez acquis un savoir-faire, ou vous êtes en train de l'acquérir, c'est votre métier, vous êtes payé pour ça, vous portez le projet.

Bien sûr, vous devez vous assurer que les exigences que vous spécifiez sont réalisables, que les contraintes techniques sont elles aussi spécifiées, que l'ensemble est cohérent. Bien sûr, vous pouvez, et parfois vous devez, dialoguer avec la maîtrise d'œuvre. Mais ce qui adviendra du système après la livraison du cahier des charges ne fait pas partie de votre mission. Cela peut légitimement vous intéresser, mais ne doit en aucun cas consommer votre énergie.

# Les questions à large spectre

## Comment utiliser ces questions ?

Ces questions peuvent servir à constituer un guide d'entretien de dernière minute, ou à défricher un terrain inconnu, en évitant l'angoisse de la page blanche. Il ne s'agit en aucun cas de les égrener de la première à la dernière : cela ne ferait que mettre l'interlocuteur mal à l'aise (ou s'il l'est déjà, de le mettre *encore plus* mal à l'aise). Et de toute façon, elles sont redondantes. Quoi qu'il en soit, ces questions génériques peuvent être une aide ; elles ne sont pas une panacée.

## Liste de questions

### Questions de situation, questions génériques

Ces questions permettent de connaître le contexte.

- En quoi consiste votre activité actuelle ? (Cette question permet de définir ou de cerner la partie du processus qui va impacter le futur utilisateur interviewé.)
- Pouvez-vous décrire la manière dont votre service (bureau, organisation, entreprise...) fonctionne actuellement ?
- Quel est votre rôle actuel ?
- Dans quelle mesure votre activité sera-t-elle impactée par le futur système ?

Les questions suivantes sont très générales. La première, très (trop) générique, est à utiliser avec modération.

- Qu'attendez-vous du futur système ?

- Que souhaitez-vous que le système fasse ?
- En quoi le futur système va-t-il améliorer votre activité ?

### Questions d'objectif

Ces questions sont à poser au maître d'ouvrage stratégique et aux parties prenantes décisionnaires.

- Quel problème voulez-vous résoudre ?
- Qu'attendez-vous de la solution future, du futur logiciel ?
- Comment saurez-vous que la solution proposée aura atteint son objectif ?
- Une fois que votre système sera opérationnel, que va-t-il vous apporter ?
- Quelles seront les retombées positives de la mise en place de la solution future, du futur logiciel ?
- Quelles seront les conséquences négatives de la mise en place de la solution future, le logiciel ?

### Questions de manque

Par définition, un manque est un besoin. Chercher les manques est donc une bonne façon de faire exprimer les besoins.

- Qu'est-ce qui manque au système actuel ?
- Comment ce manque qui pourrait-il être comblé par le futur système ?
- Qu'est-ce qui est défaillant avec le système actuel ?
- Qu'est-ce qui vous dérange le plus dans le système actuel ?
- Si une chose vous dérangeait avec le système actuel, ce serait quoi ?
- Que voudriez-vous que le système à venir fasse et que le système actuel ne fait pas ?
- Peut-on faire ensemble la liste de toutes les fonctions que le système actuel accomplit mal ?

### Questions sur le système à l'étude

Pour connaître les données :

- Quelles sont les données manipulées par le système ?
- Quelles données le système devra-t-il stocker ?
- Quelles sont les données contenues dans cette entité ?
- Quelles sont les relations entre ces deux données ?

Pour identifier les états et transitions d'un objet :

- Quel est le cheminement du dossier, d'un objet ?
- Dans quels états peut se trouver le dossier, l'objet ?

- Quel événement va modifier le dossier, l'objet ?
- Que se passe-t-il suite à cet événement ?
- Que se passe-t-il si cet événement n'a pas lieu ? (Exception)

### Questions de perspective

On demande au futur utilisateur ou toute autre partie prenante de se projeter dans l'avenir.

- Qu'attendez-vous du futur système ?
- Que voudriez-vous que le système à venir fasse ?
- Que voudriez-vous que le système à venir fasse mieux que le système actuel ?
- Qu'est-ce qui doit être amélioré par le futur système ? (Sans nécessairement définir une solution, cette question approfondit le besoin.)
- Quel critère doit-on utiliser pour prouver que le système satisfait cette exigence ?
- Pouvez-vous décrire le futur système, tel que vous l'imaginez ?

### Questions de précision

- Oui... ? (Se taire et attendre que l'interviewé précise spontanément la pensée.)
- Oui, et plus précisément ? (Éviter « oui, mais... » ; préférer « oui, et... ».)
- Un de vos collègues (si nécessaire, préserver la confidentialité) a exprimé le besoin d'avoir [...]. Êtes-vous du même avis que lui ? (Rassurer l'interlocuteur : il a le droit d'avoir un avis différent.)
- Comment faites-vous ? (Apporte du détail.)
- Pourquoi[...] ? (Remonte à des exigences de plus haut niveau ; à utiliser avec prudence, et ne jamais impliquer l'interviewé avec ce type de questions.)

### Critères de priorisation et de validation

On demande maintenant à l'utilisateur de définir ses priorités et des critères de validation de l'exigence.

- En quoi cette exigence est-elle importante pour vous ?
- Qu'est-ce qui fera que vous serez satisfait avec le futur système ?
- Comment saurez-vous que le futur système est satisfaisant ?
- Pouvez-vous indiquer les trois fonctions indispensables du futur système ?
- Si vous ne deviez conserver que trois fonctions parmi celles décrites, lesquelles conserveriez-vous ?





# Glossaire français

**Besoin.** Nécessité ou désir éprouvé par un utilisateur (AFNOR X50-150).

**Cahier des charges.** Document par lequel le demandeur exprime son *besoin* (ou celui qu'il est chargé de traduire), en termes de *fonctions de service* et de *contraintes*. Pour chacune d'elles, sont définis des *critères d'appréciation* et leurs niveaux. Chacun de ces niveaux doit être assorti d'une *flexibilité* (AFNOR X50-150).

**Client.** Le client (ou maître d'ouvrage) sera le propriétaire de l'objet du cahier des charges. Il est responsable de la définition des exigences, du financement, et de l'approbation de la solution.

**Contrainte.** Limitation à la liberté de choix du concepteur réalisateur du produit (AFNOR X50-150).

**Critère d'appréciation.** Caractère retenu pour apprécier la manière dont une fonction est remplie ou une contrainte respectée (AFNOR X50-150).

**Demande.** Expression, encore subjective et partielle, de la perception d'un besoin, à un instant donné.

**Développement de logiciel.** Processus par lequel les besoins des utilisateurs sont transformés en spécifications, les spécifications en conception, la conception en code, et le code testé, documenté et validé en vue d'un usage opérationnel (définition IEEE).

**Domaine fonctionnel.** Ensemble d'activités cohérentes concourant à une finalité de l'organisme.

**Élémentaire (exigence).** Se dit d'une exigence qui ne peut être décomposée, insécable (en anglais, *atomic*).

**Ergonomie.** Discipline scientifique qui vise la compréhension fondamentale des interactions entre les êtres humains et les autres composantes d'un système, et la mise en œuvre dans la conception de théories, de principes, de méthodes et de données pertinentes afin d'améliorer le bien-être des hommes et l'efficacité globale des systèmes (Association Internationale d'Ergonomie).

**Exigence.** Description formelle d'un objectif. L'appréciation de la satisfaction d'une exigence peut être mesurée.

**État de l'art.** Ensemble des règles respectées implicitement par les professionnels d'une activité.

**Fonction.** Action d'un produit, ou de l'un de ses constituants, exprimée exclusivement en termes de finalités (AFNOR X50-150).

**Fonction de service.** Action attendue d'un produit (ou réalisée par lui) pour répondre à un élément du besoin d'un utilisateur donné (AFNOR X50-150).

**Fournisseur.** Le fournisseur (ou maître d'œuvre) s'engage sur les conditions qu'il accepte : échéancier des livraisons, maîtrise des coûts, satisfaction des exigences.

**Ingénierie du logiciel** ou *génie logiciel* (en anglais, *Software Engineering*). Approche systématique du développement, de l'exploitation, de la maintenance et de la mise à la retraite d'un logiciel (définition IEEE).

**Logiciel.** La somme des programmes, procédures, règles de gestion, en rapport avec le fonctionnement d'un ordinateur, ainsi que la documentation et les données qui leur sont associées (définition IEEE).

**Maquette.** Objet destiné à aider un client à spécifier ses demandes.

**Processus.** Ensemble d'activités corrélées ou interactives qui transforme des éléments d'entrée en éléments de sortie (ISO 9000).

**Produit.** Ce qui est (ou qui sera) fourni à un utilisateur pour répondre à un besoin. Le produit peut être un objet, un fluide, une prestation de service, un processus industriel ou administratif (procédé, logiciel, procédure, etc.) ou toute combinaison de ceux-ci (AFNOR X50-150).

**Prototype.** Objet destiné à vérifier la faisabilité d'une solution sous un aspect particulier : conceptuel, organisationnel ou technique. Un prototype enchaîne des parties réelles (celles dont on veut montrer la faisabilité) et des parties fictives (en général, celles qui correspondent à des fonctions dont la réalisation ne semble poser aucun problème).

**Qualité.** Ensemble des propriétés et caractéristiques d'un produit, ou service, qui lui confèrent l'aptitude à satisfaire des besoins, exprimés ou implicites. La maîtrise d'ouvrage et l'utilisateur s'intéressent principalement à la qualité du produit, alors que la maîtrise d'œuvre s'intéresse à la qualité du processus. Cette dernière contribue à la qualité du produit.

**Solution.** Ensemble coordonné de moyens matériels, logiciels, humains, organisés pour satisfaire un ensemble d'exigences.

**Spécification.** Traduction d'exigences en éléments techniques.

**Système.** Ensemble cohérent d'éléments matériels et logiciels en interaction dynamique, destiné à assurer une finalité préalablement définie.

**UML (*Unified Modeling Language*).** Langage standard de représentation de l'information (données, traitements), principalement sous forme de diagrammes.

**Utilisateur.** Personne ou entité pour qui le produit a été conçu, qui exploite au moins une des fonctions du produit. (AFNOR X50-150).



# Glossaire bilingue

La littérature anglophone sur les exigences (en anglais, *requirements*) est très riche et le vocabulaire s'y rapportant fait l'objet d'un large consensus entre auteurs. Ce n'est pas le cas en français, où l'on emploie des mots différents pour désigner les mêmes objets ou concepts. De plus, les termes et expressions varient en fonction du pays (France, Canada, Belgique, Suisse), du métier (informatique technique, systèmes d'information, ingénierie système) et des auteurs.

Le lecteur ne devra donc pas s'étonner si les termes employés dans cet ouvrage sont différents de ceux avec lesquels il a été familiarisé. Malgré notre effort de minimiser le nombre de termes flous et d'utiliser autant que possible les mêmes mots pour désigner les mêmes concepts, certaines ambiguïtés n'ont pu être évitées.

Voici une liste d'équivalence (non exhaustive) entre les termes anglophones et le(s) équivalent(s) utilisé(s) en français dans cet ouvrage.

**Analyst** ou *requirements analyst*. Littéralement « analyste des exigences ». Analyste fonctionnel ; analyste métier ; assistant à la maîtrise d'ouvrage, conseil à la maîtrise d'ouvrage.

**Business requirements**. Exigences métier (au Québec : exigences d'affaires).

**Client, customer**. Client ; maîtrise d'ouvrage (stratégique ou opérationnelle).

**Elicitation (requirements)**. Recueil des besoins, ou capture des besoins ou des exigences.

**Need**. Besoin. Dans le cadre de l'analyse des besoins, terme beaucoup moins usité que *requirement*.

**NFR**, *non-functional requirements* : exigences non fonctionnelles.

**Owner**. Littéralement « propriétaire ». Maître d'ouvrage stratégique, donneur d'ordres.

**Requirement**. Besoin ou exigence (cet ouvrage fait explicitement le distinguo entre ces deux termes).

**Requirements engineering.** Ingénierie des besoins ; ingénierie des exigences.

**Software requirements.** Exigences logicielles, exigences du logiciel.

**SRS, *Software requirements specification*.** Spécification du besoin, spécification des exigences.

**SRS document, *Software requirements specification document*.** Cahier des charges ; cahier des charges fonctionnel ; spécifications fonctionnelles générales ; spécification des exigences.

# Bibliographie

Alistair Cockburn, *Rédiger des cas d'utilisation efficaces*, Eyrolles, 2001.

Yves Constantinidis, *Le logiciel à valeur ajoutée, la perspective de l'utilisateur*, Hermes Science, 2001.

Yves Constantinidis, *Définition des besoins pour le logiciel*, Hermes Science, 2006.

Tom Gilb, *Competitive Engineering : A handbook for systems engineering, requirements engineering, and software engineering management using planguage*, Butterworth-Heinemann, 2005.

Ellen Gottesdiener, *Requirements by collaboration*, Addison-Wesley, 2002.

Ellen Gottesdiener, *Software requirements memory logger*, Goal/QPC, 2005.

International Institute of Business Analysts, *A guide to Business Analysis Body of Knowledge (BABOK Guide)*, version 2.0, 2010.

Soren Lauesen, *Software Requirements – Styles and Techniques*, Addison-Wesley, 2002.

Hugues Marchat, *La gestion de projet par étapes, étude des besoins*, Eyrolles, 2008.

Paul-Hubert des Mesnards, *Réussir l'analyse des besoins*, Eyrolles, 2007.

Suzanne and James Robertson, *Mastering the Requirements Process*, Addison-Wesley, 1999.

Karl Wiegers, *Software requirements*, Microsoft Press, 2003.

Karl Wiegers, *More About Software requirements*, Microsoft Press, 2006.





# Index

## A

**accompagnement (services d'~)** 139

**acteurs** 12, 18, 43, 93

**activités (diagramme d'~)** 113

**affinités (diagramme des ~)** 83

**Afnor X50-151 (modèle)** 158

**AHP (Analytical Hierarchical Process)** 112

**ambiguïtés** 32, 83, 88

**amélioration du processus d'élaboration** 26, 187

**analyse** 107

check-list d'~ 126

comparative *Voir* étude

comparative

de documents 74

de l'existant 12, 180

de produits existants 86

des besoins 182

des exigences 16, 44, 45

des règles métier 107

des risques 63, 218

d'impact 199, 200

étape 103

étape d'~ 54

parties prenantes 54

processus 104

**analyste** 9, 11, 18, 24, 27, 28, 54

activité 2

compétences 27

engagements de l'~ 40

savoir-être 30

savoir-faire 29

**Analytical Hierarchical Process (AHP)** 112

**animateur (qualités d'~)** 29

**anticipation (technique de l'~)** 190

**arbres de décision** 121

**assistant**

maîtrise d'ouvrage *Voir* analyste

**assistant, maîtrise d'ouvrage**

*Voir* analyste

**atomic requirements** *Voir* exigences élémentaires

**attributs**

de qualité 15

des exigences 200

**automate d'états finis** 120

## B

**besoins**

alternatifs 89

concept 18

exceptionnels 89

négatifs 90

**B.O.R.D. (tableau de ~)** 214

**Box, George** 178

**brainstorming** 29, 83

**business requirement** *Voir* exigence métier

## C

**cadrage**

document 58

étape 178

**cahier**

- des charges 12, 19
- modèle 106, 155
- utilité 21
- des clauses techniques particulières (CCTP) 164

**capacité fonctionnelle** 129

**capitalisation (étape de ~)** 185

**caractéristiques de qualité** 127

**carte de processus** 113

**cas d'utilisation** 15, 42, 93, 105, 126

- avantages 96
- contenu 94
- diagramme 100
- difficultés et risques 99
- élaboration 96
- modélisation 96

**catégories d'exigences** 15, 105

**CCB (Change Control Board)** 198

**CCTP (cahier des clauses techniques particulières)** 24, 164

**cent points (méthode des ~)** 110

**Change Control Board (CCB)** 198

**changement**

- demandes de ~ 201
- gestion des 197, 209

**charges**

- enregistrement 63
- et délais, estimation 62

**check-list**

- cahier des charges 165
- d'analyse 126
- de bonne formulation 145
- de contenu 169
- de l'étape de validation 175
- de propriétés 169
- de sélection d'outils 205
- de spécification d'une exigence 152
- d'étape de spécification 153
- étape de concept et objectifs 58
- étape de recueil 91
- plan projet 47, 64

règle des 5 C 146

vérification par ~s 169

**classes (diagramme de ~)** 117

**classification des exigences** 42

**Cockburn, Alistair** 94, 95, 100, 101

**cohérence** 89

**commission de contrôle des changements** 198

**compatibilité (ergonomie)** 134

**concept**

- check-list 58
- et objectifs 49

**concision (d'ergonomie)** 136

**contexte (diagramme de ~)** 14, 53, 58, 94, 97, 116, 126, 157, 179, 180, 183

**contraintes** 13, 15

- de coût 137
- de maintenance 140
- d'environnement 138
- de projet 137
- de support 140
- organisationnelles 42
- réglementaires 42
- techniques 16, 25, 49

**contrôle explicite (ergonomie)**

136

**coût**

- contraintes de ~ 137
- de correction d'une erreur 22
- des exigences 25
- des exigences (évaluation) 124

**créativité** 25, 31, 73

**CRUD (matrice)** 124

**cycle de vie du logiciel** 35

**D**

---

**délais, enregistrement** 63

**demandes**

- concept 18
- de changement 201

**développement des exigences**

21, 44, 65

**diagramme**

- d'activités 113
- de cas d'utilisation 100
- de classes 117
- de contexte 53, 58, 94, 97, 116, 126, 157, 179, 180, 183
- de flux de données 114
- de Kano 124
- des affinités 83
- de séquence 116
- entités-associations 117
- états-transitions 120
- process map 113
- UML 113

**dictionnaire de données** 107, 126

**directeur**

- de mission 62
- de projet 62

**documentation**

- des besoins 69
- des exigences 41
- exigences de ~ 141

**documents**

- analyse de ~ 74
- de cadrage 58
- navette (méthode du ~) 191

**documents-types** 61

**domaine d'application (détermination)** 52

**donneur d'ordres** 23, 54, 86

**E**

**échelle de Likert** 85

**écoute (qualité d'~)** 30, 80, 90

**Eisenhower (principe d'~)** 110

**élémentaires (exigences)** 42

**entités-associations (diagramme)** 117

**entonnoir**

- démarche de recueil 80
- méthode de sélection 213

**environnement**

- contraintes 138

- de développement (contraintes) 138

- physique d'utilisation 141

**estimation**

- charges et délais 62

**étape**

- d'analyse 54, 103, 182
- d'analyse de l'existant 180
- de cadrage 178
- de capitalisation 185
- de planification 179
- de recueil 67, 182
- de spécification 143, 184
- de validation 167, 184
- planification 59
- processus d'exigences 43

**états-transitions (diagramme ~)**

120

**étude**

- comparative 60, 214
- de choix 213
- design 218
- d'intégrabilité 218
- d'opportunité 216

**événement métier** 94, 97

**exigences** 11, 19

- comportementales 42
- de haut niveau 42
- de qualité 42, 106
- d'intégration 140
- d'interface 106
- d'interfaces externes 15
- élémentaires 42, 97, 151
- fonctionnelles 14, 15, 106
- formulation 147
- métier 14, 16, 108
- non fonctionnelles 15, 106, 127, 151
- caractéristiques de qualité 127
- norme ISO 25000 128
- processus de développement 44
- spécifications rampantes 111
- sur les données 15
- sur les formats de données 106

## **expert**

métier 62

technique 63

**expression écrite (aptitude)** 30

## **F**

**facilité d'utilisation (utilisabilité)**  
130

**faisabilité des exigences (évaluation)** 124

**feedback** 69, 88

**fiabilité** 129

**finalité-avantage-mesure (objectifs)** 51

**flux de données (diagramme de ~)** 114

## **fonction**

à satisfaction proportionnelle  
125

attractive 125

obligatoire 125

**fonctionnalité** 129

**formalisation des objectifs** 51

**formation (exigences de ~)** 141

**forme de description des exigences** 150

**formulation des exigences** 143, 144

## **G**

**gains** 23

**généralisations** 81

## **gestion**

de configuration 197

de projet 197

des attributs 208

des changements 197, 209

des erreurs (ergonomie) 136

des exigences 45, 63, 195, 197

des versions 209

**Gottesdiener, Ellen** 76, 191

**grille pouvoir-impact** 55

**groupe de travail** 75, 87

**guidage (ergonomie)** 135

## **H**

**homogénéité (ergonomie)** 135

## **I**

**IEEE 830 (modèle)** 149, 156

**IEEE 982 (cycle de vie du logiciel)**  
35

**impact (analyse d'~)** 200

## **ingénierie**

des besoins 27

des exigences 9, 11, 13, 14, 19,  
35, 41, 44, 67

du logiciel 197

**installation** 137

**intégrabilité (étude d'~)** 140

**intégration (exigences d'~)** 140

## **interface**

contraintes 139

exigences d'~ 106

## **interview**

déroulement 77

distorsions 82

structurée individuelle 76

types de questions 78

**ISO 25000 (norme)** 128, 152

**itérations, processus d'exigences**  
189

## **J**

**jurisprudence** 17

## **K**

**Kano (diagramme de ~)** 124

## **L**

**langue de bois** 90, 146

**lettre de mission** 24, 58, 74, 178

**Likert (échelle de ~)** 85

**liste des parties prenantes** 54

**logiciel** 35

**loi de Pareto** 224

## M

**maintenabilité** 132

**maintenance**

contraintes de ~ 140

phase de ~ 35

**maître d'œuvre** 11, 36

**maître d'ouvrage** 36

engagements du ~ 40

opérationnel 11

stratégique 12

**maîtrise d'ouvrage** 54

**maquette** 29

**matrice**

CRUD 124

RACI 124

**McCall (typologie de ~)** 127

**méthode des cent points** 110

**méthodologie** 60

**migration** 138

**mise en exploitation** 137

**modèle**

Afnor X50-151 158

de cahier des charges 155

de document 61

de processus d'exigences 177

de Wiegiers 160

IEEE 830 149, 156

Volere 132, 149, 162

**modélisation** 9, 44, 63

cas d'utilisation 96

graphique 112

boîte à outils 122

maquette 122

prototype 122

techniques de ~ 28

**MTBF, mean time between failures** 130

## N

**négociation des exigences** 15, 29, 89, 110, 192

**niveau d'exigences** 42

**norme ISO 25000** 128, 152

**norme ISO/CEI 25000** 128

## O

**objectifs** 50, 105

stratégiques 42

**observation directe** 84

**observation (qualité d'~)** 31

**Ockham (rasoir d'~)** 119

**omissions** 81

**opérateurs modaux** 81

**organigrammes** 121

**outils** 60, 205

choix 210

fonctions de base et avancées 206

## P

**Pareto (loi de ~)** 224

**parties prenantes** 12, 43, 50

analyse 18, 54

concept de ~ 18

liste 54

tableau 57

**périmètre** 50, 52

**phase d'exigences** 38

**plan de recueil** 70

**planguage (planning language)** 151

**planification** 59

aptitude 29

étape 179

recueil des besoins 68

**plan projet** 59

check-list 47, 64

contenu 63

élaboration 61

**portabilité** 132

**pouvoir-impact (grille ~)** 55

**principe d'Eisenhower** 110

**principe de simplicité** 119

**priorisation, projet** 109

**processus**

- carte de ~ 113
- d'analyse 104
- d'élaboration 185
- de recueil 68
- de spécification 143
- de validation des exigences 168
- développement des exigences 45
- d'exigences
  - amélioration 187

**processus d'exigences** 60

- amélioration 189
- description formelle 44
- description pratique 45
- étapes 43
- itérations 189
- modèle 177
- récurions 188
- rétroactions 188

**profils utilisateurs** 25

- détermination 71
- exemple de typologie 72
- identification des ~ 61
- segmentation (ergonomie) 134
- typologie 71

**projet**

- d'élaboration 59
- méthodologie 60
- priorités (définition des ~) 109

**prototypage** 123

**Q**

---

**QFD (Quality Function Deployment)** 112

**qualité**

- contrôle des exigences 174

**Quality Function Deployment** 112

**quantificateurs universels** 81

**questionnaires (recueil par ~)** 85

**questionnement (recueil d'objets ; grille de ~)** 56

**R**

---

**RACI (matrice)** 124

**rasoir d'Ockham** 119

**recette** 137

**recueil** 44

- bonnes pratiques 87
- check list 91
- de planification 68
- des besoins 182
- des exigences 45
- des objectifs 51
- entonnoir 80
- étape 67
- grille de questionnement 56
- par observation directe 84
- par questionnaires 85
- plan de ~ 70
- processus 68
- risques 70
- techniques 73
- vérification 69

**récurions sur le processus d'exigences** 188

**règles**

- analyse 107
- de gestion 13, 15, 105, 107
- métier 15, 16, 105, 108
- traçabilité 109

**relationnelle (qualité ~)** 31

**relecture**

- croisée (vérification par ~) 170
- simple (vérification par ~) 170

**rendement** 131

**ressources (identification des ~)** 62

**retour sur investissement** 23

**retravail** 190

**rétroactions, processus d'exigences** 188

**réutilisation des exigences** 86, 150

**rework** 188, 190

**risques** 24

- analyse 63

**Robertson, James** 174

Robertson, Suzanne 174  
Robertson, Suzanne et James 162

## S

SAE (système à l'étude) 14  
savoir-faire 27  
screening 213  
segmentation des profils utilisateurs 134  
séquence (diagramme de ~) 116  
services d'accompagnement 139  
short-list 213  
simplicité (principe de ~) 119  
solutions (suggestions de ~) 106  
souplesse (ergonomie) 135  
sources d'exigences 61, 72, 74  
sous-spécification 25  
spécification  
    concept 18  
    des exigences 45, 184  
    étape 184  
    processus de ~ 143  
    rampante 24, 26, 123, 198  
structuration des exigences 105, 144, 149  
suggestions de solutions 106  
support (contraintes de ~) 140  
surspécification 25  
SWOT (analyse) 214  
système 13

## T

tableau  
    de B.O.R.D. 214  
    des parties prenantes 57  
tables de décision 121  
techniques  
    de l'anticipation 190  
    de recueil 73  
    de recueil des objectifs 52  
    de validation 168  
template 61  
traçabilité des exigences 207  
typologie de McCall 127

## U

UML (Unified Modeling Language) 93, 100, 105, 113, 118  
Unified Modeling Language  
Voir UML  
use cases Voir cas d'utilisation  
utilisabilité 130  
utilisateurs  
    besoins des ~ 15  
    participation des ~ 23  
    profils 25

## V

validation  
    check-list 175  
    des exigences 44, 45, 184  
    étape 167, 184  
    processus de ~ 168  
vérification  
    cas de test (élaboration de ~) 173  
    contrôle qualité des exigences 174  
    croisée 170  
    des exigences 144  
    inspection 171  
    par check-lists 169  
    par relecture  
        croisée 170  
        simple 170  
    revue formelle 171  
    simple 170  
versions (gestion des) 209  
Volere (modèle ~) 132, 149, 162

## W

Wieggers, Karl 15, 105, 160  
Wieggers (modèle de ~) 160

## Z

zone floue 39, 89, 213

# Expression des besoins

## pour le système d'information



### L'auteur

Consultant en systèmes d'information, **Yves Constantinidis** a dirigé avec succès l'élaboration de plusieurs cahiers des charges de portée nationale (ministère de la Santé, ministère de l'Éducation nationale). Informaticien de formation, il intervient sur des missions de choix de solutions, de diagnostic du système d'information et d'amélioration du processus de développement. Son expertise l'a amené à publier trois ouvrages sur l'ingénierie du système d'information et la qualité du logiciel (éditions Hermès), et à intervenir comme formateur auprès d'établissements d'enseignement supérieur (Epita, École des hautes études en santé publique).

**Michel Volle** est président d'honneur du Club des Maîtres d'Ouvrage des Systèmes d'Information.

Comment recueillir tous les besoins des acteurs du système d'information, et rien que leurs besoins réels ? Comment se mettre d'accord sur la spécification des exigences ? Comment aboutir à un cahier des charges clair, complet et consensuel ? Phase cruciale dans le choix, le développement ou la mise en œuvre d'une solution d'entreprise, la définition des besoins conditionnera en effet la réussite du projet, notamment son coût et sa qualité. Mais cette étape est complexe et délicate, en raison du nombre et de la diversité des parties prenantes, des demandes souvent divergentes, des contraintes variées et, *last but not least*, du facteur humain.

Véritable guide de terrain, nourri par la grande expérience de son auteur, cet ouvrage présente une démarche et des techniques éprouvées pour recueillir et formaliser les vrais besoins, afin d'élaborer un cahier des charges d'une qualité irréprochable, dans des délais raisonnables et au moindre coût. À partir d'un exemple permettant de mieux saisir les enjeux, la première partie expose les prérequis, puis décrit le processus et les activités préparatoires indispensables : définition des objectifs et du périmètre, analyse des parties prenantes, planification de l'élaboration. La deuxième partie détaille les quatre grandes étapes de cette méthode (recueil, analyse, spécification et validation), avec à la clé des modèles de documents, des grilles et des check-lists. Enfin, la troisième partie porte sur les techniques et les outils de gestion des exigences, et s'achève par des conseils pour s'améliorer encore. Grâce à cet ouvrage d'une clarté exceptionnelle, le lecteur sera ainsi prêt à relever les défis que pose toute mission de définition des besoins.

### À qui s'adresse ce livre ?

- Aux assistants à la maîtrise d'ouvrage (AMOA)
- Aux consultants en systèmes d'information
- Aux experts techniques et métier
- Aux architectes du système d'information
- À tous ceux qui sont impliqués dans l'élaboration d'un cahier des charges

### Au sommaire

**Méthodologie.** La méthode en action. Les enjeux d'une bonne définition des besoins. Compétences et savoir-faire. Exigences et cycle de vie du logiciel. La démarche. Définir le concept et les objectifs. Planifier le projet d'élaboration. **Développement des exigences.** L'étape de recueil. Les cas d'utilisation. L'étape d'analyse. Les exigences non fonctionnelles. Exigences projet et contraintes techniques. L'étape de spécification. Structure du cahier des charges. L'étape de validation. Améliorer le processus. **Faire vivre les exigences.** La gestion des exigences. Les outils. Au-delà des exigences. Neuf conseils.