Avant-propos

À l'heure actuelle, près des deux tiers des projets logiciels échouent à répondre de manière efficace et efficiente aux attentes des clients et des utilisateurs. Cela représente des milliards d'euros de pertes annuelles, un énorme gâchis de ressources, et parfois même des accidents mortels. La principale raison de l'échec est l'incompréhension entre client et fournisseur (ou MOA et MOE) à propos des spécifications des exigences logicielles, autrement dit, sur le contenu du cahier des charges lors des phases amont du développement. Lorsque les personnes du métier (la maîtrise d'ouvrage) et les informaticiens (la maîtrise d'œuvre) se mettent autour de la table pour définir un nouveau logiciel, on constate que les uns ont du mal à exprimer les besoins et que les autres ont des difficultés à les comprendre.

Le but de ce livre est de fournir un outil utilisable à la fois par les experts et les novices, par les futurs utilisateurs et les consultants expérimentés, pour leur permettre de partager un langage commun et une compréhension commune lorsqu'ils écoutent, expriment, ou spécifient des besoins pour un système à base de logiciel. Il fournit la démarche et les techniques dont tous les acteurs d'un projet ont besoin pour améliorer la communication et atteindre leurs objectifs. Cette communication est indispensable à chaque membre de l'équipe et à toute personne dont le travail impacte, ou est impacté par, le processus de développement logiciel, car elle simplifie le processus de définition du concept et de définition des exigences, qui constituent les deux premières phases de la création d'un logiciel.

Ce livre ne s'adresse pas qu'aux experts. Il est destiné à un large public, à savoir toute personne qui a la responsabilité d'exprimer des besoins ou d'élaborer un cahier des charges ou de participer à son

élaboration. Parmi la centaine de méthodes, de techniques et d'outils d'aide à l'expression des besoins, nous en avons retenu une dizaine, que nous avons explicitée et détaillée. D'autres techniques sont simplement citées pour mémoire. Nous avons également limité au strict minimum le vocabulaire technique, défini les termes spécialisés dans un glossaire et explicité dans des encadrés les nombreux synonymes et mots ambigus et ambivalents dans la profession.

Nous espérons que vous vous en servirez au quotidien, comme un outil pratique pour vous aider à recueillir, définir, développer et gérer les besoins.

Comment utiliser cet ouvrage

Cet ouvrage est un guide pratique que vous pouvez utiliser sur le terrain. Il est conçu pour faciliter la communication entre les équipes métier (qu'ils soient cadre de santé, infirmier, banquier ou assureur) et les informaticiens (directeurs des systèmes d'information, responsables informatiques, concepteurs et réalisateurs de logiciels) lors de la définition des besoins pour un projet logiciel. Il fournit des méthodes, des techniques et des modèles de document accessibles à tous les acteurs du projet pour recueillir, analyser, spécifier, et valider les exigences logicielles. Pour chaque étape du processus, le livre fournit des exemples tirés de cas réels, afin de montrer comment ces méthodes, techniques et modèles peuvent être mis en œuvre en pratique.

Cet ouvrage est destiné à un large public. Au risque de frustrer les experts, nous n'y avons pas inclus les techniques les plus complexes de l'ingénierie des besoins, ni celles qui requièrent des compétences techniques en informatique ou en conception des systèmes d'information.

La plupart des exemples sont tirés du monde de l'hôpital. Il n'est cependant pas nécessaire d'être un expert en informatique hospitalière pour les comprendre. Même sans être médecin ou infirmier, il est aisé d'imaginer comment les mécanismes utilisés pour définir les besoins de la prescription médicamenteuse, des rendez-vous patient ou de la gestion des lits peuvent être transposés à d'autres domaines.

Les trois premiers chapitres décrivent les enjeux, les mécanismes généraux et le processus global de définition des besoins et d'élaboration d'un cahier des charges. Les chapitres suivants détaillent chaque étape, sachant que le processus d'expression des besoins est itératif et incrémental.

Le premier chapitre positionne l'expression des besoins et le cahier des charges dans son environnement informatique et humain. On y définit le vocabulaire de base à connaître.

Le chapitre 2 présente, dans ses grandes lignes, la méthode globale et les quatre activités de base d'expression des besoins.

Le chapitre 3 donne une méthode en sept étapes. Il peut vous servir de carte pour vous orienter à chaque étape.

À partir de ce point, chaque chapitre traite d'une problématique et une seule, et les problématiques sont interdépendantes. Par exemple, l'énoncé de l'objectif, étape indispensable, est décrit au chapitre 5, mais la vérification de cet énoncé fait appel aux check-lists générales de vérification des exigences décrites au chapitre 12. Nous vous conseillons de parcourir rapidement les chapitres 4 à 12 puis, selon vos besoins, de relire l'ensemble dans le détail ou de vous concentrer sur le chapitre qui vous intéresse prioritairement.

Le chapitre 4 explicite les étapes préparatoires : étude de l'existant, planification de l'élaboration, choix des outils et des représentants des utilisateurs.

Le chapitre 5 aborde un aspect fondamental, et très souvent négligé de la construction du logiciel : la phase de concept. C'est lors de cette phase qu'est établi le « triangle d'or », les trois éléments fondamentaux et interdépendants que sont l'objectif, le périmètre et les parties prenantes. Il est important de comprendre les enjeux de cette phase, quitte à relire le chapitre dans les détails dans un deuxième temps.

Les chapitres suivants détaillent chacun une des étapes ou techniques présentées aux précédents chapitres.

Le chapitre 6 décrit la technique d'élaboration des *cas d'utilisation* (*use cases*), une manière de découvrir et de spécifier les exigences en démontrant les interactions entre l'utilisateur et le système.

Les chapitres 7, 8 et 9 détaillent les techniques de recueil, d'analyse et de spécification des exigences, et le **chapitre 10** se concentre sur les exigences d'interface, de qualité du produit et sur les contraintes.

Le chapitre 11 présente, paragraphe par paragraphe, le modèle de spécification d'exigences (cahier des charges) que nous préconisons. Il est tout à fait possible de commencer la lecture de l'ouvrage par ce

chapitre, après avoir parcouru les chapitres d'introduction, et de revenir au cœur de l'ouvrage par la suite.

Le chapitre 12 nous parle des techniques de vérification et de l'étape de validation, deux activités indispensables à l'obtention d'un cahier des charges de qualité.

Dans toute la mesure du possible, nous avons évité de faire référence à de concepts méthodologiques complexes, et nous avons limité le jargon technique, afin que cet ouvrage soit lisible et utilisable par tous. Cependant, l'ouvrage ne serait pas complet sans les annexes et le glossaire.

En annexe, nous explicitons deux notions fondamentales, dont on parle souvent sans toujours en cerner la portée : celle de *système d'information* et celle de *cycle de vie du logiciel*. Ce sont là des concepts de base indispensables à l'ingénierie du logiciel.

Un glossaire définit tous les termes techniques qui ont été utilisés dans l'ouvrage afin que tous les lecteurs « parlent la même langue ».

La première fois qu'ils apparaissent dans le texte, les termes inclus dans le glossaire sont indiqués en italique et les deux concepts fondamentaux de *système d'information* et de *cycle de vie du logiciel* sont décrits en annexe.

Vous trouverez également à la fin du livre une liste de références de ressources parmi les plus utiles pour approfondir votre compréhension des outils et des concepts décrits dans cet ouvrage.

Convention d'écriture et abréviations

Dans les schémas de cet ouvrage, nous utilisons les conventions d'écriture suivantes :

SAE, ou *système à l'étude*, désigne le système qui fait l'objet du cahier des charges en cours d'élaboration.

CU est l'abréviation de *cas d'utilisation* (en anglais *use case*) tel que décrit au chapitre 6.

SI est l'abréviation de *système d'information*[1].

La lettre « C » dans un losange blanc signifie *vérification* (check-list).

La lettre « V » dans un losange noir signifie *validation*.

On trouvera en annexe une explication de la différence entre système d'information et système informatique et la définition de ces termes et expressions dans le glossaire à la fin de l'ouvrage. Pour comprendre la différence entre vérification et validation, vous pouvez vous reporter au chapitre 12.

1. Voir en annexe la différence entre système d'information et système informatique. Voir le glossaire des termes pour leur définition.

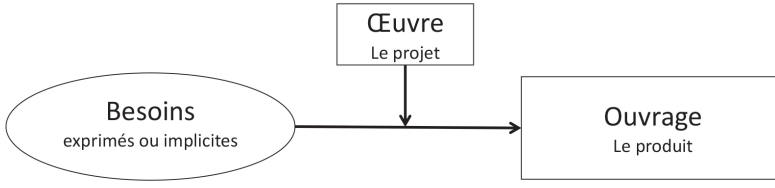
Chapitre 1

Qu'est-ce que c'est ? À quoi ça sert ?

Qu'est-ce que c'est ? Un peu de vocabulaire

Un *projet* informatique est un ensemble d'activités dont on a défini l'objectif, les délais et les ressources.

Un projet est une *œuvre*, c'est-à-dire un ensemble de travaux dont le but est de produire un *ouvrage* conforme à ses *spécifications*.



Œuvre et ouvrage

Le *maître d'ouvrage* est le propriétaire du produit (l'ouvrage). Il exprime des besoins auxquels doit satisfaire l'ouvrage final. Le maître d'ouvrage agit au nom de l'ensemble des futurs *utilisateurs* de l'ouvrage et autres *parties prenantes*.

Le *maître d'œuvre* est responsable du projet (l'œuvre) dans le respect des conditions de coûts et de délais. Il doit livrer un résultat (ouvrage) de qualité, c'est-à-dire conforme aux *besoins exprimés* (tels que fonctions et performances spécifiées) mais aussi à des *besoins implicites* (règles, normes, standards de droit ou de fait) et des contraintes (de coût, de délai, de logiciels et matériels qui doivent coexister avec le système à l'étude). Un besoin ou une contrainte formellement exprimés par un maître d'ouvrage à un maître d'œuvre s'appellent une *exigence*.

Ce livre traite en particulier du dialogue entre maîtrise d'ouvrage et maîtrise d'œuvre, de la contractualisation de ce dialogue et des techniques de communication à même de le faciliter.

Dans la plupart des projets d'envergure, l'ensemble des exigences du maître d'ouvrage est consigné dans un document, le plus souvent appelé *cahier des charges*[1]. Celui-ci est donc un document de première importance, et il est placé sous le contrôle du maître d'ouvrage. De ce document dépendra en grande partie la réussite du projet.

À quoi ça sert ?

Le but d'un cahier des charges est d'apporter une vision commune du système à l'étude, et par voie de conséquence d'améliorer la relation entre client et fournisseur. Pour ce faire, il doit permettre de :

- **cerner le véritable besoin**, par une analyse systématique et exhaustive des souhaits, une synthèse auprès des parties prenantes et une mise en évidence des priorités ;
- **stimuler le fournisseur**, par une ouverture du champ de la recherche, par une incitation à l'optimisation technologique et une libération du choix de solutions, dans le cadre des contraintes imposées ;
- **favoriser le dialogue**, en soulignant les exigences intangibles, en délimitant les marges de manœuvre et en énumérant les contraintes, tout en s'ouvrant aux suggestions.

1. Même une démarche dite « agile » ne dispense pas de formaliser les exigences si l'on veut maintenir la traçabilité entre les objectifs du projet et le produit qui en résulte.

Le cahier des charges détermine le projet, ainsi que les conditions qui permettront de le mener à bien. Il fournit des informations sur le client, ses besoins, et renseigne le fournisseur sur l'usage qui sera fait et sur les aspects techniques et commerciaux. Le fournisseur partira de ce document de référence pour proposer une *solution informatique*.

Le cahier des charges fixe les objectifs et les contraintes du projet. Très souvent, il indique aussi les éléments à prendre en compte pour le mener à bien. Il permet au fournisseur d'avoir une idée claire sur ce qu'il peut ou ne peut pas proposer au client, et sur les objectifs et les contraintes internes au client, qui devront en tout état de cause être pris en compte. Il fournit au concepteur les renseignements nécessaires pour réaliser le projet. C'est à la fois un document de référence (pour tous les responsables impliqués dans le projet) et de dialogue entre le fournisseur et le client.

Un tel document est indispensable pour permettre à chaque acteur de prendre connaissance des règles du jeu qui s'imposent à tous, pour que chacun agisse en connaissance de cause. Un bon cahier des charges apporte un gain de temps et évite de s'engager dans une fausse direction. C'est la base de travail de tout projet de réalisation.

Conservons donc à l'esprit cette particularité : un cahier des charges a deux facettes. L'une est procédurière, contractuelle, normative. L'autre est un appel à la créativité et à l'imagination du fournisseur. Nous allons décrire ces deux facettes, car ce sont ces deux facettes mises ensemble qui donnent au cahier des charges sa puissance.

En quoi est-ce utile ?

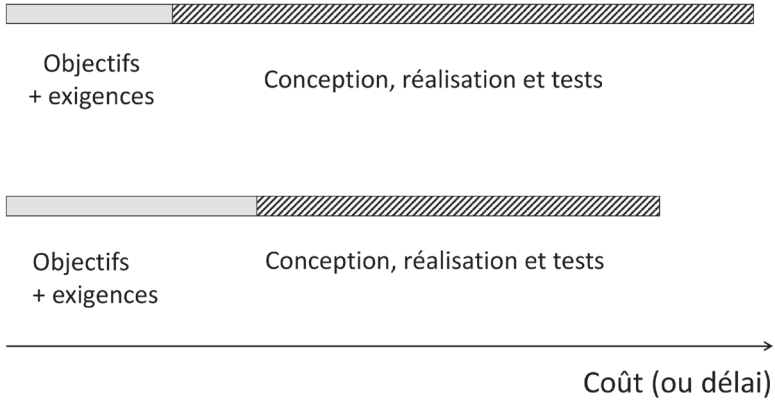
Il est difficile de convaincre une direction d'investir plus que d'habitude dans le recueil des besoins, l'élaboration du cahier des charges, et la gestion des exigences en cours de projet. Il s'agit pourtant d'un des meilleurs investissements qui soient.

Pour s'en convaincre, il suffit de se rappeler qu'une erreur de spécification d'exigences coûte en moyenne :

- deux fois plus cher à corriger lorsqu'elle est détectée en phase de conception ;
- trois fois plus cher à corriger pendant la phase de réalisation ;
- vingt fois plus cher à corriger en phase de tests ;
- cent fois plus cher à corriger sur un logiciel en exploitation.

Le coût d'une erreur croît exponentiellement en fonction de la phase du projet pendant laquelle elle est détectée.

Des centaines d'articles, de conférences et de présentations ont été faites sur l'utilité de détecter les erreurs au plus tôt. Comme cette expression de « détection d'erreurs » est un peu rebutante, tant pour les managers que pour les techniciens, nous lui préférons une image moins rigoureuse mais plus dynamique, celle de l'investissement et du retour sur investissement. Elle est illustrée par cette figure :



Investir dans le développement des exigences

Contenu du cahier des charges

Le cahier des charges doit décrire les *exigences*, c'est-à-dire *un énoncé formel de ce que l'on attend du système à venir*. Quel que soit le modèle de cahier des charges utilisé, on y indique :

- le pourquoi : les motivations à l'origine de la demande et surtout, l'objectif du futur système et les bénéfices qu'il devra apporter ;
- le qui : quels sont les acteurs concernés, en particulier, mais pas seulement, les futurs utilisateurs ;
- le quoi : ce que doit apporter le système aux utilisateurs pour atteindre le but spécifié ;

- une partie du comment : quelles sont les contraintes, liées au produit, liées à son utilisation ou à sa construction, mais sans indiquer d'éléments de solution ;
- une partie du quand, notamment les délais de livraison, qui sont une forme particulière de contraintes, sans entrer dans les détails du projet ;
- une partie du où, à savoir le périmètre géographique et organisationnel d'utilisation du futur système.

En d'autres termes, on doit décrire le service attendu de la part du système ou du fournisseur et l'usage qui sera fait du produit. On peut y ajouter des *contraintes* d'usage, et même des contraintes techniques, mais on ne doit pas y indiquer de *solution*.

Ce découpage du contenu (pourquoi, quoi, qui, quand, où… ainsi que les différents types de contraintes) a servi de base au modèle de cahier des charges que nous préconisons dans cet ouvrage.

La distinction entre description d'une exigence et description d'une solution, déjà très subtile pour des produits manufacturés, devient un vrai casse-tête dans le monde de l'immatériel. En témoigne l'exemple qui suit…

Prenons un exemple…

Un exemple provenant du monde industriel illustrera notre propos. Le maître d'ouvrage est un distributeur de produits pour fumeurs. Il constate que les briquets qu'il propose dans ses magasins sont trop chers, trop encombrants, et tombent souvent en panne. Il passe un appel d'offres pour diversifier son catalogue. Voici trois formulations différentes des exigences, suivies des réponses qu'elles sont susceptibles de déclencher :

1. « un briquet fiable, peu encombrant et peu coûteux » ;
2. « un dispositif de faible encombrement, fiable et peu coûteux permettant de faire du feu » ;
3. « un dispositif de faible encombrement, fiable et peu coûteux permettant d'allumer une cigarette ».

Dans le premier cas, la solution proposée sera un briquet. Le fournisseur a la liberté de répondre à la contrainte de coût en proposant un briquet jetable.

Avec la deuxième formulation, la solution proposée peut être un briquet ou une boîte d'allumettes. L'exigence ne formule pas de solution (d'où deux réponses très différentes) mais une fonction (faire du feu) et deux contraintes (fiable et peu coûteux).

Dans le troisième cas, la solution proposée peut être soit un briquet, soit une boîte d'allumettes, soit encore un allume-cigare d'automobile. L'exigence ne formule pas de solution (d'où trois réponses très différentes) mais une fonction (allumer une cigarette) et deux contraintes (fiable et peu coûteux).

La première formulation induit une solution et restreint le champ des possibles. Un briquet n'était pas la seule manière possible de satisfaire la clientèle. En utilisant le mot « briquet » dans le cahier des charges, on évoque déjà une partie de la solution.

Remarquons que le besoin initial n'était pas de faire du feu, mais d'allumer les cigarettes. En ce sens, la solution « allume-cigare » était une bonne solution. Si le distributeur avait précisé que le produit devait être vendable en magasin, aucun fournisseur n'aurait proposé d'allume-cigare… à moins qu'un fabricant astucieux et imaginatif ne propose un allume-cigare à piles qui tient dans la poche.

Le demandeur aurait pu ajouter qu'il a besoin d'un dispositif « autonome ». Méfions-nous des adjectifs ! Car, dans ce cas, la réponse aurait pu aussi bien être un allume-cigare (autonome, car ne nécessitant pas d'énergie autre que celle fournie par l'automobile) qu'un briquet (autonome, car l'énergie ne vient pas de l'extérieur du système).

Ce simple exemple, que l'on pourrait dérouler à l'infini, permet de mettre en lumière plusieurs points :

- il est très difficile d'exprimer les exigences sans induire de solution *a priori* ;
- une même exigence fonctionnelle peut donner lieu à des solutions très différentes ;
- l'expression de *l'exigence fonctionnelle* (ce que le système doit faire) peut être aisée à formuler, alors que l'expression des exigences de qualité ou de performance (parfois appelées « non fonctionnelles ») demande beaucoup de soin dans sa formulation : que signifient « fiable » et « peu coûteux » ?
- sans dévoiler de solution, il est souvent nécessaire d'exprimer des *contraintes d'environnement* de la solution ou des *contraintes d'in-*

terface : l'objet doit tenir dans une poche, dans la main, ou être alimenté en énergie par l'automobile ;
- méfions-nous des adjectifs ! À moins d'être définis dans un glossaire, dans une norme ou dans un texte de loi, ils ne veulent rien dire.

Nous avons pris jusqu'ici un exemple dans le monde de l'industrie manufacturière. Qu'en est-il du logiciel ? Les choses sont à la fois plus simples, car le matériau à notre disposition, le logiciel, est homogène et unique. Mais ce matériau est invisible, inodore et impalpable. De ce fait, les exigences de qualité (comme la fiabilité ou l'ergonomie) sont difficiles à exprimer. Pire encore, du fait de cette immatérialité, aucune contrainte ne va d'elle-même.

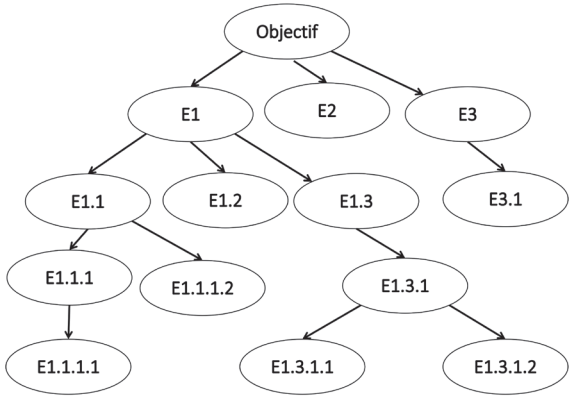
Dans le cas du briquet, la plupart des fabricants auront compris que le dispositif en question ne devrait pas s'enflammer spontanément dans la poche de l'utilisateur (même si, en toute rigueur, le cahier des charges devrait exprimer cette contrainte), ni tomber en panne après deux allumages, ni blesser celui qui le manipule. Mais dans le cas du logiciel, la fiabilité, la sécurité, la maintenabilité et l'utilisabilité sont des contraintes invisibles, donc mal maîtrisées. Sans contraintes fortes, les développements informatiques peuvent donner naissance à des produits monstrueux.

Arborescence hiérarchique des exigences

Un cahier des charges peut être plus ou moins détaillé, en fonction de l'ampleur et de l'objectif du projet. Le cahier des charges sera beaucoup plus détaillé si le but du projet est de faire développer sur mesure une application d'envergure nationale que s'il s'agit d'acquérir un logiciel sur étagère qui sera utilisé par une petite entreprise.

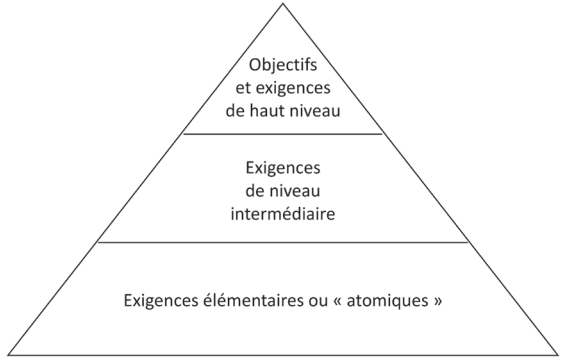
La structure d'un cahier des charges est arborescente. Au sommet, se trouvent les objectifs, qui sont les exigences de plus haut niveau. Elles se décomposent en exigences de plus en plus détaillées, et ce, jusqu'au niveau des ultimes exigences dites *élémentaires* (en anglais, *atomic requirements*) qu'il n'est plus possible de décomposer[2].

2. Atomic *vient du grec* ἄτομον *qui signifie insécable.*



Arborescence de fonctions

On peut se représenter les différentes sections d'un cahier des charges comme des « couches ». Dans l'idéal, lorsqu'on part de zéro, on élabore un cahier des charges « en largeur d'abord », c'est-à-dire par couches successives, en commençant par celle du haut.



Hiérarchie d'exigences

La couche du haut, le sommet de l'arbre, devrait en toute rigueur constituer un document autonome, que certains nomment cahier des charges préliminaire, document de vision ou rapport d'analyse préliminaire et que l'on appelle dans cet ouvrage *document de concept*.