LE STORY MAPPING

Visualisez vos user stories pour développer le bon produit

Jeff Patton
Consultant et formateur,
spécialiste des méthodes agiles

Préface de Claude Aubry

Traduit de l'américain
par Dominique Maniez

DUNOD

L'édition originale de ce livre a été publiée en anglais par O'Reilly Media Inc.
sous le titre *User Story Mapping*, ISBN 9781491904909.

Adaptation française des illustrations : WIP-Design

© Dunod, 2015
5 rue Laromiguière, 75005 Paris
www.dunod.com

ISBN 978-2-10-074030-7

Préface

Je n'ai jamais rencontré Jeff Patton. Je ne sais même pas s'il est de la famille du général. Mais je l'aime bien.

Ce n'est pas parce que j'ai échangé quelques tweets et mails avec lui, car cela ne remplace pas une conversation directe, ce que vous comprendrez comme jamais en lisant cet ouvrage. C'est la lecture de ses articles, puis de son livre, qui m'ont donné à croire que c'était un bon gars, ce Jeff.

Mon hypothèse se base, notamment, sur le fait qu'il cite volontiers le rock'n roll. Par exemple, dans l'article « Don't Know What I Want, But I Know How to Get It », il a réussi à mentionner les Sex Pistols et les Rolling Stones. Bon, dans une vidéo, il y avait aussi les Spice Girls, ce qui montre que Jeff a ses faiblesses, mais il le reconnait volontiers.

Il sera beaucoup question de conversations et d'hypothèses dans *Le story mapping*, mais d'abord de la pratique qui a donné le titre. Le story mapping est devenu célèbre suite à un seul article de Jeff, écrit en 2008 « *The New User Story Backlog is a Map* ».

Cela a suffi pour convaincre les agilistes que cette pratique serait un bon complément au « backlog Scrum ». Avec une liste d'éléments, tous au même niveau, le backlog est *plat*, tandis que la carte apporte, avec une nouvelle dimension, la « big picture » nécessaire aux personnes impliquées dans le développement.

Le story mapping apporte la carte et revisite la notion de story, centrale dans le développement agile.

Un jour, une stagiaire m'avait envoyé son mémoire sur les méthodes agiles. Le backlog y était présenté comme « *l'ensemble des stories rédigées par le Product Owner qui constitue la spécification* ». C'est un peu l'idée que peuvent en avoir ceux qui n'ont pas bien compris les notions de backlog et de stories. C'est d'abord à eux, tous ces gens qui sont perdus avec des centaines de stories, ceux qui pensent qu'elles doivent être rédigées, ceux qui pensent que c'est le Product Owner seul qui s'y colle, que s'adresse ce livre.

Patton y démonte les anciennes croyances des spécifications complètes, rédigées et figées dès le début, croyances qui perdurent, parfois même dans des organisations soi-disant passées au développement agile.

Dans ces organisations qui ont adopté Scrum, l'équipe multidisciplinaire mise en place est censée contenir toutes les personnes qui contribuent au développement du produit. Dans la pratique, le rôle de Product Owner s'avère délicat à jouer et les contours de ses responsabilités ne sont pas toujours bien définis. La place des autres gens côté métier : analystes métier, experts du domaine, designers d'interface, etc. est encore moins évidente. Ils ne sont généralement pas dans l'équipe Scrum, et leurs interactions avec les développeurs sont sources de difficultés.

Le livre de Jeff Patton leur apprend à collaborer véritablement, pour faire émerger les bonnes idées et une compréhension partagée du produit souhaité.

Les conversations, dont il est question depuis l'invention des user stories par Kent Beck, ont toujours fait sourire quand on explique que c'est cela la base d'une story. Avec ce livre truffé de retours d'expérience, les équipes sauront éviter qu'une conversation ne dégénère en une discussion de comptoir, et en tirer profit pour acquérir une perception commune du produit souhaité.

C'est en relisant l'ouvrage de Jeff que j'ai acquis la conviction de changer le titre d'un chapitre de mon livre sur Scrum. Je l'avais appelé « Définir le produit ». Mais la lecture de la prose pattonnienne m'a convaincu qu'un produit ne définit pas. Il se découvre.

La découverte par l'apprentissage validé, c'est l'occasion pour Jeff Patton de revisiter le *Lean Startup*. Le Lean Startup visant les ...start up, le livre montre comment utiliser ses idées quand on n'est pas une start up.

Il explique qu'une idée n'est qu'une hypothèse, tant qu'elle n'a pas été confrontée à ceux à qui elle est destinée, et validée par eux. Il montre comment exécuter cette boucle rapidement, pour apprendre, afin de décider quoi faire de l'idée de départ.

À cet égard, il intéressera toutes les personnes impliquées dans l'innovation.

En conclusion, vous l'aurez compris, ce livre va bien au-delà du story mapping, car il aborde toutes les facettes de la définition, pardon de la découverte d'un produit.

Avec ou sans ses cartes, Jeff raconte de bien belles histoires, qui plairont à tous ceux qui travaillent sur des produits. Il leur apporte des outils simples et efficaces et surtout leur ouvre l'accès à une nouvelle culture.

Le livre, dans sa version anglaise, a fait l'objet d'une session du club de lecture d'Agile Toulouse. Voici le résumé élaboré en commun par les participants :

« Jeff Patton nous donne à lire un livre d'une grande richesse. C'est indéniablement un ouvrage à mettre dans les mains de tout Product Owner. Ainsi ils pourront changer le monde, comme le suggère Jeff, ou à défaut appréhender pleinement leur rôle. »

Non Jef(f), t'es pas tout seul.

Claude AUBRY
Président de l'association Agile Toulouse,
Product Owner d'iceScrum,
auteur du blog Scrum, Agilité... et Rock'n'roll

Table des matières

Avant-propos

Live in it, swim in it, laugh in it, love in it /
Removes embarrassing stains from contour sheets, that's right /
And it entertains visiting relatives, it turns a sandwich into a banquet.
Tom Waits, « *Step Right Up* ».

Au départ, ce livre était prévu pour être une petite chose… en fait une brochure.

J'ai entrepris d'écrire un livre sur une pratique simple que j'appelle *story mapping*. Moi-même et pas mal d'autres personnes créons des cartes pour faciliter la collaboration et pour imaginer comment un produit sera utilisé.

Le story mapping permet de se concentrer sur les utilisateurs et leur expérience, ce qui améliore l'échange avec eux et, au bout du compte, engendre un meilleur produit.

La création d'une carte (story map) est vraiment très simple. En travaillant avec les autres, je décris la story d'un produit, en listant sur des Post-it® chaque grande étape que les utilisateurs identifient dans la story dans un flux qui va de la gauche vers la droite. Ensuite, on revient sur les détails de chaque étape et on en discute ; on rédige ces détails sur des Post-it® que l'on place verticalement en dessous de chaque étape.

Le résultat est une structure en forme de grille qui décrit une story, de la gauche vers la droite, et la décompose en détail, du haut vers le bas. C'est plaisant et c'est rapide. De plus, ces détails créent un *backlog des stories* de nos projets en développement agile.

Est-ce vraiment compliqué d'écrire un livre sur cette technique ?

Il s'avère néanmoins que même les choses simples peuvent être assez sophistiquées. Savoir pourquoi on veut créer une story map, anticiper sur ce qui va se passer quand on va en créer une, et connaître toutes les manières différentes dont on peut l'utiliser m'a pris beaucoup de temps et j'ai finalement rédigé plus de pages que je ne pensais sur cette pratique apparemment simple.

Si vous utilisez un processus de développement agile, vous remplissez probablement des backlogs avec des user stories. J'ai présupposé que dans la mesure où les stories sont une pratique courante pour vous, cela aurait été une perte de temps pour moi de les décrire dans cet ouvrage. Mais j'avais tort. Depuis une quinzaine d'années où les stories ont été décrites pour la première fois par Kent Beck, elles sont de plus en plus populaires, et de plus en plus mal comprises et mal utilisées. Cela m'attriste et, en plus, cela ruine tous les bénéfices que l'on retire du story mapping.

Je souhaite donc dans ce livre corriger le plus grand nombre possible d'idées fausses sur les stories et sur la manière dont elles sont utilisées dans le développement agile et lean. C'est la raison pour laquelle, pastichant la chanson de Tom Waits, j'ai transformé « ce sandwich en un banquet ».

Pourquoi moi ?

J'aime fabriquer des choses. Ce qui me motive, c'est la joie que j'éprouve à créer un bout de logiciel et à voir les gens l'utiliser et en tirer parti. Je ne suis pas un fanatique de la méthodologie, mais j'ai considéré que j'avais besoin d'apprendre à analyser ma pratique professionnelle pour l'améliorer. Et ce n'est qu'après une vingtaine d'années passées dans le développement logiciel que je commence à apprendre à enseigner le fruit de mon expérience. Et je suis conscient que ce que j'enseigne est une cible mouvante car ce que je comprends évolue toutes les semaines. Comment expliquer ces changements aussi rapides ? Toujours est-il que cela m'a empêché pendant des années d'écrire ce livre.

Mais le moment est venu.

Les stories et le story mapping sont des concepts très intéressants qui ont bénéficié à tant de gens, en améliorant leur vie et les produits qu'ils créent. Mais pourtant pendant que la vie de certains s'améliore, il y a de plus en plus de gens qui pataugent avec les stories et je souhaite que cela cesse.

Ce livre est ma contribution à cette cause et s'il améliore la vie, ne serait-ce que de quelques utilisateurs, j'en serais déjà heureux.

Ce livre est pour vous si vous pataugez avec des stories

Tant d'organisations ont adopté les processus de développement agile et lean, et le concept de story qui s'y rattache, qu'il est possible que vous tombiez dans les pièges suivants engendrés par les idées reçues sur les stories :

- Puisque les stories permettent de se concentrer sur la création de petites choses, il est facile de perdre la vision d'ensemble. On obtient alors souvent un produit monstrueux dont il apparaît clairement aux utilisateurs qu'il a été assemblé à partir de morceaux qui ne vont pas ensemble.
- Quand on crée un produit de taille significative, la création morcelée de petites choses laisse planer des doutes sur la date finale de réalisation ou sur la nature exacte du produit livré. Et si vous êtes le concepteur, vous vous posez également des questions.
- Comme les stories sont basées sur des conversations, les gens usent de ce prétexte pour éviter d'écrire quoi que ce soit. Ils oublient ensuite ce dont ils ont parlé et ce sur quoi ils se sont mis d'accord pendant leurs conversations.
- Comme les bonnes stories sont censées avoir des critères d'approbation, on se concentre sur l'obtention de critères d'approbation écrits, mais il n'y a pas toujours une compréhension mutuelle de ce qui doit être créé. Cela a pour conséquence que les équipes ne terminent pas le travail prévu dans les délais impartis.
- Comme les bonnes stories sont censées être écrites du point de vue de l'utilisateur et qu'il y a de nombreuses parties que l'utilisateur ne voit jamais, les membres des équipes prétendent que leur produit n'a pas d'utilisateurs si bien que les user stories ne peuvent pas fonctionner dans ce cas-là.

Si vous êtes déjà tombé dans l'un de ces pièges, je tenterai d'abord de chasser ces idées fausses qui vous ont induit en erreur. Vous apprendrez à prendre en considération la vision d'ensemble, à animer des conversations productives sur ce que les utilisateurs tentent de réaliser, et à concevoir des logiciels qui aident vraiment les utilisateurs.

À qui s'adresse cet ouvrage ?

La lecture de cet ouvrage est particulièrement recommandée si vous appartenez à l'une de ces catégories :

- **Les chefs de produits et les professionnels de l'expérience utilisateur** doivent lire cet ouvrage pour les aider à faire le lien, d'une part entre la réflexion sur les produits et l'expérience utilisateur, et d'autre part, la réflexion sur les projets tactiques et les éléments du backlog. Si vous avez des difficultés à passer de la vision d'ensemble que vous imaginez aux détails que vos équipes créent, le story mapping va vous aider. Si vous vous échinez à aider les autres à imaginer et à ressentir l'expérience des utilisateurs de votre produit, le story mapping vous sera utile. Si vous vous démenez pour trouver comment lier une bonne expérience utilisateur avec une pratique de conception de produit, ce livre vous facilitera la tâche. Si vous voulez intégrer une expérience de style lean start up dans vos méthodes de travail, cet ouvrage vous aidera.
- **Les Product Owners, les analystes et les chefs de projet** dans les entreprises d'informatique doivent lire ce livre pour les aider à faire le lien entre leurs utilisateurs en interne, les parties prenantes, et les développeurs. Si vous avez des difficultés à convaincre de nombreuses parties prenantes de votre entreprise

à être sur la même longueur d'onde, alors le story mapping va vous aider. Si vous vous démenez pour aider les développeurs à avoir une vision d'ensemble, les story maps vous faciliteront la tâche.

- **Les formateurs en développement agile et lean** qui ont pour objectif d'aider les individus et les équipes à s'améliorer doivent lire ce livre. Et au fur et à mesure que vous progresserez dans cet ouvrage, pensez à toutes les idées reçues que vos collègues véhiculent sur les stories. Utilisez les stories, des exercices simples et les pratiques décrites dans ce livre pour aider vos équipes à se professionnaliser.
- **Toute autre personne**. Quand on utilise un processus agile, on compte souvent sur les rôles comme celui de Product Owner ou d'analyste pour piloter une bonne partie du travail avec des stories, mais une utilisation efficace des stories nécessite que chacun comprenne les bases. Quand les gens ne maîtrisent pas les bases, on entend des plaintes sur le fait que les stories sont mal écrites, qu'elles sont trop grandes ou bien encore qu'elles n'ont pas assez de détails. Ce livre vous aidera, mais peut-être pas de la façon dont vous l'imaginez. Vous allez apprendre que les stories ne sont pas une méthode pour écrire de meilleures exigences, mais un moyen d'organiser de meilleures conversations. Ce livre va vous aider à comprendre les types de conversations que vous devez mener pour obtenir les informations dont vous avez besoin au bon moment.

J'espère que vous avez pu vous identifier à l'un de ces groupes que je viens de décrire et, si tel n'est pas le cas, donnez cet ouvrage à quelqu'un qui en fait partie.

Conventions utilisées dans ce livre

Comme je pense que ce n'est pas le premier livre sur le développement logiciel que vous lisez, vous ne devriez pas avoir de surprise.

Les **titres** à l'intérieur de chaque chapitre présentent le sujet traité : utilisez-les pour vous repérer ou bien pour sauter les passages qui ne vous intéressent pas dans l'immédiat.

Les points clés sont représentés de cette manière. Imaginez que je hausse la voix par rapport aux autres parties du texte.

Si vous faites une lecture rapide, lisez les points clés. Si vous les trouvez intéressants, lisez le texte qui les encadre, ce qui devrait les rendre plus intelligibles.

Les encadrés sont utilisés pour décrire plusieurs choses :

– Les concepts intéressants mais non critiques. Ils devraient être récréatifs, du moins telle était mon intention.

– Les recettes pour des pratiques spécifiques. Vous devriez pouvoir utiliser ces recettes pour vous assister dans l'apprentissage d'une pratique spécifique.

– Les stories et les exemples issus d'expériences d'autres utilisateurs. Cela devrait vous donner des idées que vous pourriez expérimenter dans votre organisation.

Cet ouvrage est organisé en sections spécifiques. Vous pouvez lire une section à la fois, ou bien employer les sections pour vous aider à trouver des idées pour résoudre un problème spécifique.

Structure de l'ouvrage

Il y a quelque temps, j'ai acheté une nouvelle imprimante laser couleur. En ouvrant le carton, j'ai vu qu'il y avait une brochure intitulée « *À lire en premier* » en grosses lettres rouges. Je me suis demandé si je devais vraiment lire cela avant toute chose parce qu'en général je ne respecte pas ce genre de consigne. Finalement, j'ai bien fait de lire les consignes car il y avait dans l'imprimante toute une série de pièces en plastique destinées à la protéger pendant le transport et si j'avais branché l'imprimante avant de les enlever, j'aurais pu l'endommager.

Cette anecdote peut sembler hors de propos, mais ce n'est pas le cas.

Ce livre contient un chapitre intitulé « *À lire en premier* » car j'y décris deux concepts majeurs et la terminologie associée que j'emploie tout au long de l'ouvrage. Je souhaite que vous intégriez ces concepts avant de commencer. Si vous entamez le story mapping avant de les avoir compris, je ne suis pas en mesure de garantir votre sécurité.

- **Le story mapping vu du ciel** – Dans les chapitres 1 à 4, vous allez prendre de la hauteur pour mieux appréhender le story mapping. Si vous utilisez les stories depuis un certain temps et avez déjà utilisé une story map, cette section devrait vous suffire pour aller de l'avant.
  Le chapitre 5 vous propose un chouette exercice pour vous aider à apprendre les concepts clés utilisés pour créer une belle story map. Faites-le en groupes dans votre entreprise et chaque participant comprendra de quoi il retourne. Et je vous promets que les futures cartes qu'ils vont créer pour vos produits seront bien meilleures après cet exercice.
- **Comprendre les user stories** – Les chapitres 6 à 12 racontent l'histoire qui se cache derrière les stories, la manière dont elles fonctionnent vraiment et le bon usage que l'on peut en faire dans les projets agiles et lean. Au sein des story maps, il y a beaucoup de petites stories que vous pouvez utiliser pour piloter le développement au quotidien. Même si vous êtes un vétéran du développement agile, je vous promets que vous apprendrez quelque chose sur les stories. Et si vous êtes néophyte en la matière, vous en apprendrez suffisamment pour surprendre vos collègues qui se prennent pour des gourous du développement agile.
- **Améliorer les backlogs** – Les chapitres 13 à 15 vous plongent dans le cycle de vie d'une story. Je traiterai des pratiques spécifiques qui vous aident à utiliser les stories et les story maps, en commençant par les grandes opportunités et en passant par le travail de découverte pour identifier un backlog rempli de stories qui décrivent un produit viable. Vous apprendrez à simplifier toutes les étapes du parcours grâce au story mapping et à d'autres techniques.
- **Améliorer la création** – Les chapitres 16 à 18 vous mènent au cœur de l'utilisation des stories d'un point de vue tactique, itération par itération ou sprint

par sprint. Vous apprendrez à créer des stories dans les délais, à les surveiller au cours de leur construction, à les terminer et à retirer un bénéfice intellectuel de chaque story que vous convertirez en un logiciel fonctionnel.

Je considère que les derniers chapitres de nombreux ouvrages sur le développement logiciel sont superflus et, en général, je les ignore. Malheureusement, je n'écris pas ce genre de chapitres si bien que vous allez devoir lire cet ouvrage en entier. Ma seule consolation est que chaque chapitre vous fournira des contenus utiles que vous allez pouvoir réinvestir immédiatement dans votre travail.

Note du traducteur

La traduction des termes du vocabulaire informatique de l'anglais vers le français est très souvent problématique car le traducteur doit s'inspirer de plusieurs sources qui fournissent parfois des résultats fort différents. Même si le traducteur s'appuie principalement sur l'usage des informaticiens professionnels, il doit aussi prendre en compte le travail des commissions de terminologie (www.culture.fr/franceterme), la terminologie prônée par les éditeurs de logiciels (Microsoft, par exemple, publie des glossaires multilingues pour toutes ses applications), ou bien les choix opérés par les premiers traducteurs de certaines technologies et que l'usage a consacrés.

D'une manière générale, la communauté francophone du développement agile semble avoir privilégié la conservation de la terminologie anglophone et c'est la raison pour laquelle nous avons pris le parti de ne pas traduire les termes comme *release*, *backlog*, *input*, *output* ou bien encore *outcome* (nous vous recommandons vivement à ce sujet de lire le chapitre intitulé « À *lire en premier* » où l'auteur explique les concepts d'*output*, d'*outcome* et d'*impact*). Pourtant, cette communauté n'est pas insensible, loin de là, aux problèmes de traduction et pour s'en convaincre, il suffit de fréquenter assidument le blog de Claude Aubry, éminent spécialiste de la méthode Scrum et qui a rédigé la préface de cet ouvrage. En effet, il existe plus de 150 pages sur le blog de Claude Aubry qui contiennent le mot « traduction » (pour vous en persuader, saisissez dans Google la requête suivante « traduction site:http://www.aubryconseil.com/ »), et certaines pages ont des noms évocateurs, comme « Agacement contre le franglais agile ». À la lecture de ce blog, on constatera que les spécialistes ne sont pas tous d'accord entre eux et que la traduction de certains termes varie également dans le temps…

Même si nous avons majoritairement respecté les usages du monde agile, nous avons néanmoins parfois choisi de traduire certains termes anglais en français, et nous les avons alors fait suivre du mot original en anglais mis entre parenthèses. Quand le contexte le justifiait, nous avons également inséré des notes de bas de page pour expliquer notre traduction. Nous espérons que nos choix terminologiques faciliteront la lecture de cet ouvrage à tous ceux qui sont déjà familiers des méthodes agiles.

À lire en premier

Cet ouvrage n'a pas d'introduction.

Oui, vous avez bien lu ! Vous vous demandez sans doute maintenant pourquoi le livre de Jeff n'a pas d'introduction. A-t-il oublié d'en écrire une ? Avec l'âge, est-ce qu'il commence à décliner ? Est-ce que son chien l'a mangée ?

Non, je n'ai pas oublié d'écrire une introduction pour ce livre. Et non, je ne commence pas à perdre mes facultés. Tout du moins, je n'en ai pas conscience. Et mon chien ne l'a pas avalée. C'est tout simplement que j'ai longtemps considéré que les auteurs passaient beaucoup trop de temps à me persuader que je devais lire leur livre, et que le lieu idéal de ces tentatives pour me convaincre se trouve être l'introduction. Cela a pour conséquence que ce qui est intéressant démarre dans la plupart des ouvrages à partir du chapitre 3, et je ne suis pas certain d'être le seul à avoir pris l'habitude de sauter les introductions.

En fait, cet ouvrage commence ici.

Et je ne vous autorise pas à sauter ce passage car il s'agit vraiment de la partie la plus importante. En fait, si vous ne deviez retenir de ce livre que les deux idées suivantes qui figurent dans ce chapitre, j'en serais déjà très content :"

> Le but de l'utilisation des stories n'est pas d'écrire de meilleures stories.
> Le but du développement de produits n'est pas de fabriquer des produits.

Laissez-moi vous expliquer !

Le jeu du téléphone arabe

Je suis sûr que vous vous souvenez de l'époque de votre enfance où vous jouiez à ce jeu étrange du téléphone où l'on murmure quelque chose à l'oreille de quelqu'un qui à son tour le répète en chuchotant à quelqu'un d'autre, et ainsi de suite, jusqu'à ce que la dernière personne du groupe révèle à voix haute le message complètement transformé, ce qui déclenche l'hilarité générale. Je joue encore en famille à ce jeu avec mes enfants quand nous sommes à table. Note à l'attention des parents : il s'agit d'une

excellente activité pour occuper des enfants qui meurent d'ennui à table à cause des conversations des grandes personnes.

Dans le monde des adultes, ce jeu se poursuit, sauf que l'on ne chuchote plus. On écrit de longs documents et on crée des présentations officielles que l'on transmet à quelqu'un qui en fera quelque chose de complètement différent de ce qui était prévu. Et cette personne utilise ce document pour créer d'autres documents qu'il donnera à différentes personnes. Mais à la différence du jeu auquel nous nous adonnions enfants, plus personne ne rit à la fin…

Quand les gens lisent des instructions, ils les interprètent. Si vous ne me croyez pas, laissez-moi vous montrer quelques exemples d'instructions qui sont vraiment très mal comprises.

Le décorateur du gâteau a écrit littéralement sur le gâteau la consigne qu'on lui avait donnée (*underneath that,* qui est d'ailleurs mal orthographié, signifie en dessous de ça).

Voici la couverture du livre de Jen Yates, *Cake Wrecks* (publié chez Andrews McMeel Publishing et que l'on pourrait traduire par *Naufrages en pâtisserie* ; je remercie Jen et John Yates pour m'avoir fourni ces photos). Ce livre tire son origine de son site Web qui est furieusement drôle :

http://www.cakewrecks.com/

N'y allez pas si vous n'avez pas au moins une heure à perdre ! Le site présente des photos de gâteaux décorés de manière improbable qui défient les lois de la raison, mais que Jen arrive quand même à expliquer. Un des thèmes récurrents de l'ouvrage et du site a trait aux exigences mal interprétées. Bien entendu, Jen n'emploie pas le terme **exigence** car elle n'est pas informaticienne, mais elle parle de *littéral*, car le lecteur lit et interprète littéralement ce qui est écrit. En regardant les photos, j'arrive à imaginer que quelqu'un écoute un client et retranscrit ce qu'il souhaite, puis transmet les consignes à une autre personne qui va décorer le gâteau :

CLIENT. – Bonjour, j'aimerais commander un gâteau.

EMPLOYÉ. – Qu'est-ce que vous voulez que l'on écrive dessus ?

CLIENT. – Pouvez-vous écrire « Au revoir Alice » en violet ?

EMPLOYÉ. – Absolument.

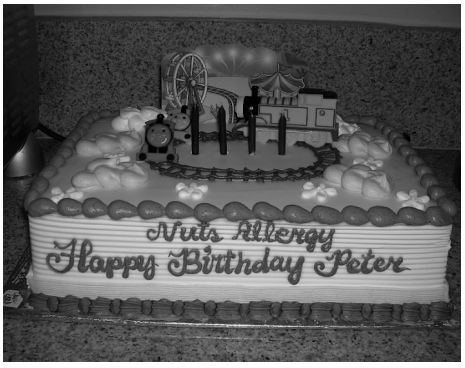
CLIENT. – Et mettre des étoiles autour ?

EMPLOYÉ. – Pas de problème. J'ai tout noté et je vais transmettre tout de suite à la personne qui va décorer le gâteau. Il sera prêt dans la matinée.

Le résultat est illustré ci-dessous.

Ici, le décorateur du gâteau a écrit littéralement « Stars » au lieu de dessiner des étoiles et il a écrit la consigne *in purple* (en violet) sur le gâteau.

En voici un autre. En développement logiciel, on appelle cela une exigence non fonctionnelle :



Dans cet exemple, le décorateur a écrit sur le gâteau que le client avait une allergie aux noisettes.

Il s'agit d'exemples amusants et on peut rire d'avoir perdu quelques dizaines d'euros pour un gâteau, mais il arrive parfois que les enjeux soient beaucoup plus importants.

Vous avez sans doute entendu parler de l'histoire du crash en 1999 de la sonde Orbiter qui a coûté 125 millions de dollars à la NASA[1]. Vous ne la connaissez peut-être pas, mais voici la chute de l'histoire : si un projet coule à cause des exigences et de sa documentation, il s'agit d'un projet de la NASA. En dépit de ses classeurs remplis d'exigences et de documentation, la sonde s'est écrasée parce que la NASA utilisait le système métrique pour ses mesures alors que les ingénieurs de chez Lockheed Martin employaient l'ancien système de mesures impérial pour développer les commandes de navigation des propulseurs du véhicule. Personne ne sait exactement où la sonde a fini sa course, mais certains pensent qu'elle tourne en orbite autour du soleil quelque part après Mars.

Paradoxalement, on s'efforce d'écrire pour communiquer plus clairement et ainsi éviter le risque de malentendu, mais trop souvent, c'est l'inverse qui se produit.

> Le partage de documents ne favorise pas la compréhension mutuelle.

Arrêtez-vous une minute et écrivez cette maxime. Écrivez-la sur un Post-it® et mettez-le dans votre poche. Envisagez de vous la faire tatouer quelque part de sorte que vous puissiez la voir quand vous vous préparez le matin pour aller au travail. Quand vous la lirez, cela vous aidera à vous souvenir de ce que je suis en train de vous raconter.

La compréhension mutuelle a lieu quand deux personnes comprennent ce qu'elles imaginent ainsi que leurs motivations. À l'évidence, il n'y a pas eu de compréhension mutuelle entre plusieurs décorateurs de gâteaux et les personnes qui leur ont communiqué des instructions écrites. Et à la NASA, quelqu'un d'important n'a pas compris la même chose que les ingénieurs qui travaillaient sur le système de guidage. Je suis certain que si vous avez été impliqué dans un développement logiciel pendant un certain temps, vous n'avez pas à chercher longtemps dans votre mémoire pour vous rappeler une situation où deux personnes croyaient être d'accord sur une fonctionnalité qu'elles voulaient ajouter au programme, pour finir par se rendre compte qu'elles imaginaient chacune quelque chose de complètement différent.

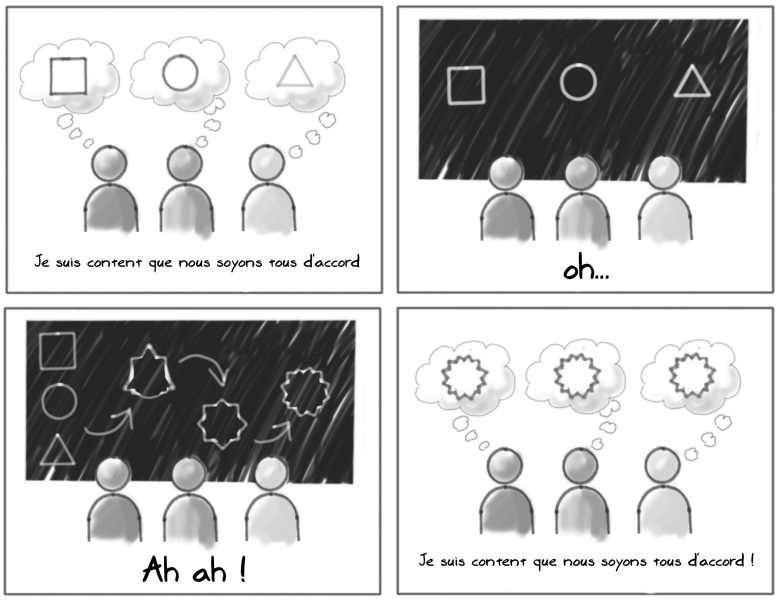
Favoriser la compréhension mutuelle est étonnamment simple

Un de mes anciens collègues, Luke Barrett, a réalisé une bande dessinée pour décrire ce problème. Pendant des années, j'ai vu Luke passer ces quatre images sous la forme de diapositives d'une présentation PowerPoint alors que je n'en voyais pas l'intérêt, les considérant comme trop évidentes. Il semblerait que je sois un peu obtus et il m'a fallu des années pour comprendre la manière dont cette bande dessinée illustre l'idée la plus importante sur l'utilisation des *stories* en développement logiciel.

Le concept est que si j'ai une idée en tête et que je la décris sous une forme écrite, il est fort possible qu'une tierce personne, quand elle va lire ce document, imagine quelque chose de différent. On peut même demander à chacun s'il est d'accord avec ce qui a été écrit et tout le monde vous répondra par l'affirmative.

1. Il y a de nombreux articles qui décrivent les causes du crash, dont notamment celui qui se trouve à www.cnn.com/TECH/space/9909/30/mars.metric.02/.

Cependant, si l'on est ensemble et que l'on se met à parler, vous pouvez me dire ce que vous pensez et je peux poser des questions. La conversation se passe encore mieux si l'on peut extérioriser nos pensées en dessinant des schémas ou en organisant nos idées à l'aide de fiches ou de Post-it®. Si nous prenons chacun le temps d'expliquer nos idées avec des mots et des images, nous favorisons la compréhension mutuelle. C'est précisément à ce moment-là que l'on se rend compte que nous avions compris les choses différemment. C'est pénible, mais maintenant on est au courant.



Il ne s'agit pas de dire qu'une personne a raison et qu'une autre a tort, mais que nous voyons les choses différemment. En combinant et en affinant nos idées différentes, nous finissons par créer une compréhension mutuelle qui inclut l'ensemble de nos meilleures idées. C'est la raison pour laquelle l'externalisation de nos idées est si importante. Nous pouvons redessiner des schémas ou déplacer des Post-it®, mais la chose la plus importante est que nos idées sont réellement en mouvement. En fait, nous faisons évoluer notre compréhension mutuelle, ce qui est extrêmement difficile à réaliser si l'on se contente de mots.

À la fin de la conversation, quand nous faisons référence à une fonctionnalité ou une amélioration, nous désignons en fait la même chose. Nous avons le sentiment d'aller de l'avant ensemble. En fait, c'est la qualité que nous gérons. Malheureusement, cela est impalpable. Il est impossible de voir ou de toucher la « compréhension mutuelle », mais vous pouvez la ressentir.

Arrêtez d'essayer d'écrire des documents parfaits

Il y a beaucoup de gens qui croient qu'il existe une façon idéale d'écrire la documentation et qui pensent que lorsqu'une personne lit des documents et parvient à une