Chapitre 4-1
Spécificités du Web

1. Introduction

Le Web, en tant que média, dispose d'un certain nombre de spécificités par rapport à un programme distribué ou client-serveur. C'est un média jeune, en pleine évolution, dont la principale caractéristique est la richesse de ses interfaces.

Cette particularité implique de faire appel à des ressources que l'on trouve rarement dans un service technique, les directeurs artistiques.

2. Création graphique et méthodes agiles

Historiquement, dans les projets gérés par un cycle en cascade, la création graphique était confiée à une agence de communication pilotée par la direction de la Communication, tandis que le reste était à la charge de la direction informatique qui coordonnait les intervenants des différents métiers impliqués en s'appuyant éventuellement sur un partenaire pour renforcer ses équipes internes ou sous-traitait tout ou partie des travaux.

L'agence faisait valider par la Communication des écrans représentant le futur site Internet, l'agencement des différentes fonctionnalités, les parcours utilisateurs, bref, elle décrivait ce que serait l'expérience utilisateur, c'est-à-dire l'émotion qu'il éprouve en parcourant le site Internet et par là même, l'association positive ou négative qu'il réalise entre cette émotion et la marque. Ce faisant, elle fixait des contraintes pour la réalisation technique ultérieure. Ces dernières étaient plus ou moins fortes selon la maîtrise qu'a cette agence du média Internet.

Casser ce modèle pour le rendre agile représente parfois un véritable casse-tête. Remplacer l'approche séquentielle par une approche itérative implique de résoudre plusieurs dilemmes.

2.1 Premier dilemme

Il n'est pas envisageable de réaliser une création graphique sans avoir une vision exhaustive des fonctionnalités. Comment savoir ce qui doit être mis en avant sur la page d'accueil si le périmètre fonctionnel doit pouvoir évoluer librement au cours de la production ?

D'un autre côté, donner libre cours aux créatifs contribue à fixer le périmètre fonctionnel de la page d'accueil, ce qui remet en cause l'approche agile dans son essence.

Se contenter de poser une charte graphique, puis laisser le champ libre à des ingénieurs pour la réalisation, revient à risquer de sacrifier l'expérience utilisateur.

Ce dilemme est récurrent dans les équipes s'appropriant une méthode agile et ce n'est que la partie émergée de l'iceberg. Il ne porte en effet que sur l'analyse du présent, mais ne considère pas du tout un autre aspect de la production web, la dépréciation liée à l'évolution technologique induisant une évolution des usages.

2.2 Second dilemme

Si autrefois le *designer* d'une solution web devait prendre en compte des contraintes de navigateur et de résolution d'écran représentant une quinzaine de configurations différentes, aujourd'hui il doit élargir sa réflexion aux tablettes et téléphones mobiles, démultipliant à tel point les configurations qu'il est impossible de toutes les tester. L'émergence de nouveaux supports et de nouveaux modes de consommation de l'information n'étant qu'à ses débuts, il n'est même pas possible d'envisager les configurations possibles à court terme. Qui peut prévoir de quoi seront capables les futures montres, lunettes ou télévisions connectées ? Une solution courante consiste à décliner des interfaces spécifiques pour chaque média en utilisant différentes feuilles de style, mais cette solution risque de n'être que temporaire.

Si la multiplication des supports implique d'adapter les modèles de production par un découplage fort entre le code et la présentation, la multiplication des interfaces risque de bouleverser un paradigme déjà ébranlé. L'apparition des écrans tactiles, de la réalité virtuelle ou augmentée, ainsi que des fonctions de commande vocale, introduites notamment par Google, représentent des enjeux considérables en termes d'expérience utilisateur.

Alors que la multiplication des feuilles de style destinées à adapter le contenu aux différentes interfaces complexifie la maintenance évolutive, la prise en compte de nouvelles interfaces risque de rendre toute solution impossible à maintenir. Les entreprises qui n'auront pas anticipé ces opportunités risquent de se voir confrontées à des choix pénibles : maintenir des solutions en double, ou bien faire l'impasse sur de nouveaux modes de consommation de l'information, au risque de prendre un retard considérable sur leurs concurrents. Toutes les sociétés françaises ont déjà vécu cette expérience lorsqu'Internet a supplanté l'historique Minitel. Il n'est pas sûr qu'elles souhaitent revivre cette expérience.

Faire du shopping depuis son téléviseur 3D dans une réalité augmentée, puis valider et payer sa commande directement depuis son téléphone n'est pas une prospective à dix ans, c'est réalisable aujourd'hui.

Nous avons donc affaire à un double dilemme, organiser le mode de production de solution web en fonction de nouvelles méthodes mais également prendre en compte le fait que les interfaces évoluent aussi vite que les processus métier, voire plus vite. Tout ceci, sans compter les évolutions des serveurs d'applications, les dépréciations technologiques et les risques de sécurisation des applications.

2.3 Commençons par ce qu'il ne faut pas faire

Il ne faut pas rester sur l'ancien modèle et considérer que la création graphique représente une contrainte à respecter, et qu'il sera éventuellement possible de l'adapter *à la marge*, avec l'accord de l'artiste l'ayant réalisée. Cette approche est à l'opposé de l'agilité. Elle est même à l'opposé du bon sens. Faire de l'agilité en acceptant que la créativité soit une contrainte, plutôt qu'une source d'innovation relève quasiment de la dissociation mentale.

Tenter un mix revient aussi à s'égarer de la philosophie de base de l'agilité. Ainsi, créer des ateliers ou des réunions spécifiques avec des directeurs artistiques, ergonomes, spécialistes de l'UX (expérience utilisateur), chefs de produits auxquels on convie le Product Owner et éventuellement le Scrum Master, réduira la créativité de l'équipe et engendrera des tensions.

Généralement, ces ateliers servent à concevoir un concept graphique à partir de la charte graphique et de différentes pistes préalablement retenues (études concurrentielles ou stratégie d'émergence).

Le problème est double. Ce concept graphique pose des contraintes qui refréneront la créativité de l'équipe au lieu de la nourrir. Elles sont validées par le Product Owner et les artistes, mais bien souvent le Scrum Master n'est là que pour apporter une garantie de faisabilité. Second effet néfaste, l'équipe étant exclue de la conception, elle se sent au mieux dévalorisée, au pire, trahie par le Scrum Master.

À l'opposé, concevoir la solution sans contrainte, puis faire appliquer la charte graphique sur le résultat une fois que celui-ci est parfaitement fonctionnel peut amener à une impasse. Ceux qui ont testé cette expérience, croyant que ce serait une bonne idée, vous déconseilleront vraiment d'essayer le design non contraint.

2.4 Utiliser pleinement Scrum

Plutôt que de conduire des ateliers, puis des réunions pour rendre compte de ces ateliers et organiser des comités pour entériner les décisions, utilisons pleinement la revue de Sprint Scrum.

Lors de cette réunion, le Product Owner et l'équipe Scrum présentent à l'ensemble des parties prenantes ce qu'ils ont réalisé dans le Sprint en cours et le Product Owner indique ce qu'il lui semble le plus judicieux de réaliser lors des prochains Sprints. Lorsque le résultat présenté correspond aux attentes de chacun, il est possible de poursuivre le plan de release tel qu'envisagé.

Il sera toujours plus facile de faire réagir les utilisateurs en leur montrant un produit en cours d'élaboration et en les faisant participer à l'élaboration, directement en présence des développeurs, que de conduire de multiples réunions.

Il ne reste plus qu'à faire travailler en bonne intelligence des designers et des techniciens.

2.4.1 Première solution : découplage du front office

Il s'agit de mettre en place une équipe qui sera exclusivement dédiée à la conception et aux développements des interfaces. On parle alors de développements front office, que l'on retrouve souvent sous l'acronyme DFO. Cette équipe est composée de spécialistes des interfaces maîtrisant le responsive web design (RWD), de développeurs maîtrisant des frameworks, tel Angular, ou un langage spécifique tel Sass (langage informatique permettant de générer des feuilles de style, acronyme de *SyntactticallyAwesome Style Sheets*), d'un directeur artistique, d'infographistes et éventuellement d'un ergonome. La présence d'un ingénieur peut renforcer le savoir-faire technique et faciliter les échanges avec les autres équipes.

Cette approche offre à la fois une opportunité et une contrainte. Elle implique un découpage produit particulier avec des lots dédiés aux interfaces et un fort découplage applicatif. Un travail d'architecture est essentiel en amont et lors de la conduite du projet. En revanche, elle permet de produire une solution qui sera très adaptable aux évolutions technologiques, notamment en termes d'interfaces. Elle permet aussi de modifier les interfaces indépendamment des règles métier (attention, l'inverse n'est pas avéré).

Elle oblige à avoir deux Product Owners distincts pour un même produit, ou bien un Product Owner qui fait le va-et-vient entre plusieurs équipes.

Enfin, elle implique une planification des lots et une organisation drastique. Dans une organisation ayant adopté l'agilité à l'échelle tel que Scrum2Scrum ou SAFe, cette organisation sera facilitée.

2.4.2 Seconde solution : le brassage culturel

Cette approche permet de bénéficier pleinement des avantages de la gestion de projet agile, mais également de ceux de la dynamique de groupe.

Elle consiste à mixer les compétences artistiques et techniques au sein d'une même équipe, ingénieurs et créatifs travaillant de concert à élaborer une solution complète, s'appuyant sur les idées et compétences ainsi réunies au sein d'une même équipe.

Elle offre par ailleurs des bénéfices à court terme mais également à long terme.

À court terme, l'équipe apporte ensemble une solution à la *user story* relatée par le Product Owner. Elle exprime magistralement le concept du 1+1=3. Là où l'ingénieur aurait disposé une suite de boîtes à remplir pour réaliser un formulaire fastidieux, le créatif proposera des sliders, une commande tactile ou vocale, chacun rebondira sur l'idée de l'autre pour arriver à une solution efficace et élégante, prenant en compte l'expérience utilisateur tout en respectant les contraintes techniques. On constate alors que le rôle du Product Owner a tendance à s'inverser. Lui qui poussait l'équipe à trouver des solutions élégantes, aura tendance maintenant à refréner sa créativité sur les fonctions non critiques.

Le bénéfice à long terme le plus évident est la compétence des équipes. Ce brassage produit un effet formateur exceptionnel, car l'information est véhiculée non plus par une autorité détentrice d'un savoir, mais par un membre de sa propre tribu. Le partage et l'appropriation sont beaucoup plus profonds. L'ingénieur commence à penser en créatif et vice versa.

En revanche, elle implique d'avoir pour chaque sprint des tâches de design pour occuper le plus efficacement possible les ressources créatives, ou d'avoir des ressources fortement polyvalentes, acceptant de sortir du cadre strict de leur mission afin d'atteindre l'objectif d'un Sprint.

3. Découplage des interfaces

Dans la catégorie des bibliothèques graphiques permettant de construire des interfaces, si Swing est considéré comme un composant léger, comparé à AWT, alors les interfaces web font figure de poids plume. L'usage des feuilles de style et du DHTML permettent nativement un couplage très faible entre les règles métier et les interfaces.

Cependant, certains langages permettent une architecture brouillon, voire pas d'architecture du tout. On retrouvera alors mêlés au sein d'un même script des règles métier, des interfaces et des traitements de surface. L'effet Flaccid Scrum décrit par Martin Fowler peut alors prendre des proportions dantesques. À vouloir produire rapidement un produit utilisable, dans l'objectif de valider des hypothèses, les développeurs ont parfois tendance à prendre des raccourcis qui se traduisent par une dette technique colossale.

Cette démarche a tendance à se produire lorsque la pression sur les délais est très forte et lorsque l'équipe est dans un esprit de défiance, bien souvent parce qu'elle se sent en difficulté.

L'approche que nous venons d'évoquer, consistant à former une équipe DFO (développements front office), permet de minimiser cet effet et force un découplage de manière systémique. Mais pourquoi insister autant sur le découplage et particulièrement celui des interfaces ?

Premièrement, parce que lors du processus itératif, les demandes de modification des utilisateurs portent principalement sur les interfaces. À moins que les règles métier aient été mal exprimées dans les User Stories, elles ne doivent évoluer qu'à la marge lors des itérations successives. Les sprints de remboursements de dette technique sont généralement plus importants que ceux consistant à intégrer des évolutions fonctionnelles.

Si les interfaces sont fortement découplées, le risque de régression sur les processu métier est alors très faible, voire inexistant.

Chapitre 4
L'approche classique du projet

1. Le modèle en cascade est une référence

Le modèle en cascade est le standard incontournable de la gestion de projet. Bien que souvent opposé aux méthodes agiles ou critiqué par leurs promoteurs, il continue à s'appliquer avec un taux de réussite certain.

Ce modèle qui repose sur une succession de phases est d'une logique simplissime. Chaque phase débute à l'achèvement de la précédente de façon à développer et enrichir un livrable en élaboration progressive. Ainsi, les spécifications sont la traduction de l'expression de besoins, le code source devient la matérialisation des spécifications, les tests sont réalisés sur un programme assemblé (on pourrait dire compilé), le logiciel est déployé et installé une fois le programme qualifié à l'issue des tests, etc.

Ce modèle de développement est au moins aussi ancien que la programmation informatique elle-même. Et ce constat amène deux remarques objectives.

Tout d'abord, un modèle de développement n'est pas une méthode de gestion de projet. Un enchaînement de phases, d'activités et de tâches, aussi logique soit-il, ne vaut pas suivi du planning, gestion des aléas ou prise de décision.

Deuxièmement, l'ancienneté de ce modèle et la persistance de son usage témoignent certes de son adéquation à une certaine typologie de projet, mais aussi par différence de ses limites.

Ces deux remarques aident à positionner respectivement le modèle en cascade et les méthodes agiles. Ces dernières ne s'inscrivent pas en opposition complète, puisqu'elles intègrent en partie l'enchaînement des phases du modèle en cascade.

Il est donc utile de bien maîtriser la conduite d'un projet selon une approche classique, d'abord pour l'appliquer à la typologie de projet idoine (cf. chapitre La prise en compte du risque), mais aussi pour faire la part des choses dans le recours aux méthodes agiles.

Remarque

Sans se hasarder à l'indication d'un pourcentage, un très grand nombre de projets conduits en waterfall réussissent toujours. Les méthodes agiles ne sont donc pas l'apanage de la modernité ni le seul moyen de mener un projet à son terme.

1.1 Estimer par anticipation pour planifier

Commençons par expliquer le terme éponyme cascade (*waterfall* en anglais). Imaginons un cours d'eau qui dévale la montagne, s'engage dans la vallée alpine attenante, poursuit et serpente dans la plaine, s'étend dans l'embouchure maritime et finit dans la mer. À la source, on jette à l'eau un bouchon en liège, celui-ci cheminera sans difficulté pour terminer sa course dans l'immensité de l'océan. Le parcours inverse, en revanche, est beaucoup plus coûteux, à l'image des saumons sauvages qui remontent les torrents et finissent épuisés à l'endroit même de leur naissance.

En gestion de projet, on indique que le passage aller d'une phase à l'autre ne coûte pas plus que la réalisation d'une phase, mais que le sens inverse multiplie le coût par 10. Autrement dit, tout changement en cours de route se traduit par un budget multiplié par 10, 100, 1000… et peut finalement remettre totalement en cause la faisabilité du projet si la modification est tardive.

On comprend donc que le changement est à éviter ou à anticiper autant que possible. Le modèle en cascade fonctionne très bien sur des spécifications stables et une plateforme technique éprouvée. Dans ces conditions, on est tout à fait capable d'estimer la durée de chaque phase, d'en apprécier la charge en jour-homme et de planifier le projet en enchaînant chaque tâche avec ce qu'il faut de marge de sécurité.

Il existe de nombreux abaques et quantité d'heuristiques pour planifier ce type de projet (42 % de tests, 40 % d'industrialisation, 20 % de chef de projet…). Il n'y a pas de recette universelle tant que les spécifications restent stables. Nous étudierons les techniques d'estimation et de planification au chapitre Planification, chiffrage et suivi au quotidien.

1.2 La contrainte de spécifications stables

Interrogeons-nous sur les conditions de spécifications stables. Il arrive fréquemment que l'expression de besoins évolue pendant la phase de cadrage. La maîtrise d'ouvrage (MOA), parfois guidée par un assistant à maîtrise d'ouvrage (AMOA), va prioriser les besoins et éclaircir les fonctionnalités attendues. Il est naturel et même souhaitable de conduire cette activité de façon itérative car la mise au point d'un cahier des charges ne peut raisonnablement pas réussir d'un seul trait.

Comme la rédaction de spécifications détaillées s'avère coûteuse, et puisqu'on ne peut pas éviter quelques itérations dans le cadrage des besoins du projet, on a l'habitude de réaliser des spécifications générales, une sorte d'intermédiaire entre une expression de besoins forcément incomplète et des spécifications détaillées nécessairement stables et cohérentes. Car dans le modèle en cascade, le coût et le délai sont portés par la réalisation de ces spécifications détaillées et, nous l'avons vu, tout changement s'avère dispendieux.

Comment dès lors garantir la stabilité des spécifications ? Tout d'abord, en laissant suffisamment de temps à la MOA pour mûrir son cahier des charges. Mais il faut aussi recueillir des validations formelles (*sign-off*) de chaque élément des spécifications. Voici les étapes habituelles pour y parvenir :

– Rédaction des spécifications.
– Revue dirigée (commentée).
– Révision et ajustement.
– Revue finale.
– Validation formelle (*sign-off*).

Au terme du *sign-off*, il est convenu que toute modification est exclue, les demandes de changement devront attendre la fin du projet.

1.3 Gestion des risques et gouvernance dans un projet waterfall

Dire qu'un projet réalisé selon le modèle en cascade n'admet aucun aléa est sans doute excessif, mais l'équipe projet doit rester vigilante face aux demandes de changements apparemment mineurs et qui peuvent avoir de grandes conséquences.

Pour éviter des tensions entre les parties prenantes du projet (aussi désignées *stakeholders*) à l'initiative de modifications du besoin et les équipes en charge de la réalisation, le chef de projet dispose de deux outils.

Tout d'abord, la gouvernance projet définit les règles de fonctionnement du projet, précise les rôles et responsabilités de chacun, détaille les phases du projet où les besoins sont acceptés, mais aussi documente les étapes indispensables à la réalisation du livrable. La gouvernance établit les instances de suivi et les instances de décision (on pourrait dire arbitrage).

Une bonne gouvernance évite les raccourcis « c'est pourtant une demande très simple » et sécurise l'enchaînement des actions aboutissant à la formation du livrable.

Le second outil est la gestion des risques ; elle permet d'objectiver tout au long du projet la situation réelle du projet et les possibilités d'aboutir pour un périmètre donné. En d'autres termes, la gestion des risques fournit les éléments pour arbitrer des demandes de changements. Mais elle a une autre vertu ; les demandes de changements « en cours de route » sont le fruit d'une décision d'adresser des risques dûment qualifiés, ce qui relègue de fait au second plan des demandes d'évolutions (le plus souvent fonctionnelles) effectuées au fil de l'eau faute d'être parvenues à concentrer tous les besoins en phase de cadrage.

Ainsi, la gouvernance et la gestion des risques autorisent certains aménagements du plan projet en cours de route, ce qui est une bonne nouvelle pour le chef de projet mais peut paraître étonnant pour les aficionados des méthodes agiles.

Les indicateurs clés Qualité, Coût et Délai seront en fin de compte les témoins de la bonne exécution d'un projet. S'ils dévient franchement de leur valeur nominale, le projet est peut-être en train de dériver à cause de demandes de changements trop profondes ou trop impactantes. Dans pareille situation, le chef de projet doit proposer au comité de pilotage de lotir les évolutions en créant des paquets de demandes à traiter sur un cycle de projets en cascades. L'autre possibilité est d'hybrider le projet en cascade avec un flux de travail en mode agile. Cette approche est très intéressante dans le cas d'un projet dont le périmètre évolue sensiblement en cours de route, ces modifications étant le fruit de conditions extérieures et non d'un manque d'anticipation de la part de l'équipe projet.

2. Les phases du projet

2.1 L'expression de besoins et le cahier des charges

Ces deux termes sont analogues, bien qu'en pratique une expression de besoins soit le point de départ à l'élaboration d'un cahier des charges plus complet.

Comme nous l'avons déjà évoqué, le cahier des charges est plus souvent décrié qu'encensé. Et pourtant, il s'agit d'un document indispensable à la réalisation d'un projet. Le principal problème lié à sa rédaction tient dans l'interprétation réputée équivoque que l'on pourrait lui prêter, suivant que l'on est du côté de la maîtrise d'ouvrage ou de la maîtrise d'œuvre. En effet, les clients rencontrent parfois de réelles difficultés à exprimer clairement leurs besoins (on entend même parfois que la meilleure expression correspond au logiciel réalisé). Il arrive aussi que, rompus à l'exercice, les clients restent délibérément flous dans leurs réponses, cherchant ainsi à maximiser leurs marges de manœuvre et à différer leurs choix.

La société en charge de la réalisation poursuit précisément l'objectif opposé : plus vite elle identifie le périmètre de la solution, meilleur sera le rendement du projet, par le truchement de la capitalisation du savoir-faire.

La situation paraît donc inextricable, et pourtant elle s'impose rapidement car le cahier des charges reste la principale annexe technique d'un contrat de développements informatiques.

Il apparaît clairement que les incompréhensions proviennent du caractère « figé » prêté au cahier des charges. Il serait une fin en soi. Or il n'en est rien et la solution tient dans une autre lecture de la remarque faite ci-dessus. Si la meilleure expression d'un besoin client est supportée par un logiciel, elle n'est qu'une dérivation d'un ensemble de spécifications qui découlent toutes du cahier des charges.

Autrement dit, le cahier des charges n'est nullement un document figé mais plutôt un canevas que le processus d'analyse va compléter et transformer jusqu'à la production du logiciel.

Et tous les processus d'analyse commencent par donner une priorité aux éléments du projet. Il n'y a donc pas de raison de presser le client pour bloquer des éléments qui pourront être définis ultérieurement, sans remettre en cause le déroulé des opérations.

2.1.1 Le contenu d'un cahier des charges

Dès lors, que trouve-t-on dans un « bon » cahier des charges ? Cela dépend évidemment du niveau d'avancement de réflexion de son rédacteur, qu'il soit représentant des utilisateurs (maître d'ouvrage) ou prestataire (maître d'œuvre).

Contexte et objectifs stratégiques

Cette section reprend les grandes lignes du *business case* du demandeur ; la situation existante, la cible. Il est habituel de ne pas dépasser quelques lignes pour rappeler le contexte, la rédaction est souvent courte et synthétique.

Objet

Est énumérée dans cette partie la portée du projet confié au maître d'œuvre. Un lotissement peut être mis en place si les responsabilités et les domaines de compétences sont trop vastes. La longueur de cette section varie considérablement d'un projet à l'autre.

Domaine métier

Le domaine métier apporte des précisions sur les règles applicables au projet. Ces règles sont décrites textuellement et font référence à des documents annexes officiels.

Des modèles de base de données constituent également un bon moyen de présenter les aspects métiers, surtout s'ils sont conceptuels (MCD).

Périmètre fonctionnel (généralement confondu avec l'expression de besoins)

La description du périmètre fonctionnel s'appuie sur quantité de formalismes : diagramme des cas d'utilisation, scénarios, diagrammes de flux de travail, cartographies fonctionnelles…

Il n'est pas rare, au moment de l'élaboration du cahier des charges, d'avoir suffisamment de matière pour développer cette partie du cahier des charges. C'est une bonne démarche dans la mesure où elle clarifie souvent les responsabilités ; cependant, une certaine souplesse doit être conservée quant à son évolution au cours du projet. De plus, il ne faut pas trop anticiper les phases d'analyse pour ne pas s'orienter vers une solution ne répondant pas au besoin réel. Certaines parties peuvent donc être délibérément esquissées alors que d'autres font l'objet d'une description beaucoup plus poussée.