

Sauvetage de projet

Pompiers volontaires (auteurs) :
Jean-Guillaume DUJARDIN &
Nicolas PEGUIN



TheCodingMachine™
TCM://

*Un projet en panne ? Un projet mal
accepté par les utilisateurs ? Un
planning qui dérape trop ?*

*Les raisons du ou des problèmes
peuvent être nombreuses :
erreurs de conception, prestataire
démissionnaire ou incompetent,
imprécisions sur le besoin...*

*En cas de crise, TheCodingMachine
établit rapidement un diagnostic et
formalise, en conséquence, un plan
d'actions pour remettre rapidement
le projet sur des rails ou, si cela
s'avère nécessaire pour stopper le
projet et le reprendre sur des bases
consolidées.*

**SEULEMENT
EN CAS
D'URGENCE :**

01 71 18 39 72

préambule

Pourquoi ce livre blanc ?

TheCodingMachine a eu souvent l'occasion d'aider des clients à sauver des projets. Ces situations critiques nous ont permis d'appréhender les origines de ces dérapages, de trouver des solutions et de les mettre en œuvre.

Le "sauvetage de projet" est un sujet passionnant à plusieurs titres :

- Il nécessite de mettre en pratique **une très large palette de compétences**. Des compétences humaines pour distinguer objectivement les problèmes réels de ceux qui sont fantasmés, des compétences techniques afin de pouvoir plonger au cœur des applications, sans oublier la créativité pour imaginer ou trouver les bonnes solutions.
- C'est une activité exigeante. Les attentes des clients sont fortes. Ces missions nécessitent de déployer une énergie importante dans un délai record afin de convaincre les différentes parties et mettre en place les bonnes actions correctrices.

Ce livre blanc est là pour vous livrer une synthèse de nos expériences, les écueils à éviter et vous exposer les méthodes que nous avons appliquées.

préambule

A qui s'adresse ce livre blanc ?

Si vous estimez que votre projet est en danger ou en pleine tourmente. Si, plus simplement, vous souhaitez approfondir la notion de risques sur un projet, ce livre blanc est fait pour vous.

Avertissement :

Ce livre blanc est le fruit de notre expérience. Vous y trouverez peut être des défauts ou en soulèverez des limites. D'autre part, toutes les solutions présentées, parce qu'elles ne s'appliquent pas à toutes les situations, ne sont pas garanties.

Ce livre a pour objectif de vous donner des idées et de partager nos expériences.

N'hésitez pas à nous faire part de votre expérience !

Note sur les auteurs et la version :

Ce livre blanc a été écrit par Jean-Guillaume DUJARDIN et Nicolas PEGUIN.

Date de publication de la première version : septembre 2012.

Introduction

Plan du document :

Le **premier volet** du document détaille les risques qui menacent votre projet et vous donne des repères forts pour les isoler.

Votre projet n'avance plus ? Est dans une situation critique ? Le **deuxième volet** propose de vous donner des clés pour sauver un projet en distinguant différentes étapes :

1. Est-il possible de sauver le projet ?
2. Quelles sont les pistes pour résoudre les problèmes humains ?
3. Comment résoudre les problèmes techniques ?

Un projet évolue dans un environnement hostile



Un projet évolue dans un environnement hostile

La différence entre le risque & une situation d'échec

Le risque est susceptible d'être géré. Il est donc possible de mener des actions correctives. A l'inverse, une situation d'échec est une accumulation de risques non gérés qui aboutissent souvent à un blocage global du projet.

Si vous vous posez la question de savoir si votre projet est totalement planté ou si vous êtes simplement en train de gérer un risque, vous êtes à une étape cruciale du développement de votre projet. En général, plus on se rend compte tard du plantage, plus c'est grave. Autrement dit, plus on accumule les risques au fur et à mesure du projet, plus le sauvetage s'annonce difficile !

Un projet évolue dans un environnement hostile

Les dangers qui vous guettent !

Chaque étape d'un projet comporte des risques. La plupart des risques qui menacent un projet sont connus à l'avance même si certains apparaissent dans des situations particulières parce qu'ils sont liés à une problématique unique.

Néanmoins, la manière de structurer la réponse aux besoins des utilisateurs (expression de besoin et cahier des charges) et le choix du prestataire constituent des étapes clés pour limiter tout débordement sur un projet à venir. Vous devrez y porter une attention particulière.

Les étapes	Risque(s)
Besoin, cahier des charges	Le recueil des besoins, les entretiens avec les utilisateurs ou les clients (surtout si l'on est sur un business model web) sont indispensables.
	Le risque est de rédiger un document complexe et pas assez complet. Si vous organisez une consultation et que vous n'avez pas de compétences techniques, laissez l'opportunité au prestataire de concevoir la solution technique la plus adaptée et de répondre avec les technologies qu'il maîtrise le mieux. Cependant, ne vous désintéressez pas de la technique. Ne pas maîtriser cet aspect est très dangereux.
	Vouloir trop de fonctionnalités est parfois risqué. Commencez petit, avec les fonctionnalités indispensables. Plus vous élargirez le périmètre plus le projet sera difficile à gérer. <i>[cf. #1 - La peur du « pas assez » ou le périmètre à la dérive]</i>
	Le lotissement de la solution doit être logique et compréhensible. Lotir ne veut pas dire construire des briques non fonctionnelles

Un projet évolue dans un environnement hostile

Choix du prestataire	<p>Ne consultez pas un seul prestataire (et surtout si c'est un de vos amis). Mais n'en consultez pas trop non plus. Un trop grand nombre de réponses ne vous permettrait pas de les traiter correctement.</p> <p>Si le prix proposé par le prestataire est significativement peu élevé, soit le périmètre du projet a été sous-estimé, soit le prestataire propose un prix hameçon.</p> <p><i>[cf. #2 - Le prix hameçon]</i></p>
	<p>Les plannings doivent être réalistes. Ils ne respectent pas forcément vos attentes. Dans la réponse du prestataire, vous devez trouver les éléments qui justifient comment tenir (ou pas) le planning.</p> <p>Il est nécessaire de finaliser le montage du projet avant le démarrage, c'est-à-dire éclaircir les points les plus compliqués trop souvent négligés (migration, intégration avec un système tiers etc.).</p>
	<p>Démarrer un projet sans contrat ou avec un contrat mal ficelé est très dangereux. Il risque de vous manquer des éléments fondamentaux tels que la propriété des développements par exemple.</p>
Conception fonctionnelle et technique	<p>Ne laissez pas démarrer un projet (sauf pour un périmètre très restreint) sans avoir validé et donc parfaitement compris les spécifications fonctionnelles.</p> <p>Les spécifications sont le gage de la bonne compréhension du projet, le développement sera la résultante de ces documents.</p>
Développements	<p>Des points d'avancement doivent être réalisés pour vous assurer de la bonne tenue du projet. Un retard peut représenter un risque technique quant à la compétence des architectes /développeurs de la solution.</p> <p><i>[cf. #3 – Mesurer plutôt que supposer]</i></p>

Un projet évolue dans un environnement hostile

Développements (suite)	L'information doit être transmise de façon régulière, les bonnes nouvelles comme les moins bonnes. Ces points doivent détailler les risques et la manière de les gérer.
	L'intégration doit permettre au prestataire de valider techniquement la solution
	Une période de développement trop longue sans implication des utilisateurs est problématique. Il est nécessaire de parer ce risque en prévoyant des recettes intermédiaires. <i>[cf. #4 – L'effet tunnel]</i>
Mise en place de la solution	Impliquez-vous dans la recette de votre projet. Réalisez des cahiers de tests, créez des jeux de données pour tester les éléments clés de votre projet <i>[cf. #5 – Le nombre d'anomalies]</i>
	Le dimensionnement et les environnements doivent permettre une utilisation optimale de l'application (gestion des serveurs, de l'infrastructure réseau, etc.) Il vaut mieux séparer les développements de la production (partie du travail qui consiste à mettre en place les serveurs, les environnements, gérer les sauvegardes etc.).

Un projet évolue dans un environnement hostile

#1 - La peur du « pas assez » ou le périmètre à la dérive

La peur du "pas assez" est un symptôme fréquent. Dès l'origine du projet, le client définit un périmètre très vaste ou exprime des besoins de plus en plus complexes au fur et à mesure du projet.

Ce cas est fréquent lorsque le client souhaite lancer une nouvelle activité. Proposer de nombreuses fonctionnalités donne l'impression rassurante d'offrir un service plus complet au client final. Dans la réalité, c'est souvent l'inverse. Il faut définir le périmètre le plus petit possible, tester rapidement et redéfinir le besoin en utilisant les retours utilisateurs ou bêta testeurs ("Pave the cow path").

Définir un périmètre très vaste donne aussi l'impression trompeuse de coûter moins cher qu'un développement en plusieurs étapes. Cela ne reste qu'un leurre. Au final, si de nombreuses fonctionnalités ne servent pas, le coût de ces développements inutiles et de l'effort nécessaire pour les produire peuvent mener le projet vers l'échec.

La dérive d'un périmètre peut aussi être attribuée à une mauvaise gestion de projet. Un prestataire en mauvaise position, comme par exemple dans le cas d'un retard important ou d'un problème de ressources, peut accepter des développements complémentaires en espérant conserver de bonnes relations avec son client. Il lui rend un mauvais service. En cumulant retard et nouveaux développements, le projet risque d'être encore plus long à finaliser.

Un projet évolue dans un environnement hostile

#2 - Le prix hameçon

Le prix proposé par le prestataire est volontairement bas afin de "hameçonner" le client. Un prestataire minimise volontairement le prix de la prestation afin de remporter le marché.

Au fil du projet, deux situations se présenteront:

- Les développements continuent car vous acceptez de nombreux avenants
- Le projet est bloqué car vous ne souhaitez pas (ou ne pouvez pas) continuer à payer.

Ce "faux prix" peut fortement dégrader la relation que vous entretenez avec votre prestataire à mesure que le projet avance.

Il est cependant nécessaire de faire la différence entre un acte délibéré (un prestataire qui hameçonne un client) et un périmètre qui évolue et grandit anormalement en cours de route (là, c'est un peu la faute du donneur d'ordre).

#3 - Mesurer plutôt que supposer

Il est important de mettre en œuvre, dès le démarrage, un dispositif de gouvernance permettant de mesurer l'avancement, gérer les risques, d'arbitrer les évolutions. Ces mesures : temps consommé, avancement etc. doivent permettre de fournir à l'ensemble de l'équipe un avis objectif sur l'état de votre projet. Ils permettent de mettre en œuvre des plan d'actions pour gérer certains risques ou re-planifier certaines parties du projet.

Un projet évolue dans un environnement hostile

4 - L'effet tunnel

L'effet tunnel consiste à développer pendant une longue période sans faire intervenir les utilisateurs.

La solution développée risque, in fine, de ne pas convenir aux besoins/souhaits des utilisateurs. La recette, longue et donc fastidieuse, risque alors d'être bâclée, les utilisateurs ne disposant pas forcément du temps nécessaire pour la faire aboutir.

#5 - Le nombre d'anomalies

Voilà une question qui revient souvent : est-ce qu'un nombre important d'anomalies durant la phase de recette indique qu'un projet est en train de se planter ?

Dans la plupart des cas, non ! Plus le nombre d'anomalies est important, plus vos utilisateurs testent la solution. C'est lorsque la solution n'est pas assez testée que l'on peut se planter. Donc, de manière contre-intuitive, cet indicateur est plutôt un facteur de qualité.

La première recette technique doit avoir permis de régler la plupart des anomalies de base. Si cela n'est pas le cas, un recadrage peut s'avérer nécessaire pour éviter l'épuisement des utilisateurs en cours de recette.

L'autre nuance que l'on peut apporter est de déterminer si les corrections sont efficaces. Des problèmes liés aux performances qui ne peuvent être corrigés indiquent peut-être des problèmes importants liés à la conception.

Les méthodes de sauvetage



Les méthodes de sauvetage

Le diagnostic est sans appel : votre projet est planté. Pas de panique, TheCodingMachine vous apporte des solutions !

Poser le bon diagnostic résout une grande partie du problème, car une fois que les problèmes sont identifiés, il est possible de mettre en place un plan d'actions efficace.

Pour garantir un résultat fiable et sérieux, solliciter une société extérieure reste, à notre avis, la meilleure option (si vous choisissez cette solution, n'hésitez pas à nous appeler). Car la toute première étape consiste à se débarrasser des problèmes périphériques qui accompagnent inévitablement la dérive d'un projet et parasitent notre vision. Faire appel à un tiers qui est indépendant et sans a priori permet de dépassionner le débat.

L'importance du contexte

Réussir le sauvetage d'un projet nécessite de plonger au cœur du problème afin de dresser un tableau objectif de l'état du projet. Quels sont les problèmes ? Peuvent-ils être résolus rapidement ? Par qui et de quelle manière ?

Les solutions dépendent étroitement de l'état du projet : le projet est-il encore en développement ou bien en production. Elles dépendent aussi de la nature des problèmes : est-ce un problème technique ? Un problème concernant le périmètre ? Ces solutions dépendent évidemment de la taille du projet. Plus un projet est petit, plus il est simple de le reprendre depuis le début.

Les méthodes de sauvetage

Méthode 1 : Ne pas avoir peur ... de supprimer le problème

Si les problèmes ont une origine technique, l'application développée a peu de chance d'être "sauvée". Corriger des erreurs graves de conception de l'architecture est souvent très coûteux. A budget équivalent, il vaut mieux reprendre les développements techniques "from scratch".

L'analyse de l'existant doit donc être méticuleuse pour éviter de jeter une application qui pourrait être sauvée, mais également pour ne pas vouloir sauver cette application à tout prix et faire durer l'échec.

Méthode 2 : Gérer les problèmes périphériques : quelles solutions pour gérer les problèmes humains ?

Il faut tout d'abord établir un constat d'échec. Cela semble évident aux chefs de projets dont le prestataire est démissionnaire. Pour celui qui aura un prestataire qui tente de le rassurer, lui promet d'aboutir et qui tente de garder une bonne relation, établir ce constat sera, en revanche, difficile.

Si l'application mérite d'être sauvée, il est nécessaire de s'attaquer aux problèmes humains. C'est souvent délicat car les sentiments qui agitent les parties sont rarement bienveillants. Il est toujours douloureux de se tromper ou d'avoir le sentiment de s'être fait avoir par un prestataire peu scrupuleux. Se lamenter n'est pourtant pas une solution.

Les méthodes de sauvetage

Impossible, malheureusement, d'indiquer ici une solution universelle. Nous vous recommandons bon sens et pragmatisme : êtes-vous victime de votre prestataire ou bien de vous-même ? Est-ce que changer le management du projet va vous permettre de porter un nouveau regard sur le projet, apporter une nouvelle motivation à l'équipe ? Est-il possible de créer un électrochoc, de déclencher une prise de conscience de la part des équipes ?

Dans ce domaine, la créativité n'a pas de limites. Voici une présentation des pistes les plus évidentes et les plus efficaces.

1. Le désengagement du prestataire (et son éventuel remplacement), permet souvent de voir le projet avec un œil neuf afin d'évacuer les problèmes récurrents. Ce désengagement est-il souhaitable, possible ? Quels coûts supplémentaires va-t-il engendrer ? Quels bénéfices va-t-il rapporter ? Comment envisager la phase de transition ?

2. Faire une pause, pour se remettre les idées en place et se réorganiser : Cette pause est-elle souhaitable, possible ? Combien de temps (trop risquerait de mettre fin au projet) ? Qui et comment planifier la suite du projet ?

Les méthodes de sauvetage

3. Changer les équipes peut redonner un second souffle au projet. Les équipes usées vont être remplacées par du sang neuf. Pour cela il est nécessaire que l'image du projet ne soit pas trop dégradée afin de ne pas décourager les nouveaux collaborateurs. Ce changement est-il souhaitable, possible ? Quelle organisation mettre en place, qui conserver ?

4. Relancer les développements par étape, morceler les problèmes, permet d'en voir le bout et ainsi de se motiver. Peut-être est-il possible de découper le projet en sous-ensembles ? Le coût de cette réorganisation est-il supportable ? Les équipes sont-elles prêtes à ce changement ?

Les méthodes de sauvetage

Méthode 3. Gérer les problèmes techniques

Si le projet peut-être sauvé, alors allons-y, n'attendons plus !
Ce qui n'empêche pas d'intégrer méthode et rigueur à notre opération de sauvetage.

Pour identifier les risques et les actions associées à la reprise de votre projet, plusieurs techniques sont envisageables. Nous vous en proposons une qui a le mérite d'être à la fois simple à partager et facile à appliquer :

Classement des risques et actions :

- Impact : le coût associé à la survenance du risque. Par exemple, la chute de la base de données principale a un coût supérieur à la chute d'un serveur de mail;
- Probabilité : la probabilité de survenance du problème;
- Effort de correction : le coût associé à la mise en place d'un correctif;
- Catégorie : la catégorie associée au risque (architecture, sécurité du code, ...).

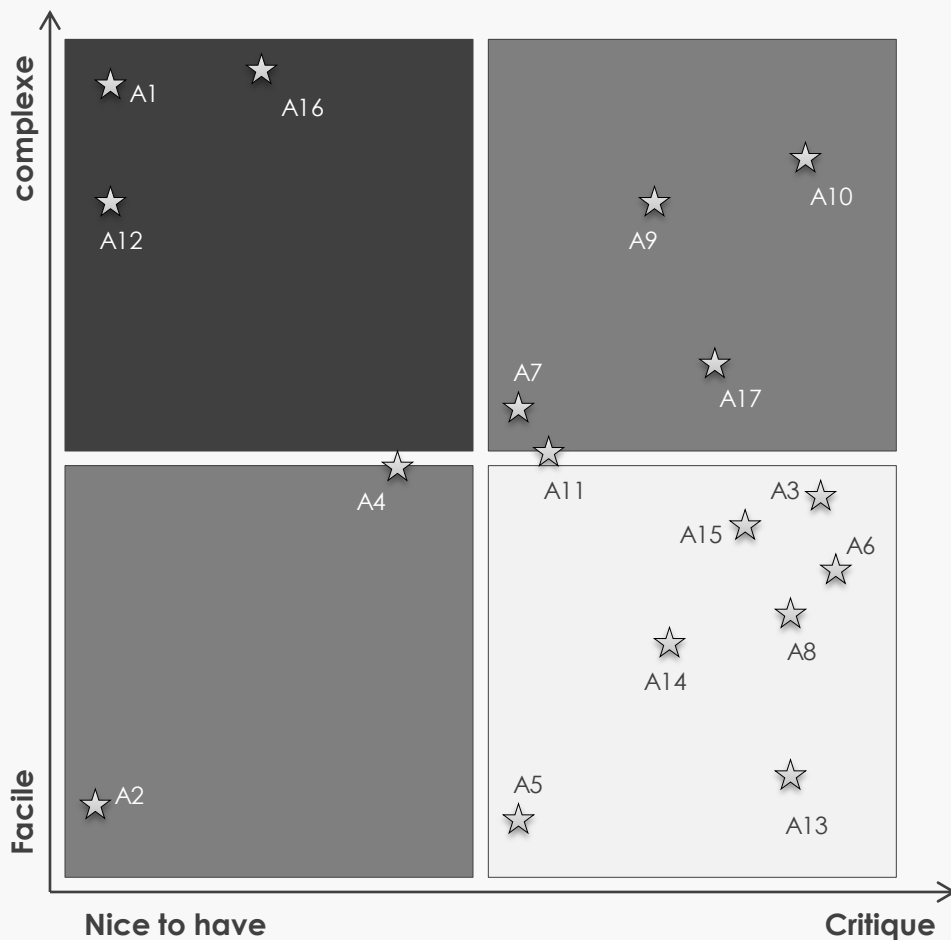
XXX: Description du risque et de l'action associée

		Catégorie
Impact:	Faible/Moyen/Fort : description de l'impact	
Probabilité:	Faible/Moyen/Fort : probabilité de survenance de l'évènement	
Effort de Correction:	Facile/Moyen/Complexe: description des tâches nécessaires pour corriger le problème	

Les méthodes de sauvetage

Les actions sont ensuite placées sur un quadrant :

- En abscisse: impact * probabilité;
- En ordonnée: complexité de mise en place du correctif.



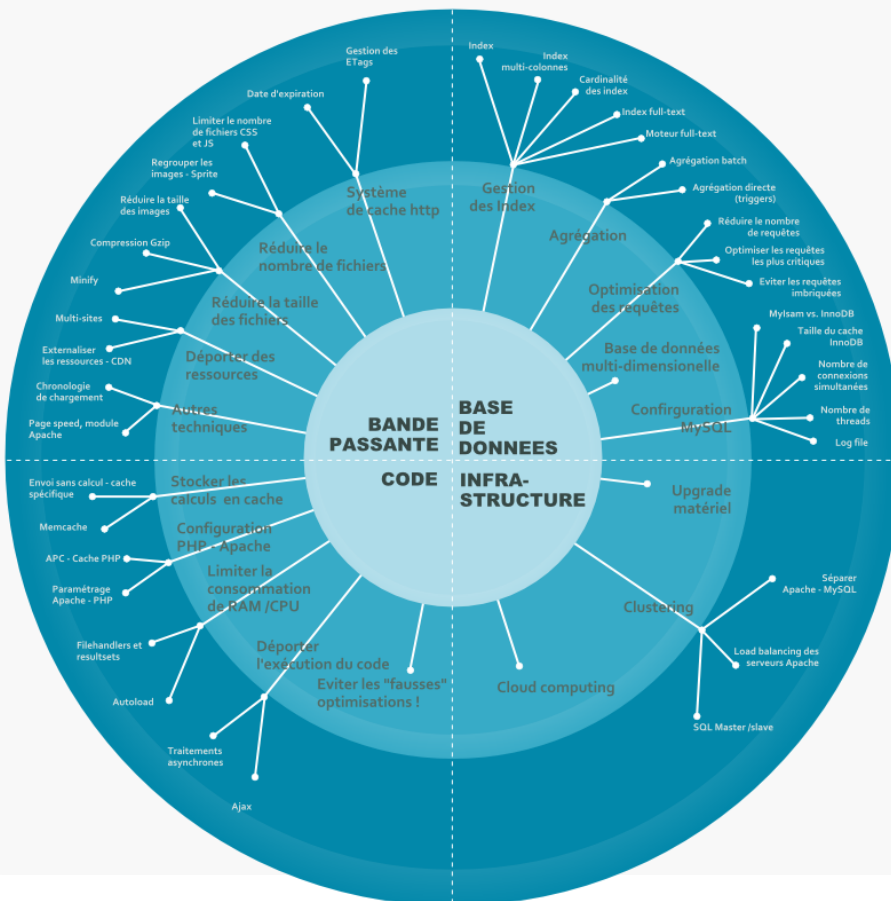
Les actions du quadrant en bas à droite seront les actions prioritaires. On remontera progressivement alors vers la section en haut à gauche.

Cas particulier : problématiques de performance

Les problèmes de performances passent souvent à la trappe dans les projets. Ils apparaissent souvent en fin de mission, s'ils n'ont pas été anticipés, et demeurent difficilement détectables avant la mise en production du projet.

TheCodingMachine a consacré un Livre Blanc à ce sujet. Nous y avons indiqué comment analyser et traiter ces problèmes particuliers. Vous pouvez télécharger le document à cette adresse :

<http://www.thecodingmachine.com/fr/hautes-performances-web>



Quelques cas traités par TheCodingMachine

Les projets et les situations évoqués ici étant purement fictifs,
toute ressemblance avec des projets existants ou ayant
existé ne saurait être que fortuite...

Cas n° 1 - Une mauvaise architecture

Le projet : Dans le cadre de la dématérialisation de ses correspondances avec ses clients, cette filiale d'un acteur majeur de la banque-assurance a demandé à un prestataire de créer un système de gestion électronique de documents (ou GED). Le projet comprenait l'externalisation de l'ouverture et de la numérisation du courrier, puis la distribution du courrier sous forme dématérialisée à travers une application qui gérait les workflows de demandes client.

Notre intervention : Devant les retards accumulés pendant le développement de ce projet et les dysfonctionnements constatés, TheCodingMachine a été mandatée pour conduire un audit, analyser finement l'application et dessiner le périmètre d'un plan de sauvetage.

Problèmes détectés : L'architecture de la solution avait été très mal pensée. Les données étaient distribuées dans deux bases de données distinctes. Le prestataire était parti avec un outil open-source et une technologie qu'il ne maîtrisait pas. Les temps de réponse étaient parfois difficile à supporter pour les utilisateurs et rendaient même parfois l'application inutilisable (avec des temps de réponse parfois supérieurs à 5mn). L'application était instable, avec parfois des arrêts de production de plusieurs jours.

Solutions apportées : Le processus de build et l'utilisation systématique de caches et sessions ont permis de consolider l'application sans remettre en cause les fondements de l'outil. L'industrialisation des développements, la mise en place de bonnes pratiques ainsi que le lancement de tests automatisés pour garantir les non régressions assurent la stabilité de l'application dans la durée.

Notre avis : Le coût du projet (500k€) rendait difficile un abandon pur et simple de la solution. Il aurait peut-être mieux valu. Notre intervention aura coûté 50k€ pour rendre l'application utilisable.

Cas n° 2 - Un choix technique inadapté

Le projet : Le client, dans le domaine de l'immobilier, souhaitait proposer à ses collaborateurs un Intranet : présentation des informations, gestion des formations, reporting etc.

Notre intervention : En rupture avec son ancien prestataire, le client devait montrer que son projet n'était pas en échec. Une partie de la conception a donc été reprise pour conserver le schéma relatif de base de données associé à la gestion des utilisateurs notamment.

Problèmes détectés : La technologie était parfaitement inadaptée aux besoins. Inexpérience du prestataire sur ce type de projet (spécialiste des sites vitrines).

Solutions apportées : Refonte du projet dans des délais record avec un budget réduit: 50k€ avait déjà été dépensé, il ne restait plus que 30k€ de budget pour faire le projet. Une rallonge de 15k€ a pu être obtenue. La structure du code a totalement été revue (utilisation d'un framework MVC, utilisation de bibliothèques open source, etc.).

Cas n° 3 - Abandon du projet par un prestataire

Le projet : Grande franchise liée à l'esthétique, le client souhaitait proposer la vente de ses services en ligne. C'est une agence marketing qui avait remporté le projet, qu'elle sous-traitait, pour la partie technique, à une SSI tunisienne. Avec les profonds changements apportés par la révolution, et compte tenu du manque de marge de manœuvre autorisé par le budget, le projet avait été abandonné par le prestataire.

Notre intervention : Pour respecter les délais, priorité très forte du client, nous avons conduit le sauvetage du projet en deux temps :

- nous avons épuré le périmètre pour mettre en pré-production une version testable par le client et lui permettre ainsi d'avoir une visibilité sur les avancées du projet,
- en parallèle, nous avons mené finalisé le développement du périmètre initial.

Problèmes détectés : Nous avons rapidement constaté que le projet avait été considérablement sous-estimé par l'agence.

Solutions apportées : L'architecture initiale (utilisation de Magento) a été conservée. En revanche, nous avons redéveloppé ou reconfiguré l'ensemble des fonctionnalités spécifiques (multi-boutique, thème, autres modules complémentaires).

Notre avis : Effectuer un sauvetage de projet pour le compte d'un tiers (et non du client final) n'est pas possible. La communication et le ressenti client est primordial dans ce genre de sauvetage. Il faut se positionner en frontal client pour le rassurer au mieux. Ne pas partir avec une agence qui dénigre la technique !

Cas n° 4 - Un périmètre à la dérive

Le projet : Cette société de vente par correspondance et de vente à domicile avait décidé de refondre son système d'informations (approvisionnement, vente, facturation, etc.). Le choix d'un prestataire off-shore et d'une organisation complexe (pas de communication directe, passage par un tiers, etc.) avait conduit le client à utiliser en production une application à moitié finie.

Notre intervention : Nous avons commencé par un audit technique de la « solution » existante (bien que peu de fonctionnalités aient été abouties). Puis nous avons décidé, conjointement avec le client, d'une nouvelle organisation.

Problèmes détectés : Deux problèmes majeurs ont été détectés :

1. Aucune spécification précise des fonctionnalités n'existait, les développements avaient tous été menés au fil de l'eau à partir d'un document général.
2. Quand les relations sont devenues tendues entre le client et le prestataire, les développements ont été accélérés au détriment de la qualité technique et des règles de codage (utilisation des outils installés).

Solutions apportées : Nous sommes repartis du point le plus avancé des développements respectant les règles et standards en essayant, au maximum, de récupérer et de réutiliser les fonctionnalités développées. Nous avons ensuite réorganisé les développements en écrivant les spécifications manquantes et en estimant/priorisant au préalable chaque tâche.

Notre avis : Il ne faut jamais abandonner les règles de base de la technique dans le seul but de gagner du temps. Cela s'avère souvent contre-productif pour la bonne conduite d'un projet. Le projet en serait, au mieux, trop compliqué à maintenir, et, dans le pire des scénarii, impossible à faire aboutir.

Conclusion

Ces méthodes ne s'appliquent pas les unes isolées des autres mais bien souvent de front. On peut estimer simplement si cela vaut la peine de sauver l'application en établissant le quadrant de gestion des problèmes techniques. Ce quadrant peut aussi être utile pour diagnostiquer rationnellement et donc accepter plus facilement que le projet est vraiment en panne.

Notre expérience nous a montré que la plupart des problèmes peuvent trouver une solution technique même si elle est parfois coûteuse. Le point le plus délicat à gérer se situe bien souvent dans les relations entre les parties prenantes. Des relations conflictuelles induites par des échecs accumulés sur un projet s'avèrent parfois difficiles à neutraliser. Il est pourtant important de dépassionner les débats pour permettre à des décisions réfléchies et rationnelles d'émerger, motivées par la réussite finale du projet.

Même s'il est compliqué de généraliser ce type de problématique, nous espérons que ce livre blanc vous aura au moins donné un peu de courage si votre projet est à l'arrêt, en panne ou même carrément raté.

Evidemment, de nombreux éléments mériteraient d'être approfondis. N'hésitez pas à nous contacter si vous souhaitez en discuter.

Un projet ? Des idées ?
N'hésitez pas à nous contacter !

TheCodingMachine™
TCM://

www.thecodingmachine.com

contact@thecodingmachine.com

01 71 18 39 72



The licensor permits others to copy, distribute, display, and perform the work. In return, licenses must give the original author credit.

The licensor permits others to copy, distribute, display, and perform the work. In return, licenses may not use the work for commercial purposes -- unless they get the licensor's permission.