Pathologie des Projets Système d'Information (SI)

Alain Chacun

Alain Chacun (www.linkedin.com/in/alainchacun) travaille comme *consultant ICT (Information Communication Technology*) pour des grandes banques d'investissement et banques privées Européennes depuis plus de 25 ans.

Ses expériences de *chef de projet* et de *test manager* l'ont conduit à mener certaines réflexions sur le cycle de vie des projets et leur industrialisation au sein des entreprises.

Gérard Balantzian : Dans le* «Chaos report 2013» (Standish Group), *se pourrait-il que les questions posées ou le type de projet choisi oriente ces résultats assez inquiétants quant au taux de projets réussis limités (inférieur à 50%) dès la première fois ? »

Alain Chacun : Je ne pense pas que les questions posées dans ce document orientent les résultats. Elles ne sont que la résultante d'un constat.

En revanche le type de projet choisi va influer sur les résultats. Des projets de complexité moyenne ou simple sont moins à risque de fait ne serait-ce que par leur durée de cycle de vie plus courte.

Si on prend l'exemple des projets web, ce sont des projets relativement simples ne demandant pas une armada de spécialistes ; prenons l'exemple de projet de type web banking, les requirements business sont assez simples, consulter son compte, faire des virements, envoyer des messages à sa banque, passer des ordres de bourse … et une banque X qui fait ce genre d'application à tout le loisir d'aller voir ce que la banque concurrente Y fait en la matière.

Par contre lorsque l'on rentre dans des projets complexes comme peuvent l'être les projets finance, là le risque est réel. Premièrement ils ont des cycles de vie longs, 1, 2 ans, même plus. Deuxièmement la complexité des requirements augmente le risque d'échec. Nous sommes dans des projets front to back qui demandent de prendre en compte les requirements couvrant un large scope, le trading, le delivery/settlement, le back office, la compta, le reporting légal (Bâle 3, FATCA, …), la gestion des avoirs (custody/asset management, corporate actions), etc…

Aussi l'architecture complexe de ce genre de projets augmente la difficulté. En effet nous sommes en présence de projets où l'on peut avoir couramment de 10 à 20 applications front to back qui

communiquent entre elles, d'où une complexité ajoutée quant à la connectivité avec des protocoles de communication divers et variés (SWIFT, FTP, MQ Series, flat file, XLS/CSV, etc…).

Le manque de vue globale dans ce genre de projet est un risque, d'où l'extrême nécessité d'avoir une bonne compréhension du *'AS IS'* (l'existant, ce que l'on a aujourd'hui) pour comprendre ce que l'on veut mettre en place, le '*TO BE'* (le devenir, ce que l'on souhaite avoir). Je n'ai vu que très rarement des banques avoir une approche adéquate dans la gestion de projet complexe.

GB : Se pourrait-il que le charisme d'un homme suffise pour changer cette réalité ?

AC : Je dirai qu'en 27 ans de métier, Je n'ai vu qu'une seule banque qui soit dans ce cas de figure, et ceci uniquement grâce au charisme d'un seul homme qui était à la tête des projets dans leur ensemble. Mais tout ne doit pas reposer que sur une ou quelque personnes car nous sommes alors typiquement dans le pire cas de figure décrit dans le niveau 1 du *CMMI (Capability Maturity Model Integration*) qui indique clairement que tout repose sur la compétence et le bon vouloir d'une ou de quelque personnes. Rien n'est normé, rien n'est structuré. Aucune procédure n'est mise en place pour s'assurer que tous les projets passent par des processus de spécifications et de contrôles similaires tout au long du *Software Development Life Cycle (SDLC* – cycle de vie des projets). Enlevez ces personnes clefs, et le projet est 'at risk'.

GB : Le secteur économique a-t-il une influence ?

AC : En effet, le secteur économique dans lequel on se trouve influe aussi. Les banques ont malheureusement trop souvent un manque d'appréhension du SDLC et négligent des parties en amont du projet qui sont pourtant cruciales quant au succès du projet, définir un business case, avoir de bons requirements, compréhension du 'AS IS' et du 'TO BE', donner aux équipes de test une dimension assurance qualité, ….

Des industries comme le nucléaire par exemple ne peuvent se permettre de tels manques. Pour avoir travaillé dans ce secteur, il est évident que l'on ne peut pas se permettre de perdre la trace d'un fut vitrifié contenant des déchets radioactifs pour 1000 ans.
Si on transpose enfin ces approches gestion de projet hors contexte 'SI' pur, sachant que les cycles de vie des projets et des méthodologies de test similaires à *ISTQB* (*International Software Testing Qualification Board*) s'appliquent aussi à des projets non SI, je dirai qu'en général la gestion de projet hors monde financier requiert beaucoup plus de rigueur sachant que d'une part les industries n'ont pas les ressources financières des banques, et que d'autre part les industries ne peuvent se permettre tant de risque sur leurs projets.

Pour ne prendre qu'un exemple, quand un constructeur automobile sort un nouveau modèle, sachant le montant des investissements requit pouvant aller jusqu'à 2 milliards d'euro, on ne peut que comprendre qu'il n'a pas le droit à l'erreur en terme de marketing, de cible client, qualité du modèle, et attrait du modèle.

GB : Le soutien continu de la direction générale est-il nécessaire ?

AC : Absolument, c'est même crucial. Si la DG ne soutient pas le projet ne serait-ce que d'un point de vue financier, le projet est mort d'avance. Aussi la réaffectation de budget d'un projet sur un autre a des effets dévastateurs sur le projet qui a son budget réduit. Cela conduit à une re-planification du projet du fait de manque de ressources, donc à une date de mise en production retardée. Au final, on peut même en venir à une démotivation des équipes projet, donc une perte d'intérêt pour celui-ci.

Mais l'aspect financier n'est pas tout car le rôle de la direction ne se limite pas à tenir les cordons de la bourse. Le rôle de la direction, au travers du sponsor, est de faire comprendre le pourquoi du changement. Les raisons du changement peuvent être diverses et variées, coûts de maintenance et de licences élevés, obsolescence de l'outil ne permettant pas d'évoluer fonctionnellement, faire face à la concurrence, besoin d'offrir de nouveaux services à la clientèle, etc…
Mais ces raisons ont besoin d'être expliquées aux utilisateurs finaux afin d'obtenir un consensus, ce qui est la base pour former une équipe projet soudée parlant d'une même voix ayant le même objectif.

GB : Comment garantir une meilleure implication des utilisateurs ?

AC : Comme dit ci-dessus, leur expliquer le pourquoi du changement. Si le sponsor, la DG échoue sur ce point, les utilisateurs adhéreront difficilement, voir jamais au projet et aux changements induits. Cela se verra surtout lorsque l'on arrivera à la phase de préparation et d'exécution de la recette utilisateur, ce que l'on appelle en Anglais les *User Acceptance Test (UAT*), car il y aura un rejet en masse de leur part.

Mais cela n'est pas tout. Il faut aussi les impliquer dans le processus de décision de la solution qui sera mise en place in fine. Si on impose une solution, elle sera difficilement accepter et encore une fois, on rencontrera des problèmes lors de la préparation et l'exécution de la recette utilisateur, et pour finir, l'obtention du sign-off.
C'est un exemple que je donne souvent ; c'est comme si on vous imposait comme voiture une Smart alors que vous avez 2 enfants et que vous les emmenez tous les jours à l'école ; la Smart ne répond pas à votre besoin.
Encore une fois il est nécessaire d'avoir une bonne compréhension de ce que l'on a, le 'AS IS', pour être capable de dire : 'ça je veux le garder', 'ceci ne répond pas ou plus à mon besoin', et 'ça je veux l'améliorer'. Ainsi cela vous donne une bonne idée de la solution que vous souhaitez obtenir, c'est-à-dire le 'TO BE'.
Ainsi vous pouvez benchmarker les différentes solutions du marché ou internes. Mais là encore il ne faut pas voir que l'aspect financier. Il faut prendre en compte le besoin utilisateur mais aussi la solidité et la réputation de l'éditeur si on se tourne vers une solution progiciel.

GB : Le « collaboratif » et la révolution numérique peuvent-ils améliorer la dynamique relationnelle en équipe et la capitalisation des connaissances ?

AC : En effet le collaboratif et la révolution numérique peuvent améliorer ces deux points.
On peut centraliser l'information pour la partager et capitaliser sur les expériences passées et en cours sur les projets ainsi que sur le travail accompli, les spécifications, les cas de test, etc… Les outils ne manquent pas, ne serait-ce que Sharepoint, le net, Wikipedia, etc…
Pour le collaboratif, des approches RAD, Agile, Scrum peuvent être appliquées sur des projets ou partie de projets permettant de dynamiser cet aspect collaboratif entre les utilisateurs, les développeurs et les testeurs. Mais cela demande une certaine rigueur dans la fréquence des réunions et un état d'esprit ; il faut savoir jouer le jeu car en fait c'est quelque part un jeu de questions/réponses.

Mais ne nous y trompons pas non plus. Ce n'est pas parce que l'on aura les meilleurs outils numériques que l'on saura partager l'information. Encore faut-il vouloir la partager ; on est alors dans le cas de la rétention d'information ; mais aussi savoir comment la partager et la structurer. Prenez l'exemple d'un outil comme Sharepoint, le nombre de fois que j'ai pu voir une structuration des arborescences mal pensée ; de ce fait on sait que l'information est là quelque part, mais on ne sait pas où exactement. C'est comme chercher une aiguille dans une meule de foin.

Aussi trop d'information tue l'information ; on peut manquer l'essentiel de l'information d'un projet parce que l'on n'a pas lu LE document qu'il fallait.
C'est pour cela qu'il est essentiel d'avoir un certains nombres de procédures et de règles qui régissent le cycle de vie des projets (SDLC). Ces règles et procédures sont là pour définir les documents, les délivrables, qui doivent être fournis tout au long du SDLC à certaines étapes. Ainsi on fournit les documents nécessaires au projet, et uniquement cela.
Ensuite il faut avoir des templates de ces documents permettant ainsi de savoir quel type d'information est requis dans chacun de ces documents.
Il faut aussi savoir être synthétique, savoir structurer sa pensée pour savoir la retranscrire sur papier, '*come to the point*' comme on dit en Anglais. Le nombre de fois où j'ai pu lire des analyses ou des stratégies de test extrêmement verbeuses où au final, on ne comprend pas le 'quoi', le 'comment', le 'qui' et le 'quand'.
Enfin il faut savoir comparer les documents entre eux. Je m'explique : un document d'analyse business (les besoins utilisateurs) doit être comparé à l'analyse fonctionnelle qui en découle sinon on coure le risque de dire blanc dans le premier, gris dans le second et on spécifiera noir dans l'architecture technique et fonctionnelle et dans le code qui en découlera. Au final on ne répondra pas au besoin de l'utilisateur. C'est là que le rôle de l'assurance qualité prend toute sa dimension, en vérifiant cette cohérence. Cela demande beaucoup de « collaboratif » entre les équipes.

GB : Le testing ne peut-il pas répondre à ce besoin d'assurance qualité?

AC : Absolument mais tout dépend du sens que l'on donne au testing. Si on pense au testing du code, on parle alors de 'dynamic testing' en ISTQB. Elle est dynamique car on teste le code contre des cas de test.
Mais le 'dynamic testing' n'est fait qu'une fois que le code est délivré. Il est alors trop tard car la revue des documents tout au long de la phase de spécification n'a pas été faite, donc tous les bugs inhérents à la phase de spécification n'ont pas été vus.
Afin de palier à ces erreurs lors de la phase de spécification, il faut revoir ces documents de spécification afin de s'assurer de leur consistance et de leur cohérence entre eux. C'est ce que l'on appelle le 'static testing' en ISTQB.

Cette phase de 'static testing', de revue des documents, est essentielle car plus sera tard la découverte d'un bug, plus sera élevé le coût pour résoudre le bug ; Prenons un exemple, le bug a été introduit lors des spécifications fonctionnelles au cours desquelles l'analyste a interprété la demande initiale de l'utilisateur. Il n'y a eu aucune revue des spécifications fonctionnelles pour s'assurer qu'elles étaient cohérentes avec les spécifications business, pas de 'static testing'. De ce fait les spécifications de l'architecture fonctionnelle, de l'architecture technique et le code ont répliqué cette interprétation. Le bug ne sera découvert que lors des tests utilisateur (UAT), qui est l'une des dernières phases de test avant la mise en production. Quel est le coût ? Il faut revoir les analyses fonctionnelles, l'architecture fonctionnelle, technique et le code. Un bug trouvé tardivement peut coûter jusqu'à 100 fois le coût initial du besoin. C'est exponentiel !

Donc ce qu'il faut comprendre, c'est qu'il faut savoir donner une dimension assurance qualité aux projets en donnant aux équipes de test ce rôle de 'static testing' en amont des projets pour s'assurer de cette cohérence tout au long de la phase de spécification afin de s'assurer que le code que l'on va développer répond au besoin initial de l'utilisateur.
Le testing est trop souvent vu comme un coût. C'est une grave erreur. Il faut le voir comme un retour sur investissement (ROI) car on va considérablement réduire le risque de bug tout au long du projet, sachant qu'un bug peut coûter 100 fois plus que le besoin initial.

Ainsi, le 'dynamic testing' (le test du code) est indissociable du 'static testing' (la revue des documents de spécification) car vous pourrez avoir les plus beaux cas de test du monde pour votre projet, ils ne sont rien si cette revue des spécifications n'a pas été faite.

GB : La gestion des priorités collectives est-elle bien assurée afin de mieux se concentrer sur l'essentiel ?

AC : Je crois que cela dépend totalement de l'organisation des entités.
Si on est dans le cas d'une entité où chaque département utilisateur défini chacun de son côté ses besoins en terme de SI, il n'y a alors pas de synergie créée quant aux besoins des utilisateurs. Ainsi on va se retrouver par exemple avec deux projets pour deux départements distincts alors que ces deux projets ont une part non négligeable de fonctionnalités similaires. On a donc une double dépense pour des besoins identiques.
Si on est dans le cas d'une entité où il y a un comité de validation des projets, on est alors censé éviter ce genre de double dépense.

Si on revient au monde 'SI' de la finance, on est confronté à des priorités que l'on doit gérer tout au long de l'année.
Par exemple vous avez des releases SWIFT qui vous imposent d'être prêt à telle date si vous voulez pouvoir continuer à trader sur les marchés financiers.
Vous avez aussi des priorités de type régulateur comme Bâle III, FATCA, MiFID, etc… qui imposent un certain nombre de réglementation aux banques pour lesquelles elles doivent être prêtes à des périodes bien précises sous peine de pénalité de la part des régulateurs.

Tout cela est planifiable d'avance car à aucun moment un régulateur ou SWIFT imposera du jour au lendemain une modification. Il y a toujours un calendrier.

Ensuite on a les priorités internes parce que l'on doit tenir la concurrence éloignée, fournir de nouveaux services au client, changer de software parce que trop couteux et/ou obsolète, etc…
C'est là que le comité de validation prend toute sa raison d'être car chaque département va prêcher pour sa chapelle pour obtenir le budget au détriment d'un autre projet d'un autre département. Le comité de validation doit alors jouer un rôle d'arbitre quant aux priorités, s'assurer que le besoin est bien nouveau et non pas déjà disponible dans un autre département, valider que le besoin est bien nécessaire et qu'il y a un retour sur investissement (ROI), quel est le business cases ?, le planifier selon le calendrier des nouvelles dispositions réglementaires et SWIFT et enfin selon le budget disponible.

Mais parfois, et malgré le fait qu'il y ait un comité qui valide les projets, vous vous apercevez que les priorités dîtes collectives changent continuellement induisant un changement de cap continuel impactant de ce fait tous les projets en cours.
On se demande alors s'il y a un pilote dans l'avion !

GB : A quoi est-ce dû ?

AC : Je dirai que la cause principale est un manque de vision à moyen et long terme. On a trop tendance aujourd'hui à gérer les entreprises sur le court terme induisant de ce fait un manque de gouvernance et de tenue de cap. D'autres facteurs peuvent aussi venir influencer, comme le copinage d'un manager d'un département avec un des membres influents du comité de validation et qui va réussir à le convaincre de lui allouer le budget initialement accordé à un autre département et aussi la loi du plus fort, un manager d'un département qui va crier plus fort que les autres, tout simplement.

Mais je pense aussi qu'il y a aussi une dimension collaborative qui est à prendre aussi en considération. C'est la relation entre les utilisateurs et les services 'SI'. Je pense que la décision de faire un projet ou non est une décision collégiale notamment entre ces deux acteurs. On s'aperçoit assez souvent qu'il y a deux cas de figure : le premier cas de figure c'est le département 'SI' qui est dans sa tour de Babel et qui est seul à prendre la décision de faire ou non un projet selon son bon vouloir. Ca ne peut pas fonctionner car les utilisateurs ont besoin du département 'SI' pour développer des outils. Le second cas de figure c'est quand les utilisateurs considèrent que le département 'SI' est à leur botte et qu'ils ne sont là que pour obéir à leurs ordres et à leurs demandes. Ca ne peut pas fonctionner non plus car le département 'SI' a aussi des contraintes inhérentes à la technique ou autres.
La dimension collaborative est extrêmement importante entre ces deux entités et ils doivent avoir leur mot à dire de façon équitable en ce qui concerne la faisabilité et le planning des projets.
Malheureusement trop souvent les départements 'SI' sont financés par les départements utilisateurs ce qui engendre cette relation hiérarchique.

GB : En matière de tableaux de bord : tout est-il mesurable ?

AC : Oui et non.
Oui parce que l'on peut mesurer les différentes phases d'un projet (spécification, QA, développement du code, testing et mise en production) en mesurant l'avancement et contrôlant les dépenses budgétaires. On a les *KPI* (*Key Progress/Performance Indicator*) pour mesurer tout cela.
Non parce que le facteur humain n'est pas mesurable, en tous les cas dans l'immédiat. Que ce soit un manque de motivation, des difficultés à réaliser le travail demandé, la volonté avérée de saboter un projet, n'importe quelle raison que ce soit… ça ne se verra qu'après coup sur les KPI.

Ceci étant dit, les chiffres du tableau de bord reflètent la réalité du projet.
On peut donc mesurer le succès, le retard ou l'échec d'un projet. Dans le premier cas on a tenu le budget et le planning à +/- 10% et le client est satisfait, on a le sign-off.
Dans le deuxième cas on a certes le sign-off de l'utilisateur mais le budget et le planning ont été multiplié par 2, 3, 4….
Dans le troisième cas, on n'a pas le sign-off des utilisateurs, on a dépassé le budget et le planning et on peut presque tout mettre au rebus. C'est une perte sèche.

Mais une fois que ce constat-là est fait, il faut savoir refaire l'historique du projet qui vient de se terminer et en tirer des leçons (*lessons learnt* en Anglais), que le projet se soit bien passé ou non. Le but est d'identifier les raisons pour lesquelles le projet s'est bien ou mal passé afin de pouvoir les comprendre et en tirer bénéfice pour les projets à venir. Il faut aussi savoir être critique sur soi-même.

GB : Quelle conclusion peut-on tirer de tout cela ?

AC : On peut dire qu'il y a encore beaucoup trop de projets qui sont soit annulés, soit en retard avec un dépassement de budget. Il y a des méthodes et des outils pour pallier à ça, encore faut-il savoir comment les utiliser. Il existe des personnes compétentes pour aider sur ce point. Mais ce qu'il faut d'abord c'est la volonté de vouloir changer les choses pour acquérir une certaine maturité dans l'industrialisation des projets SI.
Il y a une prise de conscience qui s'opère actuellement au sein des entreprises en général. Je dirai que les entreprises hors finance ont un peu plus d'avance sur le sujet. Les banques en parlent ou ont initié des choses sur le sujet mais cela reste encore trop timide.
Il faut que les banques aillent de l'avant sur ce sujet. Elles ont des moyens que les entreprises des autres secteurs n'ont pas. Les banques ont en générale des personnes de qualité dans les

départements 'SI'. Elles ont tous les atouts pour réussir de fabuleux projets et je reste persuadé qu'elles peuvent les réaliser avec des budgets plus rationalisés.
Pour finir, les entreprises en général doivent comprendre que les méthodologies projet ne sont pas un carcan dans lequel elles doivent s'insérer. Au contraire les méthodologies sont là pour s'adapter aux spécificités et à la maturité des entreprises. Ces mêmes méthodologies sont là pour grandir avec l'entreprise au fur et à mesure que celle-ci acquiert en maturité. On rentre typiquement dans une configuration CMMI. Rien ne sert de vouloir brûler les étapes car ce serait un échec. Un enfant n'apprend pas à marcher du jour au lendemain. On est exactement dans le même cas. CMMI est là pour répondre à cette (r)évolution progressive. C'est un accompagnement, du coaching auquel les entreprises, les DSI (Directeur des Systèmes d'Information) peuvent recourir.

Interview menée par Gérard Balantzian
Paris, le 17 mai 2014