C. Absil

Retour d'expérience sur la gestion agile du projet Pylos : bilan à mi parcours

Agile France
2010

A.Fayolle

Acte I
La Vente

Contexte

Développement d'un *framework* pour la configuration et le pilotage de codes d'optimisation utilisés par GdFSuez
Refonte d'un framework existant, développé en mode classique par le département IT interne
Projet construit sur le framework CubicWeb de Logilab
Projet mené en mode agile avec une équipe distribuée (Paris (2) / Louvain la Neuve (3) / Annecy (1))
Début des développements le 1er Octobre 2009
Première version complète évaluée à 2 WP de 125 jh
Intégration dans l'infrastructure du département IT interne

Logilab Alexandre Fayolle
Christine Absil
GDF SUEZ

Mode itératif

Livrer des versions successives et utilisables qui convergent vers la version finale
Ne pas perdre d'énergie à maintenir des spécifications détaillées qui ne sont pas nécessaire
Les nouvelles orientations fonctionnelles sont possibles même tard dans le projet
Les spécifications détaillées sont écrites « juste à temps »

Planification agile

Chaque livraison est un petit projet qui est planifié en tant que tel
L'équipe utilise l'expérience acquise sur les livraisons précédentes pour affiner les estimations
On utilise le temps disponible pour fixer le périmètre de la livraison (planification à temps fixe) ou on estime la date de livraison à partir du périmètre fonctionnel

Qualité

Chaque livraison doit être faite avec à l'esprit les livraisons suivantes
Il est hors de question de saborder la suite du projet en dégradant la qualité d'une livraison
Préservation de l'écologie du projet au quotidien (code, tests...)
Les compromis (hors baisse de la qualité) possibles sont détaillés dans la charte du projet

Confiance, feedback

Livraisons régulières
Progrès visibles de tous (pas d'effet tunnel)
Pilotage du projet par le choix du contenu des livraisons
Investissement du Product Owner
Possible d'arrêter le projet tôt si ça ne marche pas
Chercher la collaboration plutôt que la confrontation

Qu'est-ce que l'agilité ?

Prendre acte que le *Waterfall* ne répond pas aux besoin d'un monde mouvant
Reconnaître que le projet n'est pas joué d'avance, et donc cultiver la souplesse
Révolution douce
Sortir de la confrontation, jouer le « nous collectif »
Mettre tout le monde sur le même pont, et amener tout le monde à bon port

Entracte
(Développement)

Outils agiles

- Planification par itérations de 4 semaines
- Entrepôt de source partagé (mercurial)
- Intégration continue
- Tests automatisés
- *Pair programming* sur points cruciaux
- Sprints (au sens de la communauté Logiciel Libre, plus qu'au sens de Scrum)

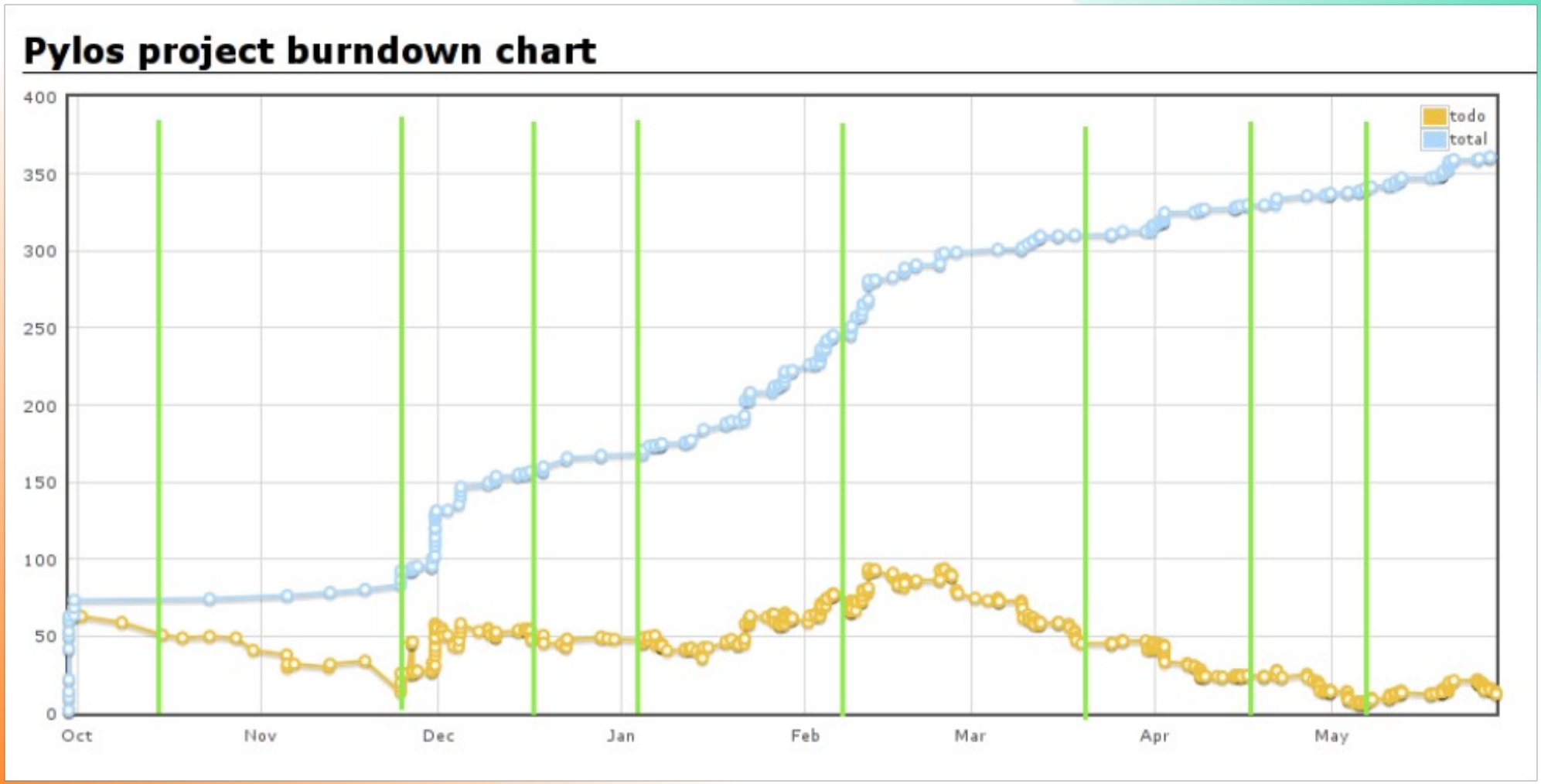
- Extranet
  - Histoires utilisateurs
  - Test cases
  - Gestion du *backlog* et des tickets
  - Suivi de l'avancement
  - Documentation
- Communication
  - Messagerie instantanée
  - Téléphone
  - Enfilades de commentaires sur l'extranet
  - Courrier électronique

Acte II
8 mois plus tard

Mini Glossaire

Pylos : le *framework* de configuration et de pilotage des codes d'optimisation
NERD : la version précédente du *framework*, développée en mode « traditionnel »
Pégase : le premier code d'optimisation porté dans Pylos, et servant de guide pour le développement du *framework*

Progression des développements



Logilab Alexandre Fayolle

Christine Absil

GDF SUEZ

Le *product owner* idéal

Connaissance du métier à informatiser
Fibre projet
Disponible à environ 100% et parfois un peu plus

Évolutions, tests

Automatiser les tests
Viser l'exhaustivité des tests
Tester une cible mouvante
Migrer les tests d'une release à l'autre
Gestion des version, utilisation du gestionnaire de sources, des branche
Combien de testeurs ?

Le monde n'est pas toujours agile

Interfaçage parfois délicats avec des intervenants qui n'ont pas la culture agile
Faire passer le message que le blâme ne résoud pas le problème
Intégrer, inviter, partager

Rideau !

C'est le moment de poser des questions

Questions importantes

Quelle durée d'itération ?
Comment découper en itération ?
Que faire quand le product owner se retrouve sur le chemin critique ?
Jusqu'à quelle taille de projet peut-on être agile ?

Bilan

L'agilité ne garantie pas la réussite du projet, mais la simplicité des méthode rend les problèmes visibles plus vite

Les responsabilités de chacun demeurent, mais le contexte est à l'ouverture et à la transparence

- Le MOA doit vraiment savoir ce qu'il veut
- Le MOE ne peut pas jouer de l'effet tunnel

La planification est faite en mode « juste à temps » et « juste assez » → gain de temps

On ne s'échine plus à blâmer, au contraire on cherche à gagner ensemble

C. Absil

GDF SUEZ

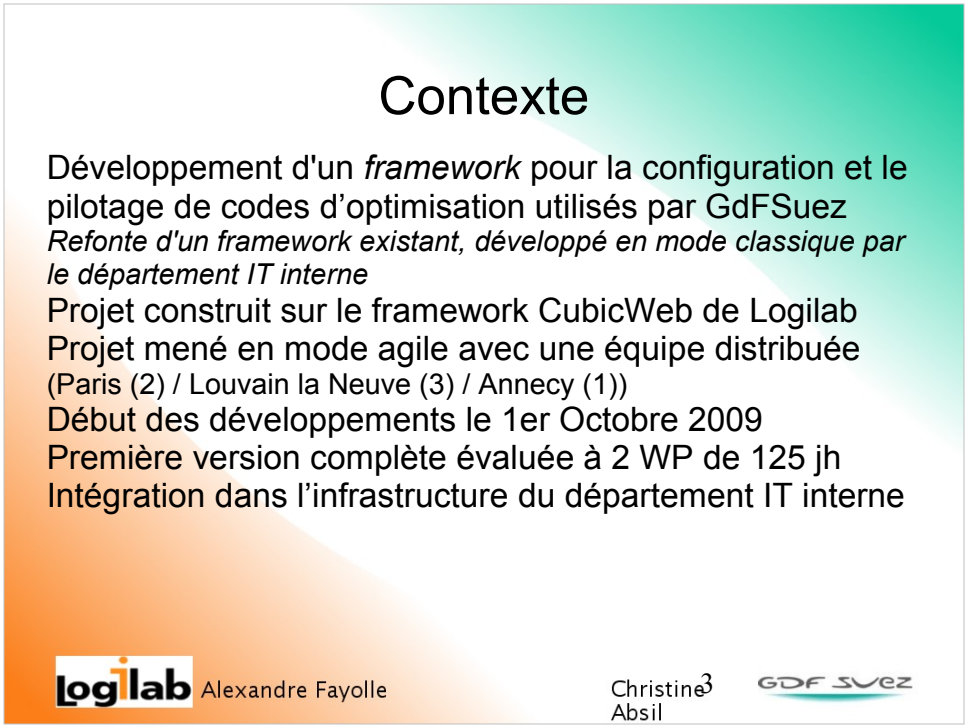
Retour d'expérience sur la gestion agile du projet Pylos : bilan à mi parcours

Agile France
2010

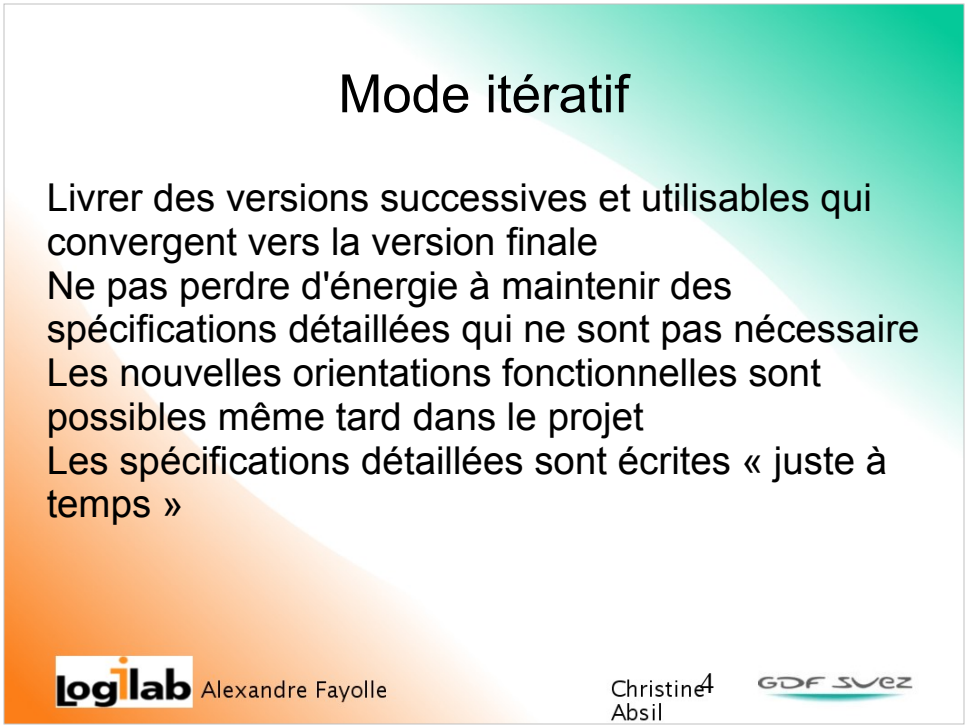
A.Fayolle

Logilab

Acte I
La Vente



CA: Bonjour, Je suis Modeller chez GDF-Suez, et je voudrais remplacer mon framework IT sous-tendant mes modèles mathématiques d'optimisation des réseaux énergétiques.
Voici mon Appel d'Offre, qui contient déjà la liste exhaustives de mes requirements *(déposant une pile de documents sur la table)*. Tandis que vous me préparez votre offre, je vais commencer à les détailler...



AF: Bigre, c'est un gros document. Je vais le lire bien entendu, mais... pour tout vous dire, sur des projets de cette taille là, j'aime travailler avec une démarche agile. L'idée générale, c'est que, comme très souvent dans une liste de besoins comme celle ci, il y a en fait une part significative dont on n'a pas besoin. L'acheteur les liste tout de meme parce qu'on lui a dit que quand on veut rajouter des choses après coup, ça lui coute plus cher. Beaucoup plus cher. Dans un projet agile, le fournisseur ne s'engage plus au forfait sur l'ensemble du projet, mais sur des unités de travail beaucoup plus petites, dont le contenu est défini au fur et à mesure du déroulement du projet. Ceci permet d'arreter les développements quand on a atteint un état satisfaisant, ou de modifier les orientations en cours de route pour répondre à des changements de besoin.

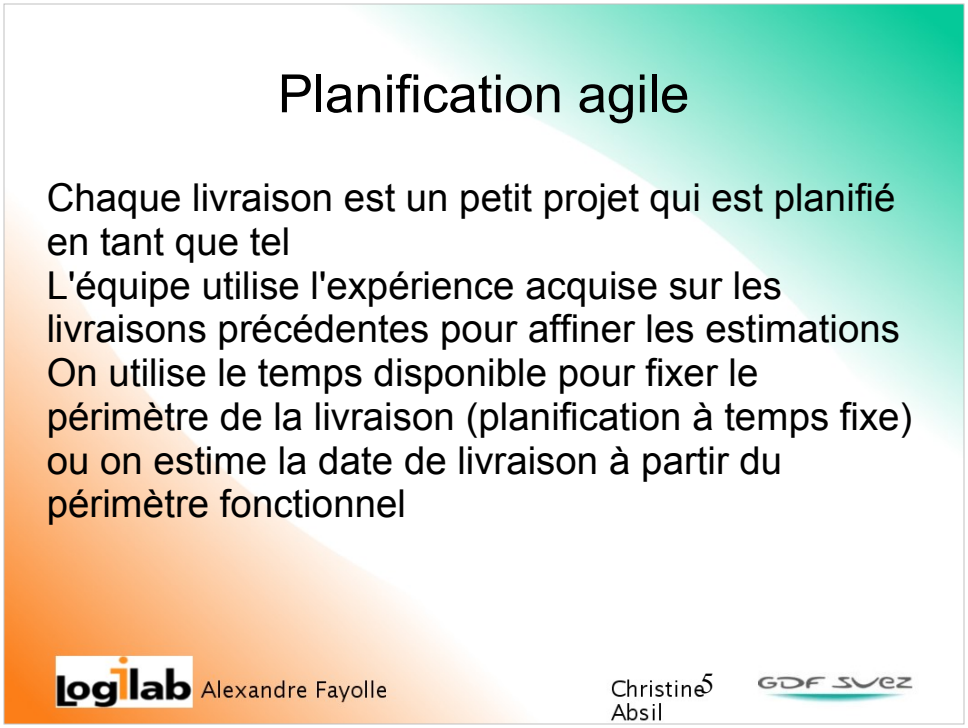
CA: Ah bon?! Comment c'est interessant ... (perplexe) ... Mais alors, que fais-je de mes requirements?

AF: Il ne faut pas les jeter, parce que c'est une bonne chose d'avoir une idée de ce qu'on voudrait voir dans l'application. Mais ce n'est pas la peine de vous lancer dans la tache énorme qui consiste à tous les détailler. Dans un projet agile, le Product Owner fournit les specifications détaillées au moment où elles sont nécessaires aux développeurs parce que la prochaine livraison concerne ces fonctionnalité. De cette façon, on n'a pas à gérer un stock périssable de spécifications.

CA: Ca m'a l'air très flou tout cela. Z'êtes surs qu'il y a bien une méthodologie épouvé là-derrière?! Je vous sens plus anarchiste qu'agile... Concrètement, donc, qui va écrire quelles specs quand?

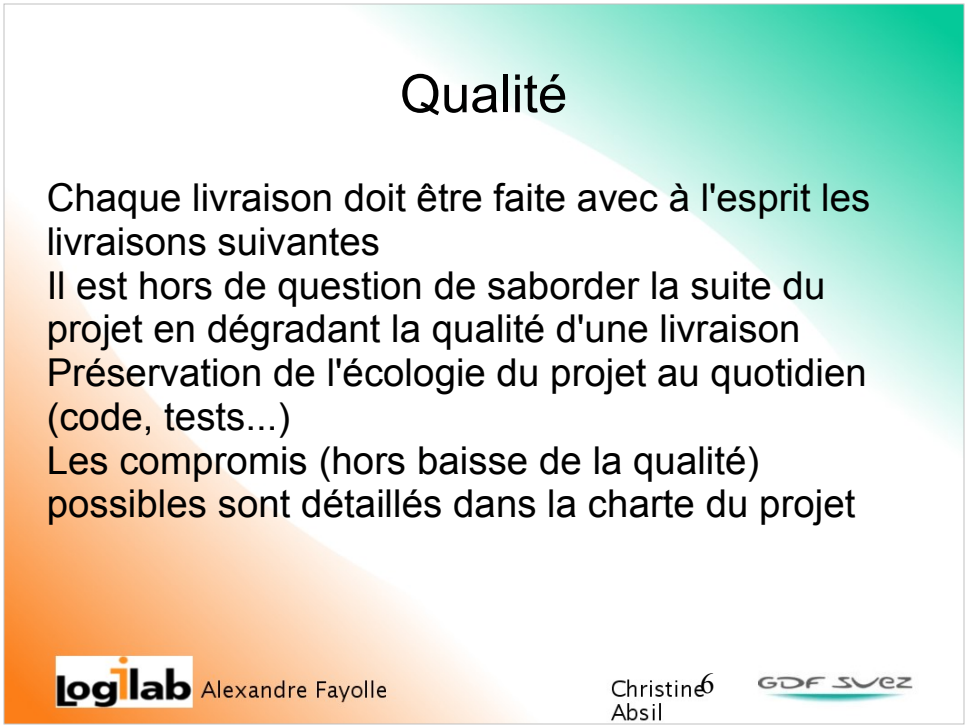
AF Le logiciel est développé par incréments successifs, en partant d'un coeur de fonctionnalités qu'on étoffe petit à petit en faisant des livraisons régulières dont vous choisissez le contenu. Pour chaque itération il faut etre en mesure de fournir les spécifications. Ca peut etre par écrit ou par oral, selon les cas. Le nombre d'itération que nous ferons sera pour partie fonction de la complexité du logiciel et de la taille de l'équipe de développement et pour partie fonction de votre budget.

CA: En parlant de budget: Tout ca me parait un peu nébuleux. Ca va me couter combien, cette plaisanterie? Et allez-vous tenir mon planning?



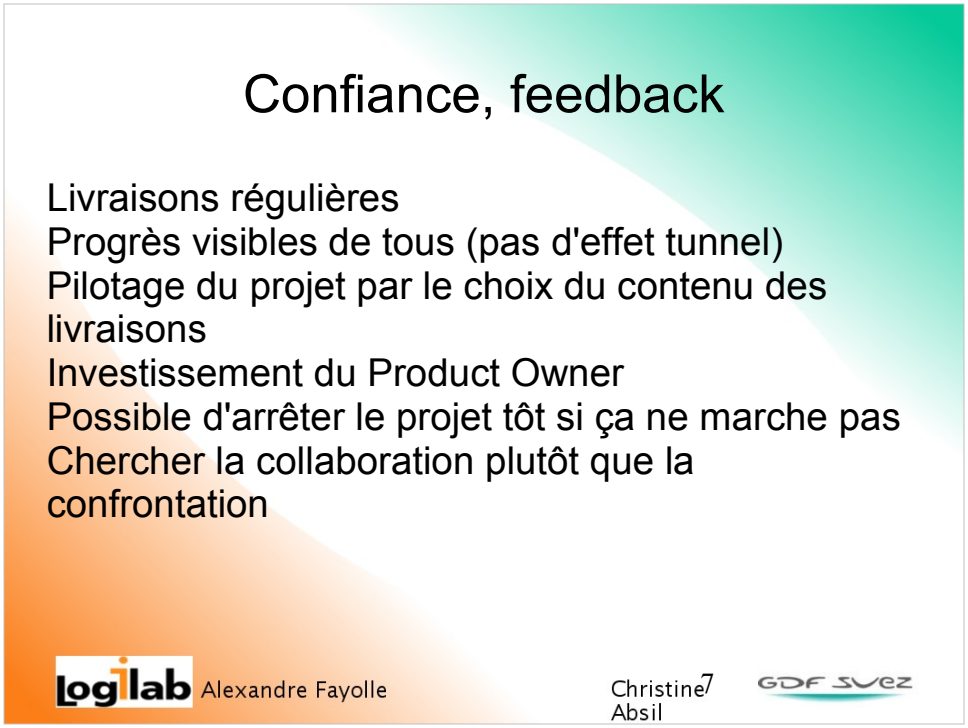
AF: On peut s'entendre sur un prix de journée. Et le cout sera directement proportionnel au temps passé. Après, vous déciderez si vous souhaitez travailler à budget fixé ou à temps fixé (c'est presque pareil), ou à périmètre fixé. Pour le planning, il sera revu à chaque livraison pour tenir compte des orientations que vous allez donner au projet.

CA: Mouais (sceptique). Je sens que cela va partir un peu dans tous les sens, cette affaire. Tout ça pourrait bien tourner en rond, non?



AF: Non, parce que nous allons ensemble piloter le projet, pour l'amener à bon port. L'agilité ce n'est pas l'anarchie, au contraire, il faut beaucoup de rigueur, pour assurer la livraison non pas d'un gros projet dans très longtemps mais de plein de plus petits projets de taille plus réduite mais qui doivent chacun etre livré en temps et en heure, et de plus ne pas compromettre la livraison suivante. Pour celà, au quotidien, l'équipe de développement doit s'attacher à garder le code dans un état de propreté impeccable. Je vous aiderai à garder le cap et à conserver la vision du projet dans son ensemble que nous consignerons dans un charte du projet. Les décisions importantes seront documentés pour que nous puissions nous souvenir des choix pris et des renoncements faits.

CA: Moui. Bon. Peut-être... Mais, qu'est ce qui me dit qu'au final, quand mon budget et/ou ma deadline sera(/ont) atteint(e) j'aurai bien mon framework "up-and-running"??



AF: Vous aurez des livraisons régulières de morceaux de logiciels fonctionnels que vous pourrez tester. C'est le contraire de l'effet tunnel trop souvent rencontré sur les gros projets. Ca vous permettre de prendre confiance dans notre travail et de rassurer votre hiérarchie. Et bien entendu de mieux piloter le projet en chamboulant les priorités en cours de route si nécessaire.

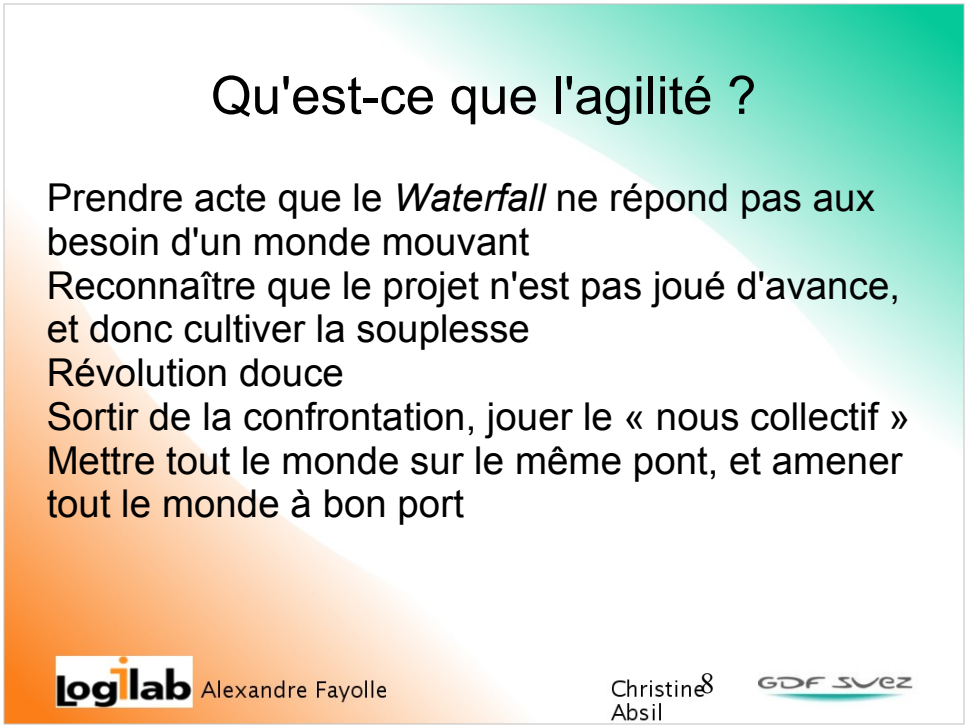
CA: Et moi, dans votre bazar, pratiquement, je ferai quoi? Et ça me prendre quelle part de mon temps?

AF: Vous serez le Product Owner. Cela signifie que, connaissant le métier à informatiser, vous décrirez les fonctionnalités attendues, indiquerez les priorités, trancherez les points douteux. Ne nous leurrons pas. Ce rôle sera très prenant. Idéalement, nous attendons de vous une disponibilité proche de 100%. Si en plus vous avez la fibre projet, ensemble, nous ferons des miracles!

CA: Et concrètement, vous, vous vous engagez à quoi? Si tout part en vrille, que rien ne fonctionne, et qu'on au final, on a construit un gros sac de noeuds, comment expliquerai-je cela à mon Chef? Sur quelles bases pourra-t-il vous coller un procès aux fesses?

AF: Nous nous engageons à vous livrer régulièrement du logiciel fonctionnel que vous pourrez tester. Nous nous engageons à vous montrer l'état du projet en permanence tel que nous le voyons dans la plus complète transparence. Si au bout de 1 ou 2 livraisons vous n'êtes pas contents de ce que vous voyez, vous pourrez arrêter les frais sans préavis, en n'ayant consommé qu'une faible part du votre budget, mais je ferai tout mon possible pour que ça n'arrive pas. Nous allons nous embarquer ensemble sur la même navire pour ce projet, travailler ensemble pour réussir, au lieu de nous tirer dans les pattes.

CA: Vous voulez dire qu'avec vos méthodes de hippies, il n'y aura personne à blamer en cas d'échec?! Si je comprends bien votre proposition, c'est un peu "Succeed together, or fail together"?! Bigre...



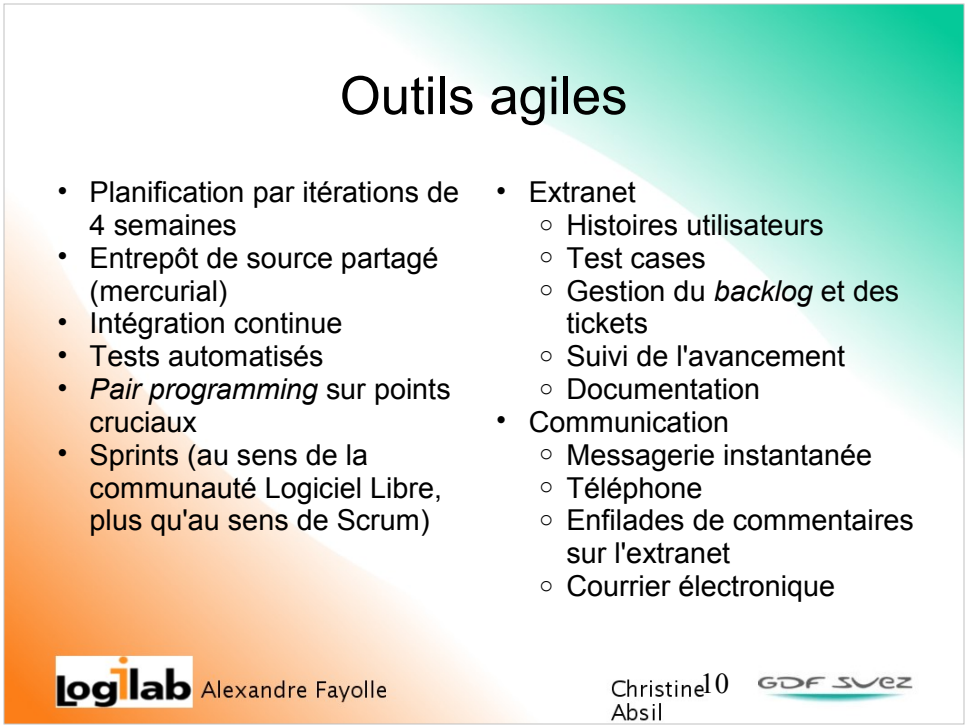
AF: Exactement. Dans un projet rien n'est jamais joué d'avance, il y a des impondérables. Chercher à tout prévoir, tout anticiper a un coût qu'il est difficile de rentabiliser. Il est plus intéressant de chercher à s'adapter aux circonstances, que de vouloir contrôler les aléas auxquels nous allons être soumis. C'est aussi ça, l'agilité. A contrario, c'est une des grandes causes d'échec des projets menés selon le mode "Waterfall".

CA: (songeur). C'est vrai que moi-même, je suis mal à l'aise lorsque mon chef me demande des plannings et des estimations détaillés sur des specifications vagues... (se ressaisissant) Cela me parait sensé, cette vision agile... Après tout, pourquoi ne pas essayer?

AF: ... Qui sait, peut-être même serez-vous tellement satisfaits que vous en demanderez d'avantage !

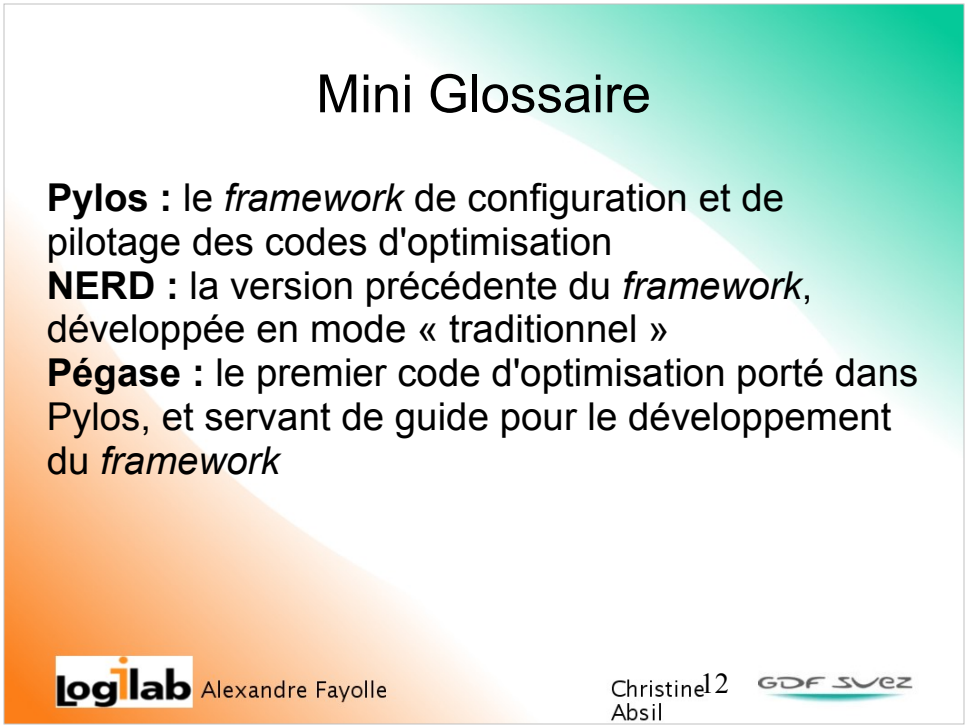
CA: Voui, bon, ne nous emballons pas non plus, hein! Nous verrons bien... Bon. Où donc dois-je signer?

Entracte
(Développement)

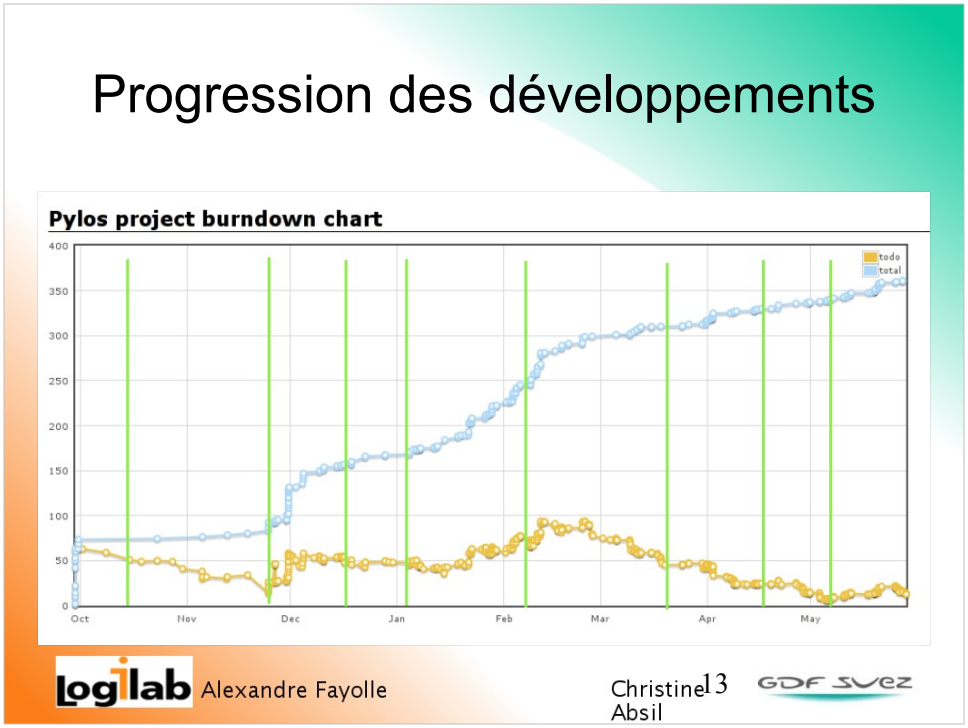


Cette partie de la pièce n'est pas au programme, mais nous pourrons en discuter après la conférence ou pendant les questions.

Acte II
8 mois plus tard



AF: on vient de livrer la version 0.9, la mise en production de Pegase est pour le mois de juin, nous sommes tous les deux à Agile France 2010, si on en profitait pour faire un petit bilan du projet et le mode de fonctionnement agile impromptu ?



CA: Excellente idée !

AF: Tu commences ?

CA: Si tu veux. D'un point de vu global, je suis satisfaite de la façon dont ça se passe. L'application est assez complète. Elle tourne, et semble tenir la route.

Meme si, retrospectivement, y a eu des moments délicats.

AF: Comme ?

CA: Et bien par exemple, au début, ma hiérarchie m'avait demandé de travailler à 50% sur un autre projet qui se terminait, ce qui m'a laissé trop peu de temps pour répondre à l'avalanche de questions en provenance de l'équipe de développement.

AF: Je me souviens de ça. Heureusement, ça n'a pas duré très longtemps et tu as pu au bout de 2 ou 3 semaines faire du Pylos à plein temps.

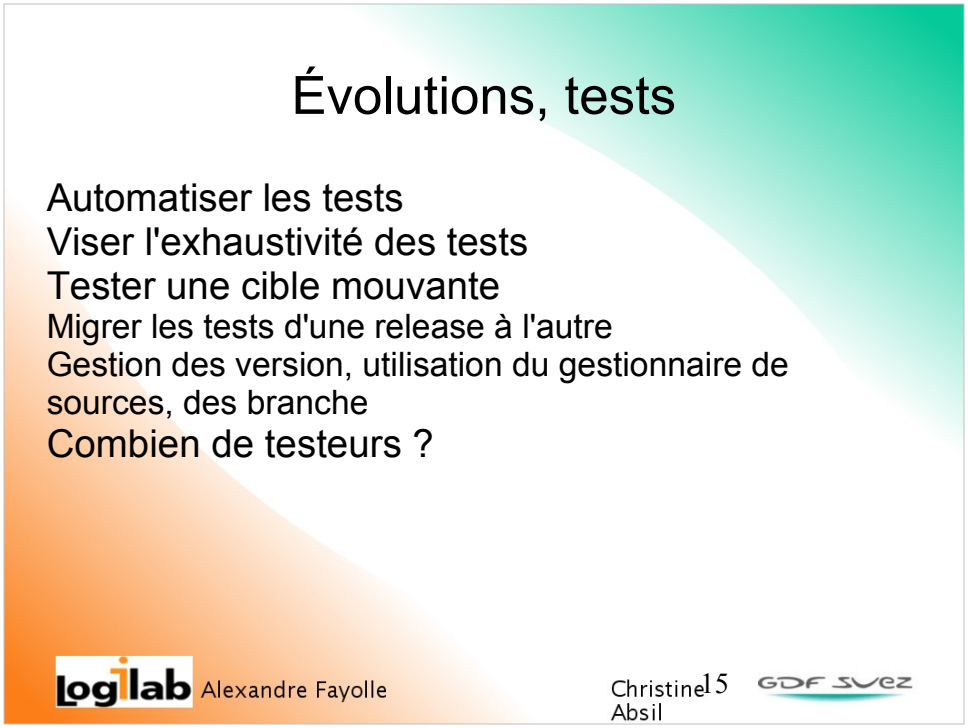
CA: Meme à 100%, ce n'est pas évident. Il y a beaucoup de chose à faire en temps. Product owner, ça demande une énorme implication. Comme, naïvement, je n'avais pas beaucoup détaillé mes Cards, l'équipe me posait beaucoup de questions auxquelles je ne savais pas directement répondre, par exemple sur la façon de calculer certaines valeurs dans des cas un peu tordus.

AF: Effectivement, mais le choix c'était: soit d'essayer de deviner (et franchement dans la plupart des cas, nous ne serions pas tombé juste), soit de te poser la question. Sur un projet Agile, on choisit de ne pas deviner, et d'utiliser le Product Owner pour obtenir les informations nécessaires juste à temps. Ca fait effectivement reposer un lourde responsabilité sur ses épaules.

Avec une petite équipe comme celle de Pylos, c'est gérable, mais avec une équipe plus grosse, on peut avoir besoin de s'organiser différemment. Par exemple en formalisant d'avantage les réunions de planification avec l'équipe, qui donnent lieu à la présentation des histoires.

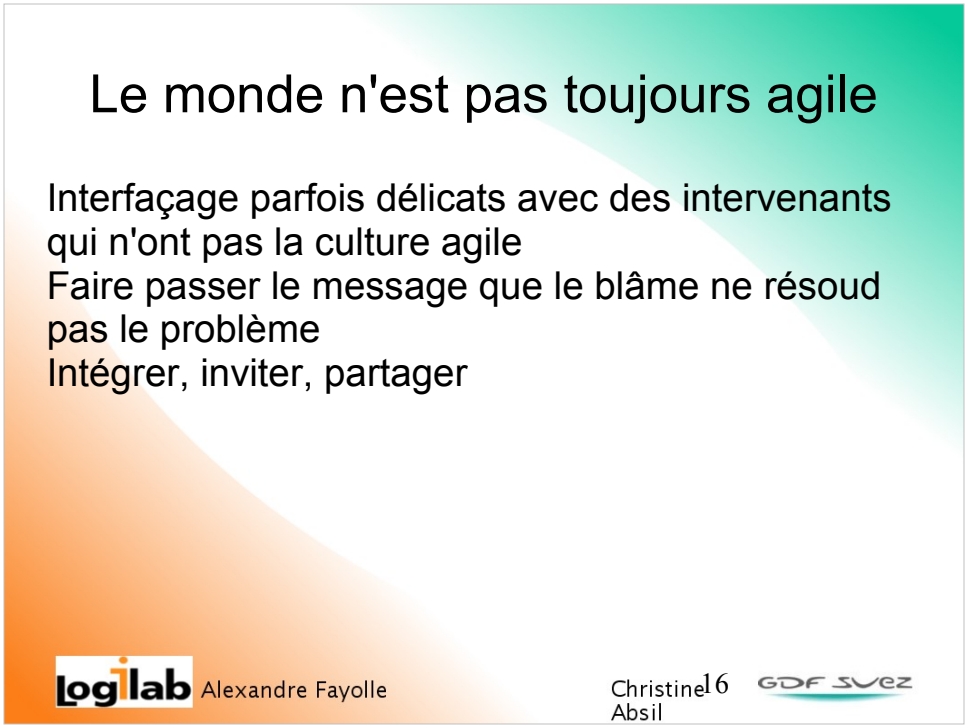
Les premières itérations du projet ont été stressantes également du coté des développeurs : on découvre un nouveau métier, avec son vocabulaire, ses conventions... Mais l'établissement d'un bon dialogue aide énormément.

CA: Sur le framework précédent, NERD, je n'ai vu ni les développeurs, ni les développements, pendant 4 mois. Au début, cela me convenait, parce que ça me libérait du temps pour d'autres activités. Mais au moment de la pré-livraison, qui s'est faite en retard, ça a été moins agréable: elle contenait bon nombre d'anomalies structurelles. Anomalies qui n'ont pas pu être corrigées dans la livraison définitive, quelques semaines plus tard. Mon manager a préféré arrêter le projet...



AF: Sur Pylos, les tests ont commencé beaucoup plus tot : david, patrick et toi, avez commencé assez rapidement à tester sporadiquement ; Sophie, votre Testinator, est arrivée en décembre, à mi-temps.
CA: Ceci dit, c'est pas évident non plus, de doser le nombre de personnes à affecter aux tests. Nous l'avons clairement sous-estimé. Heureusement, Koen est venu prêter main forte à Sophie depuis un mois
AF: Oui, c'est un petit bémol de l'agilité : quand on a des histoires qui viennent s'empiler les unes sur les autres, il est difficile de détecter les régressions.

Pour les développeurs qui vont avoir tendance à tester les chemins qu'ils savent fonctionner correctement, c'est facile de passer à coté...
CA: Et pour les testeurs, c'est pas évident de décider ce qu'il convient de tester, ou de retester. Ni quand le faire. Ils ont en outre toujours le sentiment d'avoir une bataille de retard...
AF: C'est plus simple à gérer quand toute l'équipe est au meme endroit parce qu'alors on peut raccourcir le délai entre implémentation et tests.
CA: On n'a pas pu organiser cela. Il faudra essayer de faire plus de sessions tous ensembles dans la suite du projet.
AF: Un autre point important du bilan, c'est la planification des livraisons successives et de leur contenu. Sur Pylos, on avait environ 80% des fonctionnalités de Pegase disponible dans la livraison 0.5 fin janvier. Si tu t'en souviens, ça nous a permis de faire une version 0.6 speciale "eye candy" pour une démonstration chez tes utilisateurs, avec plein de trucs cools comme une intégration google maps avec des icônes personnalisées, des sparklines pour visualiser les séries temporelles...
CA: Oui, ca a beaucoup plu. Pour cette version-là, on a changé le mode de planification: je vous ai demandé de me donner le maximum de fonctionnalités dans le temps imparti, alors que précédemment, je choisissais, pour chaque itération, un bloc de fonctionnalités cohérent, pour arriver le plus rapidement possible une applicaton utilisable



AF: Il y a eu des moments un peu plus tendus aussi.

CA: Tu penses à ces dernières semaines, avec les mauvaises surprises qu'on a eu au moment de l'installation sur les serveurs de notre service IT ?

AF: Par exemple. Ca a été difficile d'interfacer notre projet agile avec une équipe extérieure qui n'a pas du tout une culture agile.

CA: Ben oui. Je le savais depuis le début, ca: qu'on devrait se frotter à une organisation de 1.000 personnes, et à tous ses processes, peu flexibles. On a eu de la chance, on aurait pu tomber sur des gens bien moins agréables. Ils ont fait beaucoup d'efforts pour s'adapter.

AF: Effectivement, et les problèmes rencontrés n'étaient pas tous de leur fait. Le plus important de mon point de vue, c'est que nous avons créé une bonne dynamique d'équipe, et que la question n'est plus de savoir qui est fautif, mais bien de trouver une façon de corriger les problèmes, ensemble.

CA: A t'entendre, Pylos c'est le pays des Bisounours !

AF: Tu ne penses pas que c'est un projet où il fait bon travailler ?

CA: Maintenant que tu le dis... Il y a du taff, il y a des défis à relever, mais le fait de ne pas avoir à me battre contre mon fournisseur, de ne pas passer mon temps à prouver sa faute, ou à couvrir mes arrières, est agréable. L'atmosphère de confiance réciproque, la flexibilité de pouvoir m'adapter aux contraintes externes et managériales, tout ça rend le travail moins stressant, plus efficace et plus agréable. C'est même confortable pour moi de pouvoir renvoyer mon manager à ses propres responsabilités! Et s'il n'est pas 100% satisfait, je lui assure simplement qu'on fait de notre mieux, et que, à ce prix là ?!, il a les meilleurs...

Rideau !

C'est le moment de poser des questions

Questions importantes

Quelle durée d'itération ?
Comment découper en itération ?
Que faire quand le product owner se retrouve sur le chemin critique ?
Jusqu'à quelle taille de projet peut-on être agile ?

Goldratt : Theory of Constraints

Bilan

L'agilité ne garantie pas la réussite du projet, mais la simplicité des méthode rend les problèmes visibles plus vite

Les responsabilités de chacun demeurent, mais le contexte est à l'ouverture et à la transparence

- Le MOA doit vraiment savoir ce qu'il veut
- Le MOE ne peut pas jouer de l'effet tunnel

La planification est faite en mode « juste à temps » et « juste assez » → gain de temps

On ne s'échine plus à blâmer, au contraire on cherche à gagner ensemble