Présentation de la démarche DevTeam

Auteur : Marc Dexet
Présentation lors de la réunion DevTeam du 09/01/2017

1 INTRODUCTION

L'IAS produit, adapte et maintient des solutions logicielles dans le cadre de projets spatiaux. Afin de mieux répondre aux contraintes d'assurance qualité de plus en plus fortes, le service informatique de l'IAS veut améliorer ses processus de développement. Il souhaite en accroître la qualité et mettre en œuvre des pratiques pertinentes et reconnues, dénommées « bonnes pratiques ».

La littérature sur les façons d'améliorer le fonctionnement et « la performance » d'une structure est très riche, variée et parfois contradictoire. Les démarches participatives, lentes, régulières et incrémentales font néanmoins consensus.

L'objet de ce mémo est de présenter la démarche « DevTeam », ses raisons d'être et des propositions quant à son fonctionnement.

2 PRESENTATION DE LA DEMARCHE

Cette démarche est appelée « DevTeam », jeu de mots plus ou moins heureux sur Développement et Team.

2.1 DEFINITION

Elle est envisagée comme une démarche d'amélioration avec les objectifs suivants :

- Améliorer la qualité de nos développements
- Améliorer la confiance en notre travail
- Apporter une valeur ajoutée à nos utilisateurs
- Accroître le plaisir à ce que nous faisons
- Mettre en œuvre les bonnes pratiques

Elle s'adresse à l'ensemble des personnes impliquées dans le développement logiciel au sein de l'IAS.
Elle prend en compte tous les domaines, techniques et étapes du développement logiciel.
Elle s'appuie sur un ensemble cohérent de valeurs et de processus afin d'atteindre les objectifs.
Elle se veut collective, incitative, pragmatique et itérative.

Elle est animée et coordonnée par l'auteur sous la responsabilité du chef du service informatique avec le support des personnes volontaires.

3 PRECISION ET PREMIERES ITERATIONS

Afin de ne pas condamner la démarche dès le départ par excès d'optimisme ou d'ambition, il faut préciser les périmètres énoncés et les limiter à des dimensions plus réalistes lors des premières itérations.

3.1 ENSEMBLE DES PERSONNES CONCERNEES

Il faut considérer que la communauté des personnes concernées se scinde en deux groupes. Le premier groupe est constitué des développeurs, chefs de projet, ingénieurs système et responsables du service informatique. C'est le noyau actif, moteur et directement impacté par la démarche.

Le second groupe est celui des ingénieurs des autres services, chercheurs, thésards et stagiaires de l'IAS : Ils sont des bénéficiaires[1] et d'éventuels contributeurs.

Nous pourrons après quelques itérations envisager l'élargissement du groupe moteur au-delà du service informatique.

3.2 SUJETS D'INTERET

Par étape, nous entendons étape du cycle de vie d'une solution logicielle, c'est-à-dire les phases de spécification, de maquettage, de réalisation, de la validation, de déploiement, ou de mise à jour…

Par domaine lié au développement, nous entendons la modélisation, le codage, les tests, la documentation, le design d'interfaces, le packaging, …

Par technique de développement, nous entendons les langages, les paradigmes de programmation, les outils de développement (IDE, conteneurs), la gestion de la persistance (BDD), les méthodologies et les services de la forge.

Les frontières entre ces découpages sont perméables.

Par pragmatisme, les sujets en prise directe avec notre réalité, ou pouvant avoir un impact à court ou moyen terme sur celle-ci seront privilégiés, mais sans être au détriment de la prospective.

3.3 ENSEMBLE DE VALEURS

L'amélioration réelle et concrète nécessite une évolution de nos « façon de faire » et « façon d'être ». Il ne faut pas se limiter à l'outillage et à la technique : il faut également travailler sur la perception de nos métiers, de nos objectifs et de nos relations aux autres.

Cette amélioration passe par l'adhésion à un ensemble de principes, positions et prescriptions. Ce sont les valeurs formant le socle d'une culture commune.

Nous devons identifier ce qui est pertinent et ce qui ne l'est pas. Définir ce qui est obligatoire, interdit, recommandé, déconseillé ou facultatif en fonction[2] du niveau de compétence[3] ou de maturité[4] attendu. Ces valeurs doivent être objectives et cohérentes entre elles. Elles sont le résultat d'un consensus ou d'une expertise. Elles sont inscrites dans un référentiel identifié, reconnu et accessible.

---

[1] Par exemple en les sensibilisant aux bonnes pratiques et recommandations quant à leurs propres bases de code, via des sessions de formations dédiées ou la remise d'un guide.

[2] Ce qui est obligatoire pour un logiciel en phase de mise en production peut être facultatif pour un prototype.

[3] Par exemple : « Novice, débutant confirmé, efficace, compétent, expert » selon le modèle de Dreyfuss et Dreyfuss.

[4] Plusieurs niveaux de maturité sont concevables : par destination d'un logiciel « prototype, développement, production ». Par objectif de mise en œuvre de pratique : « compréhension et mise en place, pratique opérationnelle, pratique maîtrisée ».

Elles doivent être respectées autant que possible, mais peuvent être ignorées, modifiées ou redéfinies dans un cadre spécifique et justifiable.

Ce socle de culture permet de fluidifier les relations, faciliter les discussions et l'intégration de nouveaux arrivants. Il permet d'objectiver les non-dits ou attentes implicites, et offre une voie de résolution des conflits.

Il faut toutefois rester modeste et pragmatique. Cet ensemble de valeurs ne doit pas prétendre à l'exhaustivité et couvrir la totalité des sujets. Il doit rester opérationnel et vivant.

3.4 PROCESSUS

Les processus ne sont pas tous prédéfinis et restent à créer, c'est bien là tout l'intérêt de la démarche. Ils doivent être simples, légers, et intuitifs sous peine de freiner les contributions et d'être mal vécu ou mal perçus[5].

La majorité des processus commencent avec l'expression d'un besoin ou d'une proposition. Elle se fait par mail, lors d'une conversation, ou par une observation. Elle doit être précisée puis formalisée comme une entrée (ticket). C'est la phase de recueil.

En fonction de sa priorité ou de sa complexité, l'entrée est inscrite à l'ordre du jour de la prochaine réunion, nécessite une réunion exceptionnelle, la constitution d'un groupe de travail ou est simplement traité immédiatement par le responsable de la démarche. C'est la phase de traitement.

La phase de prise de décision fait suite à une analyse, une expertise ou un consensus par l'assemblée, un groupe de travail ou pour les cas les plus simples par le responsable de la démarche.

Les phases suivantes dépendent des processus.

En voici proposés sous forme d'esquisses (très) légères.

3.4.1 Processus d'inscription dans le référentiel

Les valeurs (prise de position, choix, principes, …) considérées comme pertinentes sont inscrites dans un référentiel collectif dédié.
La valeur est décrite par un nom, un descriptif, des liens vers des ressources exhaustives, un niveau d'exigence et éventuellement un niveau de compétence et de maturité. Elle peut faire l'objet d'un article plus élaboré si nécessaire. Elle est associée à une thématique du référentiel.
L'ouvrage, en version prépublié, est modifié en conséquence par un porteur. Avant publication, l'ouvrage et ses modifications sont revues. Après validation, les modifications sont publiées et la valeur est officiellement reconnue.

3.4.2 Processus de résolution des valeurs

Une personne exprime un problème de valeur ou une demande de précision. Après recueil, le problème est évalué lors du traitement et aboutit à une prise de position dont le résultat est inscrit dans le référentiel selon le processus de d'inscription.

5 La notion de processus ne doit être confondu avec la notion de procédure. C'est juste un moyen de mettre un nom sur un ensemble d'activités en vue d'obtenir des résultats. Comme M. Jourdain, nous faisons du processus sans le savoir.

3.4.3 Processus de résolution des points d'amélioration

Une personne soumet une proposition d'amélioration. Après recueil, la proposition est évaluée lors du traitement et aboutit à une prise de position dont le résultat est inscrit dans le référentiel selon le processus de d'inscription.

3.4.4 Processus de revue de code ou de procédure

Selon un processus à définir, une personne se voit proposer une revue de code ou de procédure. Cette revue est volontairement bienveillante et pédagogique. Elle sert de support à la démarche d'amélioration. La revue en soi est à définir car il en existe plusieurs sortes. A l'issue de la revue, les points relevés sont pris en compte par le créateur. Ils peuvent être notés comme proposition d'amélioration ou problème de valeur et rentrer dans le processus d'inscription.

3.4.5 Processus de montée en compétence

Une personne identifie un besoin de montée en compétence collectif. Après recueil, il est évalué lors du traitement. Il peut être adressé au correspondant formation, traité en interne ou faire l'objet d'un focus par un membre ou d'un intervenant extérieur proche (labos « amis »). L'action résultante peut faire l'objet d'une présentation, voire de valeurs à inscrire.

3.4.6 Processus de formation interne

Une personne offre ses services pour monter une formation interne. Sa proposition est inscrite à l'ordre du jour de la prochaine réunion ou faire l'objet d'une communication interne. En fonction de l'importance du publique ou du sujet, elle peut faire l'objet d'un support logistique (en accord avec le chef de service) ou être présentée via un réseau métier (Loops ou Argos ? ☺). L'action résultante peut faire l'objet d'une présentation, voire de valeurs à inscrire.

3.5 LES MOYENS ET OUTILS

La démarche s'appuie sur un certains nombres d'outils.

3.5.1 Les espaces dédiés

Il existe un groupe gitlab[6] et un espace redmine[7] DevTeam.
La position du incodewetrust[8] doit être précisée, mais il sert plutôt à valoriser les contributions personnelles.

L'espace Redmine est utilisé pour le suivi de la démarche. Son système de gestion des demandes permet de recueillir les besoins, construire les ordres du jour et effectuer le suivi.

Le groupe gitlab sert au développement des produits de la démarche.

3.5.2 La réunion DevTeam

Une réunion consacrée est régulièrement organisée au moins une fois par mois. Elle comprend un ordre du jour minimal, un compte rendu collectif et un plan d'action mis à disposition dans l'espace Redmine et décliné en issues Redmine.
Cette réunion doit faciliter les échanges et les partages. Tout le monde est encouragé à y contribuer et y présenter ses points de vue, ses retours d'expérience, ou ses veilles technologiques. Malgré le formalisme, c'est la convivialité qui en déterminera le succès.

[6] https://git.ias.u-psud.fr/DevTeam
[7] https://idoc-projets.ias.u-psud.fr/redmine/projects/devteam
[8] https://incodewetrust.ias.u-psud.fr/

Selon les besoins, d'autres réunions complémentaires et ciblées peuvent être organisées plus fréquemment.

L'auteur suggère fortement de favoriser des réunions fréquentes et courtes.

3.5.3 Les guides référentiels

La construction de valeurs communes se fera à travers des ouvrages collectifs, à la fois guide et référentiel.

Ces guides traitent des points essentiels de leur domaine, centralisent en un lieu unique les valeurs jugées pertinentes. Ils fournissent un ensemble d'opinions utiles à ceux qui n'ont pas d'avis ou veulent faire l'économie d'une réflexion sur des sujets. Ils proposent des solutions pragmatiques et opérationnelles comme des modèles, des pointeurs ou des checklists avec plusieurs niveaux de compétences ou de maturité.

Hébergé par le gitlab du laboratoire, et rédigés sous forme textes (asciidoc), l'auteur propose de les traiter comme des codes, avec gestion de branches, cycle de mise à jour et de publication.

Le contenu sera publiable sous forme de site web statique ou de PDF. La gestion sous forme de backlog des tickets gitlab permet de structurer le contenu éditorial et de dérouler le processus d'inscription.

Ces guides sont

- le guide du développement https://git.ias.u-psud.fr/DevTeam/DevGuide
- le guide de la documentation https://git.ias.u-psud.fr/DevTeam/DocumentationGuide

L'exercice d'établissement d'une table des matières est une bonne entrée en la matière pour discerner les points importants.

3.5.4 Les services de la forge

Les services de la forge sont des outils privilégiés pour l'application des bonnes pratiques, car leur utilisation implique d'en respecter un certain nombre. Le service Sonar notamment permet via des indicateurs d'en suivre la mise en œuvre.

3.5.5 Les focus

Les focus sont des ateliers de formation internes de 1 à 2h, sur des sujets à approfondir. Cela est pris en charge par le processus de montée en compétences. Préparés régulièrement ils constituent un outil assez efficace.

Exemple de thèmes identifiables : les base de la programmation objet, les git-flow, dockerisation d'une base de donnée, jobs gitlab, job jenkins, fonctionnement du protocole http, …

4 PROPOSITIONS D'AXES DE TRAVAIL

Bien que la démarche repose sur la prise en charge des points remontés, il est important que des fils conducteurs en sous-tendent la progression.

Voici des propositions d'axes de travail à discuter et prioriser

- Réduction de la dette technique

    - Introduction aux règles de codage et de conception[9]
    - Mise en œuvre des tests unitaires
    - Utilisation de la forge logicielle
- Gestion de la documentation
    - La gestion de la documentation technique d'un projet
    - Factorisation et automatisation de la production de documents qualité demandés par les agences
- Gestion des environnements de développement
    - Du poste du développeur à la mise en production, identification des freins à l'agilité
    - Indépendance des développements : virtualisation, conteneurisation, intégration.

L'auteur propose, lors des premières itérations, afin de « roder » le processus, de s'astreindre à traiter un sujet ou deux dans les axes de travail suggérés ultérieurement.

5 APPROCHE EGOLESS

Pour assurer le bon fonctionnement de la démarche, l'auteur propose de souscrire à l'approche « Egoless ».

Notre travail est une part importante de notre vie et à travers lui, nous nous construisons une image de nous-même. La démarche d'amélioration, ses évaluations, revues et décisions peuvent être source de conflit. Un ensemble de valeurs communes peut aider, à condition d'avoir une attitude bienveillante envers les gens et critique avec leur travail. C'est l'objectif de l'approche Egoless.

Egoless est très utilisée dans les méthodes agiles, le pair-programing, le monde DevOps ou par les Géants de l'Internet (GAFA). Elle repose sur 10 principes de conduite par rapport à soi et aux autres extraits de « The Psychology of Computer Programing », Weinberg, 1971

Nous ne verrons que les plus pertinents pour la démarche.

5.1 « UNDERSTAND AND ACCEPT THAT YOU WILL MAKE MISTAKES »

Tout le monde fait des erreurs, tout le temps. Seuls le nombre, la densité, la gravité changent avec l'expérience et la compétence. Le nier est catastrophique, l'accepter permet de les gérer et de limiter leurs conséquences au plus tôt. Non seulement ce n'est pas un sujet de honte, mais c'est une source d'apprentissage. Il faut essayer et échouer pour progresser. Montrer ses erreurs permet de les analyser, d'en discuter, d'en tirer les conséquences et d'augmenter la connaissance du collectif.

5.2 « YOU'RE NOT YOUR CODE »

Nous ne sommes pas notre production. Ce principe est un corolaire du premier. Il faut dépassionner la relation à notre production. Dans la chaîne qui va de l'idée à la mise en œuvre, de la conception au codage, vous introduisez peut-être des erreurs. Ne vous sentez pas insulté par la revue de votre travail. Ce n'est pas jeter un doute sur vos qualités, c'est un moyen de s'assurer contre les conséquences. C'est aussi une occasion pour vous d'apprendre et de gagner en confiance.

[9] A partir des ouvrages « clean code » et « pragmatic programmer »

5.3 « No matter how much "karate" you know, someone else will always know more »

Derrière ce titre énigmatique, il y a des principes très simples d'humilité et d'écoute. Ne pensez jamais avoir fait le tour d'un sujet, il existe toujours des aspects dont vous ignorez l'existence. En restant attentif, vous pouvez apprendre des « techniques » de personnes auxquelles vous ne pensiez pas.

5.4 « Fight for what you believe, but gracefully accept defeat »

Vous devez porter vos idées avec conviction par une argumentation solide. Mais vous devez accepter de perdre un débat d'idées. Ce n'est pas que vous ayez forcément tort, c'est que d'autres idées ont été jugées plus pertinentes. Il ne faut pas en garder rancune, cela défavoriserait vos idées et nuirait à l'ambiance. Par contre, gardez vos notes et le sourire, la roue tourne, le choix fait peut-être inadapté, une nouvelle situation se présenter et vos idées pourraient alors être entendues.

5.5 « Critique code instead of people – be kind to the coder, not to the code »

C'est un principe essentiel du regard sur autrui et son travail. Sans lui l'approche Egoless n'a plus de sens. L'objectif est d'améliorer la qualité du travail, d'amener l'autre à gagner en compétences et en confiance. Etre bienveillant et encourageant permet d'affirmer la relation de confiance et le respect, d'éviter de générer des situations ingérables et de permettre la réciprocité : la personne dont le travail est évalué sera évaluateur à son tour.

Cela favorise la sincérité du regard et l'amélioration, car pour que l'approche fonctionne, il ne faut pas avoir peur de donner son avis ni peur d'en recevoir. Il faut avoir un regard acéré et factuel sur le travail d'autrui.

6 ANNEXES

6.1 LES 10 PRINCIPES EGOLESS

1. Understand and accept that you will make mistakes.
2. You are not your code.
3. No matter how much "karate" you know, someone else will always know more.
4. Don't rewrite code without consultation.
5. Treat people who know less than you with respect, deference, and patience.
6. The only constant in the world is change.
7. The only true authority stems from knowledge, not from position.
8. Fight for what you believe, but gracefully accept defeat.
9. Don't be "the guy in the room."
10. Critique code instead of people – be kind to the coder, not to the code.

Pour plus de détails

- https://blog.codinghorror.com/the-ten-commandments-of-egoless-programming/
- https://blog.octo.com/egoless-programming/
- http://codingwithempathy.com/2016/03/01/reflections-on-the-ten-commandments-of-egoless-programming