Comment être #TechLead dans une pizza team XXL sans finir sous l'eau

Speaker : Damien Beaufils (Octo Technology)
Format : Conférence
Date : 7 avril 2017

Retour d'XP de 1 an dans une grosse équipe.

Objectif : quel rituel et pratique mettre en place dans une équipe de plus de 10 personnes.

Définition d'un Tech Lead : au service de l'équipe avec 4 facettes coach, expert, formateur et facilitateur.
Damien considère qu'**un Tech Lead doit passer 1/3 de son temps à coder.** Permet de partager les mêmes problèmes que les autres codeurs.
Le Tech Lead doit idéalement être référent sur les pratiques Sotfware Craftmanship.
Un Tech Lead n'est ni un super-héros ni un dictateur.

Contexte du projet :

- Site grand public
- Equipe de 10 développeurs + tech lead. 20 personnes au total
- Projet démarré depuis 1 an lorsque Damien l'a rejoint
- 1 itération = 1 release + 4 patchs

Plus il y'a de personnes, plus le nombre de liens de communication augmente.

Phase d'observation pendant 1 semaine : pair programming en binommage tout en taisant son égo et ses réflexes. Ne pas arriver avec ses gros sabots : lever de bouclier + pas le contexte du pourquoi du comment.

Signaux détectés :

1. Commits passés tards le soir, voir la nuit
2. 5 livrables par itération
3. Nombre de bugs dans le backlog en augmentation
4. Le « build est rouge mais c'est normal »
5. « Tester unitairement c'est compliqué »
6. « On reçoit tellement de mails de Jenkins qu'ils vont directement dans la corbeille »
7. « Hier j'ai passé toute l'après-midi à réparer le code »

Le Tech Lead doit chercher la root cause :

- Propriété du code non collective
- Pratiques de dév hétéorgènes
- La qualité du produit n'est ni mesuré ni suivi

Pour améliorer la propriété collective du code :

1. Rétrospective technique :
    a. 1 heure, tous les lundis. Evite le rush de fin de semaine. Une heure ne pose pas trop de problème au management

    b. On colle les problèmes sur un tableau qu'on dépile. Evite de déranger tout le monde
    c. Objectif : partager le maximum de connaissances en un minimum de temps
2. Standards de code
    a. Affichés sur une « Table de la loi » sur un mur : management visuel. Ne souhaitait pas un wiki
3. Revue de code
    a. Rendue obligatoire et bloquante
    b. Présentation orale des retours à l'auteur-e
4. Pair / Mob programming
    a. Obligatoire quand on travaille sur du code legacy (code sans test). C'est arrivé suite à un bug majeur de prod. Permet d'éviter les régressions et partager des techniques de refactoring du code legacy

Pourquoi avoir une propriété du code collective ?
Toute personne peut quitter l'équipe du jour au lendemain.

Comment savoir si c'est le cas ?
N'importe qui peut partir en congés n'importe quand.

Quid de la qualité ?

Pyramide des tests : il faut plus de TU que de tests d'intégration, fonctionnels, end-to-end.
Les TU sont plus stables et plus rapides.
Dans l'équipe, personne n'était d'accord sur la définition d'un TU.
Atelier sur la définition d'un TU, TI, TF. Quels objectifs et intentions via ces tests ? Comment je les reconnais dans le code ? Est-ce que je peux tester autrement ?

Lorsqu'un test charge tout le contexte Spring -> pas un TU.
Ils ont sélectionné des tests faisant référence.

Ils ont fait le choix de ne pas mesurer la couverture de tests. Raison : ce n'est pas un indicateur de qualité. Au mieux, il s'agit d'un indicateur de non-qualité. En effet, ne dit rien sur les assertions faites dans les tests. 2nde raison : ne pas donner le bâton pour se faire battre. Ne doit pas être un objectif. Permet d'éviter les débats stériles : pourquoi 60% et pas 80% ?

Ils ont écrit un script pour déterminer leur pyramide de tests en catégorisant les tests unitaires. Résultat : la base de TU de 18% n'était pas assez solide.

Investissement à la pratique de TDD pour les 10 développeurs, quel que soit le niveau.
Formation de 3 jours. Rôle du technique : donner l'envie aux développeurs et démontrer le ROI de TDD par des données factuelles.
Une formation collective sur le TDD coute moins cher qu'une formation individuelle par du coaching ?
Rôle du tech lead : vendre une démarche d'amélioration de la qualité aux personnes qui ont les sous.

Mentorer chaque développeur-euse via des **One-on-One** (O3)

- 30 personnes par personne, par itération
- Un moment privilégié d'échange, ritualisé
- Objectifs :
    - Récupérer de l'information, prise de température (ex : le dev s'ennuie)
    - Donner du feedback (positif ou d'ajustement)
    - Rechercher des axes de progression (ex : un dév souhaite faire de l'Ops mais ne comprend pas du tous les tickets dans le backlog). Rôle du Tech Lead : re-router la personne. Lui proposer d'aller voir telle personne afin de faire le prochain billet en pair-programming
    - Déléguer : permet de responsabiliser l'équipe de dév en leur déléguant des tâches de tech lead

Etre attentif à la communication dans l'équipe

- Recadrer les défauts de communication
    - Ex : « Ton code » vs « Le code »
    - Principes Egoless programming
- Quand on travaille dans une équipe mixte homme/femme, avoir un discours appelant à la diversité
    - Ex : à l'oral, éviter le « Allez les gars ! ». Proposition : « Allez les gens ! », « Allez l'équipe »
    - Ex : à l'écrit, éviter le « les développeurs ». Proposition écrite : « développeur-euse-s » ou « équipe de développement »

Tests

Les tests fonctionnels instables ont été supprimés après avoir été redescendus dans les étages de la pyramide.
La pyramide de test a mis 6 mois à revenir à l'état de l'art. Cela s'est fait au fil de l'eau grâce à la pratique du TDD et du pair-programming.
Suite à la remarque d'un sponsor « Pourquoi ne pas faire que des tests fonctionnels ? », ils ont revu leur discours.
Mesurer le tests de bug restant à corriger encourage l'équipe lorsque la courbe descend (plus de bugs corrigés qu'introduits). En un an, ils ont gagné 15 bugs.
Le nombre de livraisons nécessaires par itération est mesuré. Faire des patchs fait perdre du temps à tout le monde. Moyenne passée de 4 livraisons à 1,5 par itération.

Produire du code de qualité a un impact social : fierté des qualités, peu de rush.