La boite à outils DevOps, ce qu'il y a dedans va vous surprendre…

Winael

2020-02-24T15:18:42+01:00

Suite à la publication d'un article sur ce qu'est DevOps (et ce qu'il n'est pas), un des auteurs de commentaires a présenté l'infographie suivante comme étant un panel d'outils DevOps :

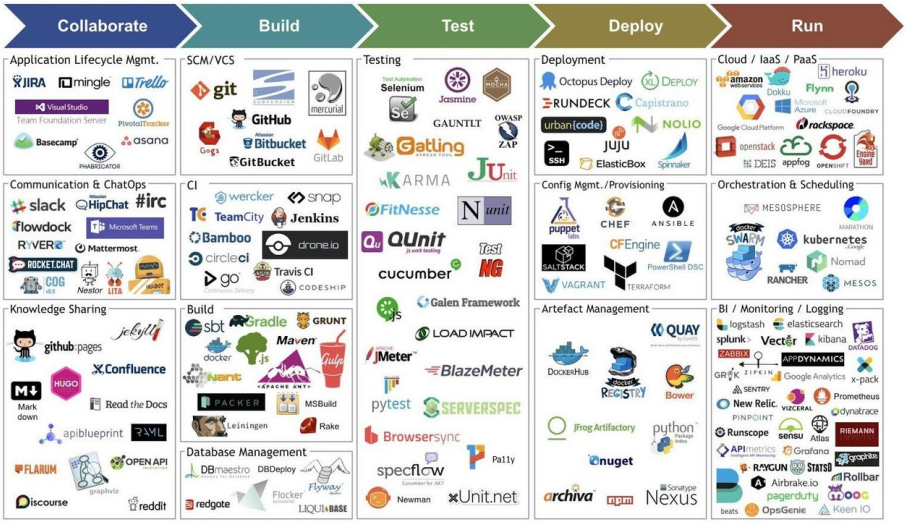

Figure 1: Outils "DevOps"

On y retrouve un certain nombre d'outils, utilisés quotidiennement par de nombreux acteurs IT, qui permettent ou améliorent le travail collaboratif pour certains de ces outils, l'automatisation pour d'autres, la gestion de versions de code, la gestion de configuration etc.

La question est donc de savoir si ces outils peuvent-être considérés comme DevOps ou non.

Didier, Fadia et Joann... utilisateurs d'outils "devops"

Didier est développeur. Il appartient à une équipe qui travaille sur un produit P. Cette équipe à son "PO" dédié, son "Scrum Master" dédié. Le code est poussé sans un **dépôt Git**, puis Didier exécute une **pipeline Jenkins** pour builder son composant, son **image Docker**, qui sera ensuite déployée dans un **cluster Kubernetes**.

Fadia est intégratrice. C'est elle qui est chargée de créer la/les pipelines pour les équipes de développeurs en fonction de leurs besoins (Créer des **buckets Couchbase**, et ajouter les configuration dans la pipeline, permettre à la pipeline d'utiliser **SonarQube pour les tests automatisés**, etc). L'équipe de Fadia recevant des demandes de différentes équipes de développement gère son backlog selon les principes Kanban. Chaque personne prends des tâches qui sont demandées à l'équipe d'intégration. L'équipe de Fadia à son "PO" et "Scrum Master" dédié. **L'équipe de Fadia fournit du service à l'équipe de Didier.**

Joann est lui dans l'équipe de production. Il s'occupe de l'exploitation des solutions développées par la société dans laquelle travaillent Didier, Fadia et Joann. **Il a en charge le bon fonctionnement des applications**, voire l'évolution des pipelines mise en place par l'équipe de Fadia. L'équipe de Joann, de par ses contraintes de production **travaillent selon les principes Kanban**. Un référent est associé à chaque équipe de développement pour assurer une meilleure gestion des incidents et des évolutions. L'équipe de Joann n'a pas de "PO", mais à un "Scrum Master" dédié.

On voit bien que les outils utilisés sont inclus dans l'infographie. Pour autant, on est très loin de DevOps. Toutes les équipes sont silotées, ne partagent par les objectifs, et pire non pas réellement de gouvernance commune.

Le marteau ne fait pas l'artisan, les outils d'automatisations ne font pas le DevOps

Il y a quelques années, j'ai reçu pour Noël une superbe boite à outils, contenant perceuses, marteaux, tournevis, scies… Et pourtant, dès que j'ai des travaux à faire à mon domicile, je suis souvent obligé d'appeler mon père à la rescousse. **J'ai beau avoir tous les outils du parfait bricoleur... ça ne fait pas de moi un bricoleur.** Je n'ai jamais appris, je ne sais même pas choisir le bon forêt en fonction du mur que je dois percer.

Pour DevOps, c'est pareil, vous avez beau avoir tous les outils de l'infographie ci-dessus, si vous n'avez pas appris à travailler selon la philosophie DevOps, vous

risquez, comme moi avec mon mur, de faire de sacré dégâts : Pas de prises en compte de l'aspect sécurité au moment du développement, des tests automatisés, de la production, du temps de mise sur le marché, de la qualité produit.

C'est au moment du lancement du produit que l'on s'aperçoit que les logs sont mal formatées et mal prises en charges dans le concentrateur de logs et qu'il est difficile de créer des tableaux de bords ou des alertes efficace.

On s'aperçoit également que le cycle de vie, et le patch management n'a pas été implémenté, que l'application n'est pas assez bien testées et que **de gros problèmes de fond vont présenter des risques**.

Le temps passé à corriger ces problèmes de fond risque de faire passer le produit à côté de son marché potentiel, une application concurrente étant arrivée entre temps et proposant une offre ayant séduit votre cœur de cible. Et si tout cela aurait pu être évité dès le commencement du projet ?

Scrum, un exemple de boite à outils DevOps

Contrairement à ce qui est répandu un peu partout, **Scrum N'EST PAS une méthode, ni moins une méthodologie Agile**.

Tout d'abord, ce n'est pas une méthode ou méthodologie, car c'est un **cadre de processus léger**, une sorte de boite à outils proposant des pratiques recommandées et n'imposant que peu de choses comme ses rôles, et ses cérémonies, qui permettent le mieux travailler ensemble.

D'autre part, Scrum n'est pas Agile, car ce cadre de processus est apparu plusieurs années avant le Manifeste du Développement Agile de Logiciel et a été **conçu pour le développement, la livraison et la maintenance de produits complexes, y compris donc les projets non-informatique**. Il est donc possible d'utiliser Scrum dans le cadre du développement d'une fusée ou d'une campagne marketing.

Mais alors pourquoi proposer Scrum comme boite à outils DevOps, si Scrum n'est pas Agile ?

Lors de son intervention à Agile en Seine 2018, Guillaume Lepetit, ancien DSI de TF1 qui a mis en place DevOps dans ce groupe média, avait eu cette phrase délicieuse : *"Ils (les collaborateurs) allaient pas pouvoir faire du DevOps sans avoir les bases de l'Agilité."*.

Or d'après Diana Larsen et James Shore, auteurs du Modèle d'Aisance Agile, **la première zone à franchir pour qu'une organisation devienne Agile, est la zone dite de Concentration ("Focusing")**. C'est dans cette zone que l'on va se **concentrer sur la production de valeur métier**. Elle s'appuie

notamment sur l'apprentissage qu'une équipe peut tirer de cadre de processus tel que **Scrum**.

Les spécificités de Scrum sont donc toutes indiquées pour qu'une équipe puisse commencer à mettre DevOps en place :

En Scrum, il est nécessaire pour une équipe de **regrouper l'ensemble des compétences nécessaires au développement du produit**. Ce qui veut dire pas seulement les devs, pas seulement les devs + les ops, mais aussi les personnes compétentes en monitoring, en sécurité, voire en UX également. Peut-être que certaines compétences seront moins nécessaires sur certains sprint. Dans ce cas, les personnes ayant ces compétences seront intégrées dans l'équipes de développement sur les sprints où ces compétences seront nécessaires.

Il faut bien comprendre que **le terme équipe de développement ne veut pas dire *"équipe de developpeurs"***. L'autre notion à retenir, c'est que **c'est toujours l'équipe dans son ensemble, qui est responsable de l'incrément produit potentiellement livrable**, peut importe les profils qui l'a compose. Les Devs, les Ops, les Secs sont autant responsable de l'incrément produit.

Il n'y a donc qu'un seul backlog pour l'ensemble des tâches, DEV, SEC, OPS, QA et **l'équipe partage les même objectif et chacun de ses membres doit être aligné sur la même vision produit** dont le PO à la charge.

Scrum s'appuie sur 3 piliers : Transparence, Inspection et Adaptation.

Ainsi l'une des cérémonies les plus importantes en Scrum est la **Rétrospective**. Elle permet à l'équipe, à chaque fin de sprint, de **réfléchir à l'amélioration continue du travailler ensemble, en se basant sur les expériences passées**.

L'équipe Scrum acquiert donc, sprint après sprint, une **culture commune**, de la **transparence** et du **partage**, basé sur les **mesures du passé**. Et puisqu'elle doit être en mesure de pouvoir livrer l'incrément du produit fini (répondant à tous les critères d'acceptation à chaque fin de sprint), l'**automatisation**, et la **sécurité en sont donc des pré-requis**.

Scrum permet de mettre en place facilement les **4 piliers DevOps**, c'est pourquoi ce cadre de processus peut être considéré comme **boite à outils DevOps**.

En mettant en place Scrum, Didier, Fadia et Joann vont donc se retrouver dans la **même équipe**, **partager les même objectifs**, avec un **backlog unique** avec **un seul PO** et un seul **Scrum Master unique**.

Ils travailleront enfin ensemble et non séparé et deviendront DevOps.

Retrouvez l'article en epub ou pdf