DEVOPS ET MÉTHODOLOGIE AGILE

Développement logiciel agile et le DevOps

Le développement agile est un terme générique pour plusieurs méthodologies de développement logiciel. Les méthodologies les plus populaires incluent Scrum, Kanban, Scaled Agile Framework, Lean Development et Extreme Programming (XP).

Bien que chacune des méthodologies agiles soit unique dans son approche distinctive, elles distribuent toutes une représentation commune et des valeurs absolues que vous retrouverez dans le manifeste agile. Ils incorporent tous essentiellement l'itération et la rétroaction continue : planification continue, des tests continus, une intégration continue et d'autres formes d'évolution continue du projet et du logiciel qu'elle fournit pour **affiner et livrer constamment un système logiciel**. Plus important encore, les méthodes agiles se concentrent toutes incontestablement sur la capacité des personnes à coopérer et à **prendre des décisions collectivement**, rapidement et efficacement.

Au départ, les équipes agiles étaient spécialement composées de développeurs. Et au fur et à mesure, l'agile a grandi pour englober d'autres équipes comme les équipes de test QA (Quality Assurance) afin d'augmenter la vitesse de livraison des logiciels.

Maintenant, l'agile se développe à nouveau pour englober les membres de la livraison et de support afin d'étendre l'agilité sur la globalité de la livraison. L'idée derrière le DevOps est d'étendre cette pratique en incluant l'équipe d'exploitation (OPS) et en misant sur un état d'esprit qui encourage la **communication**, la

collaboration, l'**intégration** et l'**automatisation** entre les développeurs et les opérations informatiques afin d' améliorer la vitesse et la qualité de livraison des logiciels.

Les équipes d'opération adoptant une démarche DevOps se concentrent sur l'automatisation des processus de livraison pour **améliorer la fréquence, la vitesse, la sécurité et la maintenabilité d'une livraison de logiciel**. Pour cela, ils offrent aux développeurs une meilleure visibilité de l'infrastructure de production et incite à donner ces équipes l'indépendance essentielle pour créer, valider, livrer et prendre en charge leurs propres applications.

Exemple de scénarios

Dans cette partie, nous allons examiner le scénario d'une équipe n'optant pas pour la démarche DevOps et inversement. Cela nous permettra d'avoir une vision plus claire sur ce que le DevOps peut accomplir comme prouesse.

Exemple d'un scénario sans le DevOps

L'équipe de développement qui a pour objectif de livrer rapidement autant de fonctionnalités que possible, envoie à l'équipe d'opération une nouvelle version de leur application sans avoir effectué de tests automatisés. En même temps, l'équipe d'opération effectue une mise à jour de leur système sans prévenir l'équipe de développement.

L'objectif de l'équipe d'opération est de limiter les dégâts de leur système. Ils livrent avec appréhension le code des développeurs et le système plante. Les opérateurs blâment les développeurs en leur disant qu'ils ont fourni du code défectueux. Les développeurs quant à eux se mettent sur la défensive et précisent que l'application fonctionnait très bien dans leur environnement de test.

Au bout du compte, l'application est désormais inopérante dans l'environnement de production et l'équipe d'opération commence à déboguer* le système et à stabiliser la production. Pendant ce temps, l'équipe de développement ne peut plus livrer en production de nouvelles applications.

Définition

Déboguer : action qui consiste à analyser les bugs d'un programme informatique

Exemple d'un scénario avec le DevOps

Les équipes de développement et d'opérations se réunissent avant de commencer leur nouveau projet logiciel et prévoient et analysent la façon dont ils vont créer un logiciel fonctionnel prêt à être déployé.

Ils conçoivent ensemble les tests à effectuer et l'infrastructure à utiliser et à gérer. Les besoins de chacun sont clairs, bien définis et reconnus par toutes les équipes. Ils continuent alors de travailler comme ceci tout au long du projet.

Chaque jour, un nouveau code est déployé au fur et à mesure que les développeurs le complètent. Les tests automatisés garantissent que le code est prêt à être déployé. Une fois que le code a réussi tous les tests automatisés, il est déployé auprès d'un petit nombre d'utilisateurs. Le nouveau code est surveillé pendant une courte période pour s'assurer qu'il n'y a pas de problèmes imprévus. Il est ensuite exposé aux utilisateurs restants une fois que la surveillance montre qu'il est stable. Beaucoup, sinon la totalité, des étapes après la planification et le développement se font sans intervention manuelle afin de **réduire les erreurs**

humaines.

Quelles sont les valeurs du DevOps ?

Le DevOps se concentre fortement sur l'établissement d'une culture collaborative et l'amélioration de l'efficacité grâce à des outils d'automatisation. Alors que certaines organisations et personnes peuvent avoir tendance à valoriser l'une plus que l'autre, la réalité est qu'il faut une **combinaison de culture et d'outils** pour réussir l'adoption du modèle DevOps. Voici ce qu'il convient de savoir sur ces deux principales valeurs du DevOps.

Culture DevOps

De nombreuses valeurs DevOps se reposent sur des valeurs agiles, car le DevOps est une extension d'Agile. La culture DevOps se caractérise par une collaboration engagée, une responsabilité partagée, des équipes plus autonomes, une réduction des silos, l'amélioration de la qualité des livraisons et l'intégration de l'automatisation.

Dans les méthodes agiles, les équipes telles que les développeurs, les testeurs et les UX designers travaillent et collaborent sur les mêmes objectifs. Le DevOps inclut simplement les équipes d'opérations dans les processus et objectifs de cette équipe agile. Pour y parvenir, Dev et Ops doivent **briser les silos**, incontestablement collaborer les uns avec les autres, partager les mêmes responsabilités et trouver ensemble la meilleure façon de faire fonctionner, sécuriser et livrer les mises à jour logiciels avec une rétroaction de qualité accrue et une automatisation de la livraison.

Outils DevOps

Les outils DevOps comprennent l’automatisation, la gestion et la configuration de l’infrastructure, des systèmes de test, de build, de déploiement, de contrôle des versions et des outils de surveillance.

Cependant, j’ai vite remarqué que certaines organisations perçoivent souvent le DevOps comme un outil lorsqu'elles commencent à importer cette démarche dans leurs équipes. L'adoption de l'automatisation et de l'outillage fait bien sûr partie du DevOps, mais uniquement lorsqu'elle est combinée avec des pratiques de bout en bout de collaboration.