DevOps

> *A la fin de ce document on sera capable d'identifier les méthodes d'implémentations du DevOps en entreprise.*

Imaginons que l'on vient d'arriver dans une entreprise où l'on nous demande de mettre en place une démarche DevOps et les outils associés.

Les principaux avantages du DevOps sont :

- l'accélération des déploiements applicatifs
- la réduction du Time-to-Market.

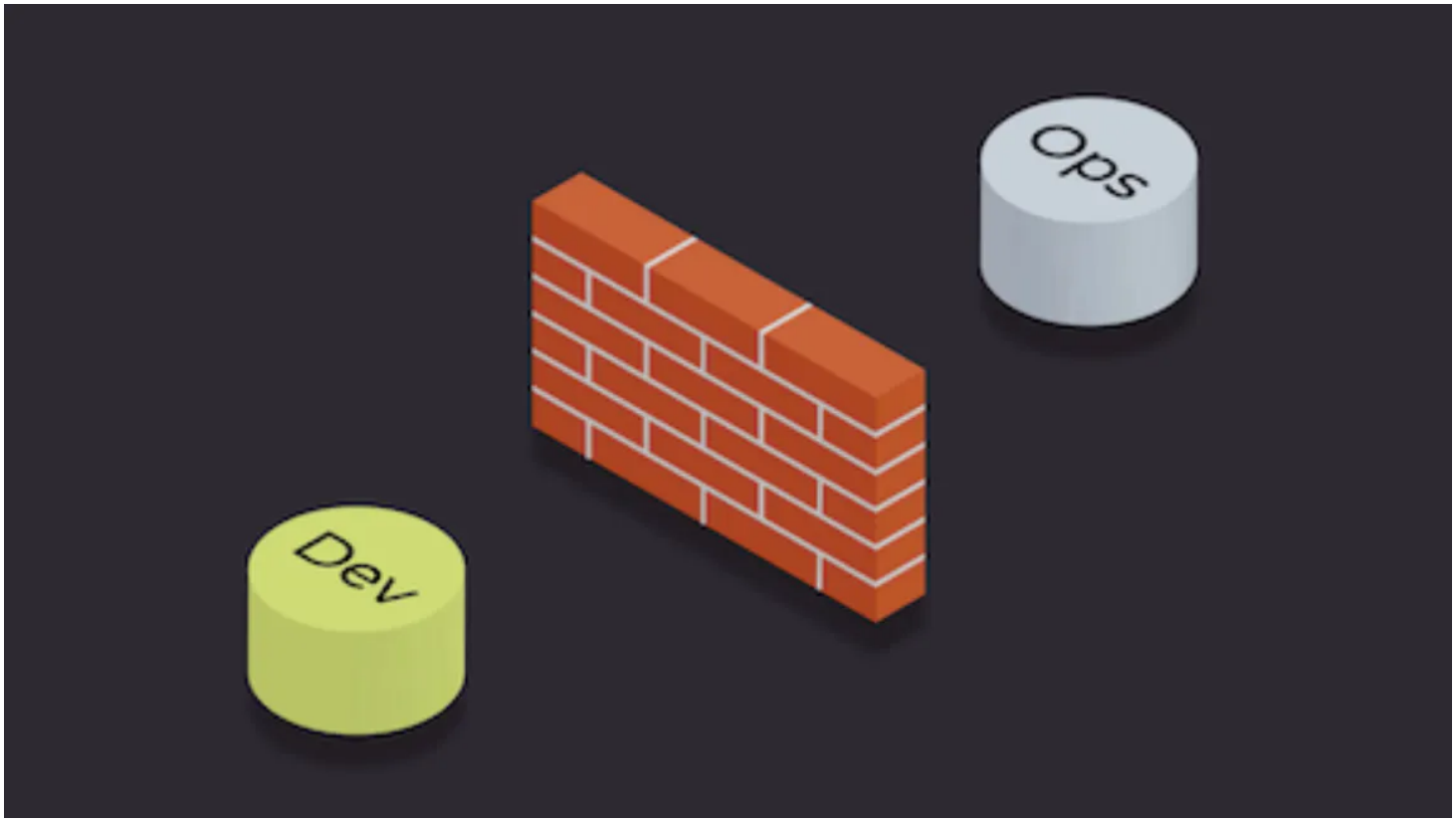
> *Le time-to-market est le temps nécessaire pour qu'un produit soit développé, fabriqué et disponible à le marché.*

Historique du DevOps

Il faut savoir que dans les années 2000, les équipes de développement et d'opérations étaient séparées. Les développeurs développaient et les opérations déployaient parce que chacun avaient des

Light

responsabilités différentes. Le mur de la confusion a donc été créé entre les deux équipes :



> *Le mur de la confusion est un terme utilisé pour décrire la situation où les équipes de développement et d'opérations ne se comprennent pas et ne travaillent pas ensemble. Ils ont donc des objectifs antagonistes.*

Donc **l'équipe de développement** a pour **objectif** de **faire évoluer l'application** en ajoutant des fonctionnalités, en corrigeant des bugs, le plus rapidement possible et cela **dans un délai et coût moindre**.

Tant dit que **l'équipe d'opérations** a pour **objectif** de **maintenir l'application en production et donc opérationnelle**. Tout cela en garantissant la sécurité, la stabilité de l'application et la qualité du service.

> *DevOps est donc un ensemble de pratiques qui met l'emphase sur l'automatisation des processus entres les équipe de développement et d'opérations.*

Ces pratiques permettent de **développer**, **tester**, et **livrer les applications** avec une plus grande **rapidité **, fréquence et **fiabilité**.

Les avantages du DevOps sont :

- gain de confiance des équipes entre elles
- accélération des livraisons et des déploiements
- résolution des tickets plus rapide
- gestion plus efficace des tâches non planifiées
- créer des environnements à la demande.

> *Une conséquence indirecte de l'implémentation de cette culture DevOps est ce que l'on appelle le mean time to recovery (MTTR), c'est-à-dire la capacité à rendre à nouveau opérationnel un système. Entre la détection d'un bug de production, et la correction de celui-ci, il se passe maintenant au maximum 4 minutes, contre plusieurs heures auparavant. Cela démontre vraiment la puissance du DevOps.*

CALMS

Culture

Le DevOps aborde principalement les **défis humains** liés à la communication et à la répartition des responsabilités entre les équipes. Dans cette optique, le DevOps s'aligne sur l'approche agile, tout en intégrant d'autres équipes telles que les opérations, les testeurs, les designers, les développeurs, les chefs de projet, et en général, tous ceux dont les compétences sont nécessaires pour livrer un produit de qualité. Donc les réunir par exemple aux Daily Meeting !

Automation

L'automatisation est un élément clé du DevOps. Elle permet de réduire les erreurs humaines et d'accélérer les processus de développement et de déploiement. L'automatisation permet également de réduire les coûts et de gagner du temps. Par exemple, l'automatisation des tests permet de réduire le temps de test et de détection des bugs. L'automatisation des déploiements permet de réduire le temps de déploiement et de réduire les erreurs humaines.

Bleu/vert

L'un des principaux principes utilisés dans les pratiques DevOps est le déploiement bleu/vert. Cette approche implique de maintenir deux environnements de serveurs, appelés "bleu" et "vert". Le serveur "bleu" est celui qui répond aux requêtes publiques, jouant ainsi le rôle de l'environnement de

production. Le serveur "vert", quant à lui, est un environnement de test où les modifications sont déployées.

Lorsqu'une nouvelle version est prête à être déployée, elle est d'abord déployée sur le serveur "vert". Une fois que les tests ont été effectués avec succès sur le serveur "vert", les rôles sont inversés. Les requêtes publiques sont alors redirigées vers le serveur "vert", qui devient le nouvel environnement de production.

L'avantage clé de cette méthode de déploiement est la possibilité de revenir rapidement à un état antérieur en cas de problème. Si un dysfonctionnement est détecté sur le serveur "vert" après le déploiement, il suffit de rediriger les requêtes vers le serveur "bleu" pour restaurer l'état précédent de l'environnement de production.

Cette approche de déploiement bleu/vert permet d'assurer une transition fluide entre les versions, en minimisant les interruptions de service et en offrant une stratégie de récupération rapide en cas d'incident.

Créer des environnements à la demande

En utilisant l'approche DevOps, il est possible de **créer des environnements à la demande** en utilisant des outils d'**Infrastructure-as-Code**. Cela permet de réduire considérablement les délais d'attente pour la mise à disposition d' environnements de test et de déploiement, par rapport aux méthodes traditionnelles.

Par exemple, au lieu d'attendre plusieurs jours voire semaines pour obtenir un environnement fourni par l'équipe de production, il est maintenant possible de créer un environnement en quelques dizaines

de minutes, voire quelques minutes seulement. Cela permet de créer facilement et rapidement autant d'environnements que nécessaire pour faciliter les tests et les activités agiles.

Ces environnements sont codifiés et versionnés, ce qui facilite leur reproduction rapide. Pour optimiser les coûts, il est courant de détruire complètement les environnements une fois qu'ils ne sont plus nécessaires. Cela permet d'économiser des ressources et de maintenir un environnement propre et efficace.

tester plus fréquemmment

Les tests peuvent être exécutés plus fréquemment. Bien que l'utilisation de techniques telles que le Test-Driven Development ou le Behavior-Driven Development permette de maximiser les tests unitaires, qui produisent des résultats en quelques minutes grâce à leur rapidité d'exécution, il est souvent nécessaire d'approfondir certains types de tests tels que les tests de performance, les tests de charge ou même les tests end-to-end.

Pour ces types de tests, il est important de disposer d'un environnement correctement configuré afin de simuler les contraintes de production ou de tester l'ensemble des applications de manière intégrée. Étant donné que les tests et l'infrastructure sont versionnés au même niveau que le code, toute modification apportée à l'un de ces trois éléments (tests, infrastructure et code) relance automatiquement l'ensemble du processus. Ainsi, il est facile de rejouer des campagnes complètes de tests de non-régression en quelques minutes, voire quelques heures si la couverture end-to-end concerne une grande partie des fonctionnalités.

Lean

> *Le Lean, ou Lean Management, vient de l'industrie, et plus particulièrement de Toyota dans les années 1990. Lean sert à qualifier une gestion des ressources sans gaspillage.*

Le Lean est une approche de gestion et d'organisation qui vise à maximiser la valeur pour le client tout en minimisant le gaspillage. Il s'agit d'une philosophie qui a été initialement développée par Toyota dans les années 1950 et qui est depuis lors appliquée dans divers domaines, y compris le développement logiciel et l'ingénierie des systèmes, notamment dans le contexte de DevOps.

Le Lean repose sur plusieurs principes fondamentaux, dont les principaux sont :

L'identification de la valeur : il est essentiel de comprendre ce qui est réellement important pour le client. Cela permet de se concentrer sur les activités qui ajoutent de la valeur et d'éliminer celles qui sont considérées comme du gaspillage.

- L'identification du flux de valeur : il s'agit d'analyser les étapes nécessaires pour transformer les inputs en outputs de valeur. L'objectif est d'optimiser ce flux en éliminant les obstacles, les retards et les inefficiences.
- La création de flux continu : cela implique de mettre en place des processus qui permettent de livrer de manière régulière et prévisible. Il s'agit de réduire les temps d'attente et les retards entre les différentes étapes du flux de valeur.
- La recherche de la perfection : le Lean encourage une culture d'amélioration continue. Il s'agit de remettre en question les processus existants, de chercher à les optimiser en permanence et d'impliquer tous les membres de l' équipe dans cette démarche.

Pour illustrer ces principes, prenons l'exemple d'une entreprise de développement logiciel qui adopte le Lean. Supposons que cette entreprise reçoive régulièrement des demandes de fonctionnalités de la part de ses clients. Dans une approche Lean, l'entreprise commencera par identifier la valeur réelle pour ses clients en analysant les demandes et en priorisant celles qui apportent le plus de valeur.

Ensuite, l'entreprise examinera le flux de valeur, c'est-à-dire les différentes étapes nécessaires pour transformer une demande de fonctionnalité en un produit livré. Cela peut inclure des activités telles que l'analyse des besoins, la conception, le développement, les tests et le déploiement. L'entreprise identifiera les étapes qui génèrent du gaspillage, comme les retards, les erreurs ou les reprises, et cherchera à les éliminer ou à les réduire.

En mettant en place un flux continu, l'entreprise s'efforcera de livrer les fonctionnalités de manière régulière et prévisible. Cela peut se faire en utilisant des pratiques de développement Agile, en automatisant les tests et les déploiements, et en favorisant la collaboration et la communication entre les équipes.

Enfin, l'entreprise cherchera constamment à s'améliorer en appliquant une culture d'amélioration continue. Elle collectera des métriques pour évaluer ses performances, analysera les goulots d'étranglement et les problèmes récurrents, et mettra en place des actions correctives pour les résoudre.

En adoptant cette approche Lean, l'entreprise de développement logiciel sera en mesure de répondre plus rapidement aux besoins de ses clients, de réduire les retards et les erreurs, et d'optimiser l'utilisation de ses ressources, ce qui aboutit à une meilleure satisfaction client et à une plus grande efficacité opérationnelle.

Value Stream Map

Le VSM permet de visualiser de manière claire et détaillée l'ensemble des activités et des interactions qui contribuent à la création et à la livraison d'un produit ou d'un service. Il aide à identifier les activités à faible valeur ajoutée, les gaspillages, les retards, les boucles de rétroaction et les opportunités d'amélioration.

> *Par exemple, anciennement la mise en place d'un environnement de test pouvait prendre plusieurs jours, voire plusieurs semaines. Grâce à l'automatisation, il est maintenant possible de créer un environnement en quelques minutes seulement. Cela permet de réduire considérablement les délais et d'optimiser l'utilisation des ressources.*

En Lean, les rétrospectives et les post mortem sont des pratiques importantes pour favoriser l'apprentissage continu et l'amélioration des processus. Elles permettent aux équipes de réfléchir sur leur travail, d'identifier les succès, les problèmes et les opportunités d'amélioration, et de prendre des mesures pour améliorer les performances futures.

Les rétrospectives sont des réunions régulières, généralement organisées à la fin d'une itération ou d'un projet, où l'équipe examine collectivement son travail récent. L'objectif est de faire un bilan des succès, des défis rencontrés et des leçons apprises. Les rétrospectives peuvent se dérouler sous différentes formes, telles que des discussions structurées, des jeux de rôles ou des questionnaires. L'équipe se concentre sur les aspects positifs et négatifs du processus de travail, identifie les problèmes récurrents, propose des solutions et décide des actions à entreprendre pour améliorer les itérations suivantes.

Les post mortem, quant à eux, sont des réunions organisées après la clôture d'un projet ou d'une phase majeure. Ils se concentrent sur l'analyse des réussites, des échecs et des enseignements tirés de l'ensemble du projet. Les post mortem permettent de prendre du recul et d'évaluer les performances globales de l'équipe et du processus. L'objectif est d' identifier les principaux facteurs de succès et les problèmes rencontrés, de documenter les meilleures pratiques et les leçons apprises, et de formuler des recommandations pour les futurs projets.

Dans les deux cas, les rétrospectives et les post mortem encouragent la transparence, la communication ouverte et la collaboration au sein de l'équipe. Ils offrent un espace sécurisé où les membres de l'équipe peuvent exprimer leurs opinions, partager leurs expériences et contribuer à l'amélioration continue. Ces pratiques permettent d'identifier les opportunités de renforcer les points forts, d'atténuer les faiblesses et d'ajuster les processus afin d'obtenir de meilleurs résultats à l'avenir.

Mesure

Les métriques sont des indicateurs quantitatifs qui permettent de mesurer les performances d'un processus ou d'un système. Elles sont utilisées pour évaluer les performances actuelles, identifier les problèmes et les opportunités d'amélioration, et prendre des décisions éclairées.

> *Par exemple, le temps moyen de résolution des incidents est une métrique qui permet de mesurer la rapidité avec laquelle les incidents sont résolus. Si cette métrique est élevée, cela peut indiquer que le processus de gestion des incidents est inefficace et qu'il y a des opportunités d'amélioration.*

Les différents indicateurs utilisés peuvent être :

- Combien de temps la nouvelle fonctionnalité a pris pour passer du développement à la production ?
- Combien de fois un bug récurrent apparaît ?
- Combien de personnes utilisent le produit en temps réel ?
- Combien d'utilisateurs a-t-on gagnés ou perdus en une semaine ?

Sharing

Le fameux mur de confusions est largement dû à un manque de communication entre les équipes. Il est donc important de partager les connaissances et les informations entre les équipes.

Le partage des connaissances est une pratique importante pour favoriser l'apprentissage continu et l'amélioration des processus. Il permet aux membres de l'équipe de partager leurs expériences, leurs idées et leurs connaissances, et de bénéficier des connaissances des autres.

Pattern et anti-pattern

> *Un pattern est une solution à un problème dans un contexte donné et un anti-pattern est une solution qui semble intuitive mais qui est en fait une mauvaise pratique.*

Anti-pattern

Silo Dev & Silo Ops

Le premier anti-pattern que le DevOps cherche à éliminer est la séparartion entre les équipes de développement et d'ops, ce fameux mûr de confusion.

> *D'un côté, nous avons des développeurs dont le métier est de développer des user stories, de faire évoluer l'application, d'implémenter de nouvelles features et de corriger les bugs. Et de l'autre, nous avons des ops dont le métier est de maintenir en condition opérationnelle l'application, de réparer les serveurs qui ne fonctionnent pas, d'appliquer des patchs de sécurité et de faire évoluer les produits.*

Ces deux équipes ont donc deux vitesses différentes : une équipe voulant aller vite et avoir le maximum de changements, et une autre équipe voulant garantir la stabilité de l'application. De plus, généralement, ces équipes sont motivées par le nombre de bugs corrigés ou de nouvelles features produites, pour les développeurs ; et par le nombre de bugs critiques en production, pour les ops.

NoOps

Le deuxième anti-pattern est le NoOps. Il s'agit de l'automatisation de toutes les tâches d'ops. Cela peut être intéressant pour les petites structures, mais cela peut être dangereux pour les grandes structures.

> *En effet, si les développeurs sont responsables de la production, ils peuvent être tentés de faire des changements rapides et de ne pas prendre en compte les problèmes de stabilité. Cela peut conduire à des problèmes de sécurité, de performance et de stabilité.*

Ops dans l'équipe de développement

Le troisième anti-pattern est d'avoir des ops dans l'équipe de développement. Cela montre que l'organisation ne prend pas conscience de l'importance des compétences requises pour maintenir une application en production.

Pattern

Collaboration entre Dev et Ops

La clé pour la réussite de l'implémentaion du DevOps est la collaboration entre les équipes de développement et d'ops.

> *Les développeurs doivent comprendre les problèmes d'ops et les ops doivent comprendre les problèmes de développement. A l'occasion de Daily Meeting, pour que les Ops soient au courant des changements et des problèmes de développement qui adviendront au cours du sprint, et inversement.*

Dev et équipe cloud

Pour les organisations disposant d'un département IT assez traditionnel qui ne peut pas ou ne veut pas changer (assez) rapidement, et pour les organisations qui exécutent toutes leurs applications dans le cloud public (Amazon EC2, Azure, etc.), il est probablement utile de traiter les opérations comme une équipe qui fournit simplement l'infrastructure sur laquelle les applications sont déployées et fonctionnent. Cela peut-être une équipe de DevOps virtuelle.

Service externe DevOps - DevOps as a Service

Pour les organisations qui ne disposent pas de ressources internes pour mettre en œuvre le DevOps, il existe des services externes qui peuvent aider à mettre en œuvre le DevOps. Ces services sont appelés DevOps as a Service.

Equipe DevOps temporaires

Cette équipe temporaire a pour objectif de rapprocher les développeurs et les opérations, idéalement vers un modèle de collaboration entre dev et ops, et éventuellement de devenir obsolète.

Les membres de l'équipe temporaire abordent les problématiques spécifiques de chaque équipe :

Ils apportent des idées telles que les réunions quotidiennes (stand-up meetings) et le Kanban pour améliorer les pratiques des équipes Ops.

Ils se concentrent sur des détails tels que les équilibreurs de charge (load-balancers), la gestion des cartes réseau et le SSL pour soutenir les efforts des équipes Dev.

Équipe d'évangélistes DevOps

Egalement connue sous le nom de coachs DevOps, est un pattern efficace que est expérimenté et mis en place avec succès dans plusieurs entreprises. En tant que coach DevOps, mon rôle consistait à aider les équipes de développement et d'opérations à mettre en œuvre le DevOps, ainsi que les pratiques d'intégration et de livraison continues.

Mis en pratique avec des activité comme le DevOps où chacun se met en situation de l'autre, ou encore le DevOps Dojo où les équipes de développement et d'opérations travaillent ensemble pour

résoudre des problèmes réels.

DevOps axée sur les conteneurs

Une des promesses des conteneurs est la portabilité des applications ! Ils s'exécutent de la même manière sur n'importe quel système d'exploitation et sur n'importe quel cloud. Cela permet aux développeurs de se concentrer sur le développement de l'application et aux opérations de se concentrer sur la gestion de l'infrastructure de ce conteneur.

> *D'un coté, les développeurs ne se chargent que de créer l'image, afin que l'application puisse tourner correctement. De l'autre, les ops opèrent cette image avec toutes les contraintes liées au maintien en condition opérationnelle.*

Les quatres signaux dorés

Ce sont quatre signaux à observer constamment pour grandir une haute disponibilité des services.

Latence

La latence représente le délai entre l'envoi d'une demande et la réception de la réponse. Elle est généralement mesurée du côté du serveur, mais peut également être mesurée du côté du client pour prendre en compte les variations de vitesse du réseau. L'équipe d'exploitation est principalement responsable de la gestion de la latence côté serveur, tandis que la latence côté client est plus pertinente pour les utilisateurs finaux.

Le seuil de latence cible peut varier en fonction du type d'application. Les systèmes automatisés tels que les API ou les serveurs peuvent nécessiter des temps de réponse beaucoup plus rapides que ceux attendus par un utilisateur sur un téléphone mobile. Il est également important de suivre séparément la latence des requêtes réussies et des requêtes échouées, car les requêtes échouées ont tendance à échouer rapidement sans nécessiter de traitement supplémentaire.

Traffic

Le trafic représente le nombre de demandes circulant sur le réseau, telles que les requêtes HTTP vers un serveur web ou une API, ou les messages envoyés à une file d'attente de traitement. Les périodes de pointe de trafic peuvent exercer une pression supplémentaire sur votre infrastructure, la poussant à ses limites et entraînant des effets indésirables.

Le trafic est un indicateur clé car il permet de distinguer les problèmes de capacité des configurations système inappropriées, qui peuvent causer des problèmes même pendant les périodes de trafic faible. Dans le cas de systèmes distribués, il peut également aider à la planification de la capacité en prévision de la demande future.

Erreur

Les erreurs fournissent des informations sur les erreurs de configuration de votre infrastructure, les bugs présents dans votre code ou les dépendances non résolues. Par exemple, un taux d'erreur élevé peut indiquer une défaillance de base de données ou une panne de réseau.

Après un déploiement de code, les erreurs peuvent révéler des bugs qui ont survécu aux tests ou qui n'ont été détectés qu'une fois en production. Les messages d'erreur fournissent des informations supplémentaires sur la nature précise du problème. Les erreurs peuvent également avoir un impact sur d'autres métriques, en faussant artificiellement la latence ou en générant des tentatives répétées qui finissent par saturer d'autres systèmes distribués.

Saturation

La saturation se réfère à la charge exercée sur votre réseau et les ressources de votre serveur. Chaque ressource a une limite au-delà de laquelle les performances se détériorent ou deviennent indisponibles. Cela s'applique aux ressources telles que l'utilisation du CPU, de la mémoire, de l'espace de stockage et des opérations par seconde. Il est essentiel de comprendre la conception et les spécificités de votre système distribué pour identifier les parties qui pourraient se saturer en premier. Ces mesures sont souvent des indicateurs avancés, permettant d'ajuster la capacité avant que les performances ne se dégradent.

L'atteinte de la limite de saturation peut avoir plusieurs effets sur votre service. Par exemple, un CPU saturé peut entraîner des retards dans les réponses, un espace de stockage plein peut provoquer des échecs d'écriture sur le disque, et une saturation du réseau peut entraîner une perte de paquets.

Références & sources

- OpenClassrooms
- DevOps - Wikipedia

Last updated on June 18, 2023

‹ La veille informatique

Design Patterns ›

Portefolio Dasek Joiakim