Hewlett Packard Enterprise

EN PARTENARIAT AVEC ZDNet.fr

Harmoniser les relations de travail entre les équipes de développeurs et celles de production est l'un des leviers les plus efficaces pour aller plus loin dans la performance de la DSI. Industrialisation des releases, traçabilité, méthodes et outils ; le point sur les ce qu'il faut faire pour aller de l'avant.

Sommaire

DevOps : une nouveauté ? Réellement la panacée ?

Automatisation, collaboration, cycles continus… l'approche DevOps bouscule tout sur son passage. Pourtant, d'autres approches de ce type l'ont précédée. Qu'apporte cette nouvelle démarche ? Tout est-il vraiment rose et merveilleux en pays DevOps ?

La démarche DevOps suscite un intérêt croissant dans les entreprises. Tout d'abord elle permet d'améliorer les relations entre les équipes système, de production, et des études/développement… et surtout avec les utilisateurs.

Toutefois, cette approche concrétise l'aboutissement de plusieurs approches tout en bénéficiant de multiples innovations technologiques facilitant son émergence. Ce qui n'empêche pas les difficultés d'implémentation ou la résistance au changement.

Un pas de plus vers la concrétisation du rêve

Les spécialistes du développement ont suivi les évolutions majeures comme les ateliers de génie logiciel, les langages de quatrième génération ou les démarches itératives de type RAD, entre autres. Autant d'approches qui ont mis en avant la collaboration dès l'amont du projet avec l'utilisateur final, et rapproché un peu les équipes informatiques. Certains de ces outils restent d'ailleurs parfois d'actualité.

Cependant, on constate des différences majeures que vient combler le DevOps. Ainsi, l'un des atouts majeurs consiste à mettre en place un processus global (avant, pendant et après le projet) impliquant tous les intéressés avec des rôles clairement définis (idéalement avec l'accord de tous). Ainsi, les tests ou les recettes font partie du processus et ne sont pas des tâches annexes ou connexes. Les outils DevOps (libres ou commerciaux) proposent de nombreux moyens de collaboration et de gestion de référentiels communs intégrant tous les aspects du projet, avec un suivi et des mécanismes d'automatisation.

L'automatisation favorise cycles continus et collaboration

Parmi les technologies favorisant l'apparition du DevOps, on retrouve : la virtualisation, l'automatisation, la standardisation des référentiels de code, ou encore les plateformes API. Ces plateformes (ou leurs équivalents) permettent une réutilisation optimale d'un code optimisé et partagé par tous les projets, voire même par des partenaires ou clients (avec la sécurité adaptée). Là aussi, les anciens se souviendront des Web Services, visant le même objectif.

Cette organisation favorise le développement en continu, car "il suffit" de recueillir les demandes et besoins des utilisateurs pour modifier le code, sans forcément interrompre l'application. A la manière des services sur Internet.

L'automatisation décharge les opérations, mais aussi les développeurs ou les testeurs de tâches automatisables, pour lesquelles leur valeur ajoutée est quasiment nulle. Ils peuvent alors e consacrer à améliorer les fonctions existantes ou assurer un suivi de qualité par exemple. Une automatisation qui améliore non seulement la productivité, mais aussi la qualité de l'application puisqu'elle réduit les risques d'erreurs de manipulation.

Autre atout intéressant, l'utilisateur final peut interagir avec l'application en direct dans un environnement virtuel simulant son espace de travail et (et même avec des données réelles, voire en temps réel). Chaque remarque ou demande est alors enregistrée. Une itération permet alors de modifier l'application et de la soumettre à nouveau à l'utilisateur. Un processus qui se répétera jusqu'à l'application finale. Puis, après les derniers tests, l'application passera très simplement en production (environnement qu'elle utilise en simulé depuis plusieurs semaines). Fini l'angoisse des manques ou erreurs qui devront attendre plusieurs mois pour réécrire une version 2.0. Désormais, des revues hebdomadaires permettent d'intégrer des évolutions/modification en continu, à tout moment. Sans négliger pour autant les phases de tests ou de recettes, devenues beaucoup plus simples et rapides.

Attention aux chocs avec la réalité du terrain

Malheureusement, tout n'allant pas toujours pour le mieux dans le meilleur des mondes, l'existant limite fortement les bénéfices d'une telle approche. Comment faire travailler les équipes de cette façon sur un projet isolé en les laissant en partie sur des projets traditionnels ? En sachant qu'il faut pouvoir les monopoliser à tout moment. Quel est l'intérêt de cette démarche sans une automatisation optimale de la gestion des ressources informatiques ? Le parc existant le permet-il ?

Et comment sensibiliser les informaticiens à l'absolue nécessité d'évoluer vers des profils différents impliquant une évolution de leurs tâches et responsabilités ? N'en doutons pas, l'avenir du développement repose sur les plateformes API, leur gestion, leur sécurisation et une forte automatisation. Pour lesquels une bonne connaissance des technologies déjà en place reste, elle aussi, indispensable.

Autant de questions que tentent de résoudre chaque jour les pionniers de ces nouvelles approches depuis 5 à 10 ans. Avec le succès croissant de DevOps, espérons que de bonnes pratiques, prenant aussi en compte l'existant organisationnel, finiront par s'imposer.

Attention à ne pas confondre DevOps et Dev + Ops

Rapprocher développement et production ne va pas de soi. Chacun doit comprendre l'autre et collaborer pour accélérer la mise en production. Indispensable à l'entreprise agile, DevOps repose essentiellement sur des facteurs humains.

Pour que l'informatique améliore l'efficacité des activités de l'entreprise, encore faut-il que les personnes participant au processus de conception des applications collaborent intelligemment en ce sens. Sinon, comment s'assurer que tout sera mis en œuvre pour concrétiser la stratégie de l'entreprise ?

Sortir des silos conflictuels internes à la DSI

Certes, la réussite passe sans aucun doute par une écoute et une compréhension des utilisateurs métiers. Cependant, comment y parvenir si tous les informaticiens ne perçoivent pas la même chose, et travaillent chacun dans leur coin sans communiquer, générant des incompréhensions et des délais inacceptables pour l'entreprise agile ?

Dans l'informatique traditionnelle (encore très majoritaire), le développement d'une application d'entreprise et sa mise en production sont considérés comme deux projets distincts, et parfois même comme trois projets (avec les tests entre les deux).

La démarche DevOps vise à résoudre ce problème majeur, en harmonisant les relations entre le développement et la production (et les tests/qualité, évidemment). Défi de taille quand les développeurs souhaitent avant innover et faire évoluer les applications, tandis que les Ops cherchent à maintenir la stabilité du système informatique. D'ailleurs, chacun suit ses processus, et travaille avec ses propres outils, rarement communicants. Résultats : les relations entre ses équipes deviennent souvent conflictuelles. Et les retards, catastrophiques…

Une autre vision et une culture différente, partagées

Avec le DevOps, la communication entre ces informaticiens est organisée afin de satisfaire réellement les besoins des utilisateurs métiers, de réduire au minimum les délais de déploiement et de modification, et de déployer au mieux la stratégie opérationnelle de l'entreprise.

Outre le rapprochement pour accomplir leurs tâches autrement, les équipes de développement et de production doivent partager non seulement des informations, mais surtout une vision et une culture communes de collaboration et via des processus très automatisés.

Un fort changement culturel s'impose, avec une forte volonté affichée par la DSI et la direction générale de l'entreprise. En effet, modifier les processus et les modes de travail ne va jamais de soi... Car, le déploiement de divers outils, aussi pertinents soient-ils, ne sert à rien si les équipes informatiques freinent des quatre fers à l'idée de se répartir différemment les tâches et les rôles.

Des équipes mixtes deviennent indispensables, avec des personnes à même de communiquer au maximum : des équipes DevOps, et non pas Dev +Ops.

De nouveaux repères pour mieux voir l'ensemble

L'évolution culturelle sévère d'autant plus difficile que chacun doit faire évoluer ses repères. Ainsi, il ne s'agit plus de déterminer uniquement qui va désormais assumer telle ou telle tâche, mais de faire en sorte qu'elle sera assurée de façon optimale pour parvenir au plus vite et de façon optimale au résultat. Les objectifs des deux camps doivent être alignés, les outils partagés ainsi que les processus et les métriques.

On constate la difficulté de cette évolution, y compris dans la manière dont chacun perçoit les bénéfices et avantages du DevOps. Ainsi, le développeur mettra en avant l'intégration continue, la simplicité de maintenance du code ou l'automatisation des tests.

Un ingénieur de production IT plaidera en faveur l'automatisation des déploiements d'infrastructure et l'optimisation des environnements facilitée par la mesure et le décisionnel en temps réel. Et tout cela est absolument vrai. Cependant, chacun raisonne encore en fonction des frontières traditionnellement préétablies au vu de leurs attributions respectives.

Les projets DevOps réussis montrent une implication réelle entre production et développement, où chacun connait les outils et les modes de travail de l'autre, ses contraintes et les ponts indispensables entre les deux disciplines à chaque étape. Faire tomber ce type de cloisons entre les individus supprime bien des ralentissements onéreux, et sources de mécontentement auprès des utilisateurs métier.

L'approche Devops ne consiste pas uniquement à améliorer l'efficacité de chaque silo, mais bien à créer une synergie globale visant à mieux servir le client final, en s'appuyant sur une communication constructive. Et l'engagement de chacun n'est plus délimité par une frontière relative à une fiche de poste.

Pourquoi les entreprises adoptent-elles une approche DevOps ?

Evolution réglementaire ou des marchés, concurrence, besoin d'agilité, transformation digitale... Les raisons invoquées par les adeptes touchent aux problématiques les plus répandues. Toute ressemblance avec votre entreprise ne serait que pure coïncidence...

Pourquoi l'entreprise nécessite-t-elle une informatique agile, à même de proposer des applications qui s'adaptent au plus vite aux évolutions de ses métiers ? Finalement, on est en droit de se poser la question.

Mettre un frein final à la course à l'échalote

Avec la conception d'application selon des cycles en V (ou la lettre de votre choix), il s'agit pour la DSI de remettre sur l'ouvrage toute l'application, après une étude fonctionnelle poussée. L'étape suivante consiste à concevoir le nouveau code afin de générer une application (généralement très modifiée, parfois totalement refondue). Après tous les tests de ce nouveau code, le déploiement intervient entre six et 24 mois après la définition des besoins par les utilisateurs finaux. Résultats : les directions opérationnelles disposent enfin de l'application répondant bien aux besoins... qu'elles ont exprimés plusieurs mois auparavant. Quelques semaines après, on repart dans un cycle de plusieurs mois... Bref : le temps de mettre l'application à disposition, les besoins ont généralement évolué !

D'où la nécessité d'une méthode permettant, dans un premier temps, de concevoir et de déployer rapidement une application de qualité, puis de faire en sorte que des modifications plus ou moins importantes puissent être disponibles en quelques heures ou quelques jours. Cette différenciation ne justifie-t-elle pas à elle seule l'intérêt pour le DevOps ? Le déploiement rapide de la stratégie de l'entreprise et de ses évolutions n'incarne-t-il pas la méthode applicative idéale ? Certes, il faut prendre en compte le terrain psychologique des équipes de développement et de production, ainsi que le besoin de faire évoluer leurs modes de travail et leurs habitudes. Néanmoins, si l'informatique pouvait arrêter de courir après l'organisation sans jamais la rattraper...

Transformation numérique, time-to-market et agilité

Une étude IDC/Automic sur le DevOps a été réalisée au second semestre 2015 auprès de 201 entreprises françaises. Parmi les résultats, on constate que pour 52% des entreprises interrogées, « la transformation numérique » est la principale raison invoquée pour l'adoption d'une approche DevOps. Rien d'étonnant lorsqu'on sait que ces méthodes font les jours heureux des géants du Web, et que les entreprises ont pris conscience de la nécessité de passer à l'ère numérique (ou digitale).

Par ailleurs, 44% de ces entreprises avaient déjà initié une approche DevOps ou l'avaient planifié. Et deux tiers d'entre elles étaient motivés par des enjeux de digitalisation incontournable pour leurs activités. L'étude détaille ainsi que 21% des sondées mettent en avant le contexte de leur marché, 21% le time-to-market, et 19% le besoin en développement agile. Le « contexte de leur marché » englobe évidemment les contraintes réglementaires qui évoluent à rythme effréné. Or, pour les intégrer, l'approche traditionnelle s'avère inappropriée, puisqu'une mise à jour continue des règles impactant souvent plusieurs processus s'impose désormais.

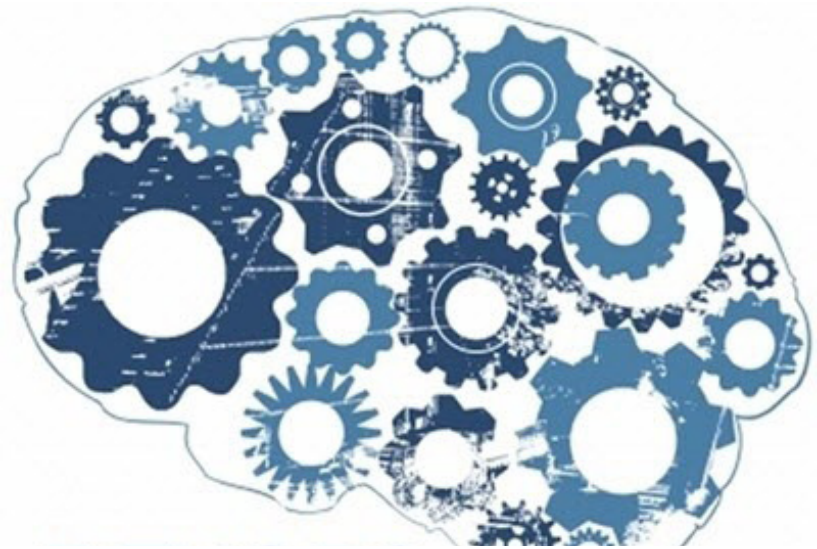
Non seulement la démarche DevOps accélère la livraison des applications et (dans son stade avancé) la mise place des modifications en continu, mais en plus, elle offre une visibilité des processus et des transactions évoluées ainsi qu'une traçabilité des actions et des opérations. Encore un avantage au service des audits de plus en plus exigeants sur tous ces aspects.

C'est donc sans surprise que l'étude IDC révèle que 51% des grandes entreprises avaient déployé des méthodes agiles (fin 2015), et que 65% d'entre elles s'étaient alors tournée vers le DevOps. 71% des entreprises concernées témoignent d'une meilleure collaboration avec les métiers. Enfin, dans ces projets DevOps, 70% des indicateurs partagés le sont avec les directions opérationnelles.

Autre avantage du DevOps, une forte automatisation oblige l'entreprise à réfléchir sur les processus et les rôles de chacun. Alors, une bonne gouvernance devient indispensable. Or justement, de plus en plus d'entreprises adoptent des solutions de type ITMS pour maîtriser les processus, automatiser, redéfinir précisément les rôles. Tout cela n'est-il pas finalement complémentaire ?

Livraison et déploiement continus : bénéfices et contraintes

Parmi les bénéfices du DevOps, l'industrialisation de la livraison de code et l'automatisation du déploiement arrivent en bonne place. Deux notions à ne pas confondre, imposant aussi des contraintes. Les anciennes approches feraient-elles mieux ?



L'approche DevOps vise à automatiser et optimiser la mise à disposition d'infrastructures afin de réduire la livraison d'applications et leur déploiement. Mais, à quoi bon mettre en place du développement agile si le cycle de vie de l'application ne suit pas le mouvement ?

Pour répondre au plus vite et au mieux aux attentes des métiers, architectes, développeurs, testeurs, services informatiques opérationnels et équipes de production informatiques doivent mieux communiquer et collaborer. De plus en plus, l'utilisateur exige de disposer d'améliorations opérationnelles en deux ou trois semaines... C'est pourquoi une logique de mises à jour et de correctifs incrémentiels devient incontournable. Donc une industrialisation optimale tout au long du cycle de vie applicatif.

Distinguer livraison et déploiement continus

La livraison continue d'application ou de modifications (continuous delivery) et le déploiement continu (continuous deployment) constituent deux des piliers de la démarche DevOps. Et doivent évidemment s'articuler autour d'un retour continu des utilisateurs finaux. Néanmoins, il convient de distinguer ces deux notions.

Avec une méthode de Continuous Delivery, les développeurs produisent du code applicatif sur des cycles courts, en s'assurant de pouvoir le modifier ou le mettre à jour à tout moment (idéalement, sans arrêter la production). En optimisant ou automatisant le développement, les tests et la mise à disposition du code, les équipes réduisent fortement les délais et les coûts, tout en procurant à l'entreprise une informatique agile, réactive, voire proactive. Cependant, si le code à disposition est déployable en un ou deux clics, il n'est pas forcément automatiquement déployé.

En revanche, dans le cadre du déploiement continu, chaque modification est automatiquement déployée si les phases de tests (automatisés ou non) sont positives. Le déploiement continu nécessite donc un niveau supérieur de maturité et de maîtrise.

Des gains bien au-delà de la productivité

Outre les gains de temps liés à l'automatisation de la plupart des étapes, la livraison continue apporte des avantages en lisibilité et en traçabilité. En effet, elle est souvent liée à l'utilisation d'une plateforme logicielle ou applicative de nouvelle génération (type plateforme API). Le développeur réutilise du code éprouvé ou fait appel à des API externes, elles aussi éprouvées. Par conséquent, les risques d'erreurs sont réduits et la productivité fortement améliorée. De plus, tout étant tracé et enregistré, il devient plus facile de repérer des erreurs ou de remonter la chaîne des incidents. Pour peu que l'entreprise ne néglige pas les phases de tests et de qualité.

Tout est lié, tout est impacté. Mais...

Néanmoins, il faut toujours garder à l'esprit les contraintes de ce type d'approche. Outre la conduite du changement des informaticiens (pas toujours évidente), la DSI ne devra pas négliger les considérations d'architectures applicatives. Certes, les nouvelles fonctionnalités peuvent plus vite être déployées, et tout le processus devient plus fluide. Cependant, revenir en arrière ou apporter une modification majeure impose parfois de corriger le code initial et de remettre en question l'existant. Dans ce cas, il faudra reprendre le processus pour remplacer de grandes parties de code. Toutefois, cela est-il plus traumatisant que la rupture souvent imposée par les nouvelles versions majeures d'ERP ?

En abattant les murs séparant les équipes de développement, de tests/qualité, et de production, le DevOps oblige à insuffler une culture résolument différente, y compris par une remise en cause de tout ce qui a été la règle jusqu'à un moment donné.

Or, une remise en cause partielle peut avoir des conséquences importantes. Ainsi, la révision d'un processus de tests automatisé (lié par exemple à une nouvelle règle de conformité réglementaire) peut obliger à modifier du code déjà en production. Et, s'il s'agit d'une API externe à l'entreprise, tous les impacts devront être bien identifiés. Néanmoins, cela serait-il plus simple avec une approche de développement traditionnelle ? En effet, avec ces nouvelles plateformes, tout est lisible, documenté et tracé.

On constate avec intérêt que les arguments généralement avancés par des informaticiens "conservateurs" pour critiquer l'approche DevOps font appel à des situations que leurs propres méthodes de travail n'ont pas su et ne savent toujours pas résoudre...

Au-delà de l'automatisation, le moteur de règles devient acteur du DevOps

L'automatisation repose sur un moteur de règles, et non plus sur des scripts plus ou moins basiques. Il va devenir de plus en plus contextuel et intelligent. Capable de détecter des anomalies et même proposer des recommandations.

A un stade avancé de maturité DevOps, l'entreprise devrait recourir fortement à l'automatisation. Il est vrai que depuis des années les équipes de développement, de production et même de tests utilisent des scripts permettant de se délester de certaines tâches sans valeur ajoutée.

Toutefois, tous ces fichiers répartis sont rarement cohérents dans un ensemble, et relèvent plus d'astuces individuelles que d'une démarche de projet. En outre, leur suivi, la gestion de leurs évolutions et leur maintenance dépendent souvent du bon vouloir de ceux qui les utilisent.Parfois, certains développeurs savent même comment contourner certaines mesures automatisées qui ne leur conviennent pas.

Au cœur du référentiel DevOps, l'automatisation repose sur un moteur de règles (ou un ensemble d'algorithmes) utilisant des paramètres partagés par tous.

Un moteur de règle intelligent

Un tel moteur apporte la cohérence et au système d'information et chacun accède ainsi au même référentiel, avec un langage commun, donc compréhensible par tous les acteurs du projet. Parce que le DevOps efficace se doit de mesurer et de tracer toute action, le recours à des outils d'analyse des données permet de déployer une gouvernance en temps réel. D'où l'adoption de plus en plus courante d'outil reposant sur Hadoop pour intégrer le Big Data.

En effet, les mesures et la traçabilité de l'ensemble des actions et opérations génèrent rapidement de gros volumes. La proximité de ce moteur avec le référentiel et sa position au cœur du projet peuvent même étendre son rôle au-delà de la simple vérification et du déclenchement d'actions programmées.

Une dimension contextuelle appréciable

Un projet de développement applicatif est un terrain mouvant, en constante évolution. C'est pourquoi l'adaptabilité devient vite indispensable. Le moteur doit donc être en mesure d'assurer la cohérence continue des règles qu'il gère et des opérations déclenchées. Un peu plus d'intelligence est donc la bienvenue.

Avec les outils d'analyse de données, le moteur de règles devient plus intelligent et réactif. Il peut de lui-même alerter sur des modèles de comportement anormaux, ou le dépassement de seuils prédéfinis. Une caractéristique intéressante puisque le moteur de règles contextuel peut même prévenir d'une incohérence avant déploiement. Cela favorise la qualité du projet et simplifie le travail de toute l'équipe (développement, tests, déploiement, production…)

Du proactif jusqu'à la décision d'agir

Les algorithmes prédictifs, utilisant les analyses et schémas du projet, apportent au moteur de règles une dimension proactive. Ainsi, des statistiques corrélées à des éléments contextuels lui permettent d'anticiper des situations dont les caractéristiques engendrent généralement des incidents ou des pannes. Bien entendu, la pertinence de ces analyses augmente avec le temps. Les schémas détectés peuvent concerner les applications ou fonctions, mais aussi les utilisateurs, les développeurs, etc.

Associé avec des outils de machine learning, le moteur peut apprendre et détecter lui-même des schémas comportementaux. Cette « connaissance » alimentée chaque jour augmente la pertinence des résultats.

Provenant des travaux sur l'intelligence artificielle, le machine learning dote un programme de la capacité d'autoapprentissage. En bonne connaissance du contexte, et correctement paramétré, ce moteur pourra même parvenir à proposer des solutions pour résoudre un problème détecté, ou mettre en avant de bonnes pratiques efficaces.

Et pourquoi ne pas imaginer qu'il puisse un jour déployer lui-même la solution apparemment la plus appropriée ?

Traçabilité et mesures au cœur du moteur DevOps

La mise en place d'un outil de DevOps permet une meilleure traçabilité des actions entre les équipes. Quels sont les avantages induits par cette traçabilité ? Y a-t-il un besoin de gouvernance et de redéfinition des rôles ou de création de nouveaux profils ?

Parce qu'il vise aussi à faire disparaître les murs entre développeurs et informaticiens des équipes opérationnelles, l'approche DevOps amène nécessairement à une redéfinition précise des rôles. Or, lorsqu'il s'agit d'attribuer différemment les tâches dans un contexte où de nombreuses opérations sont automatisées, la gouvernance globale devient complexe

C'est pourquoi la mesure et la traçabilité s'imposent, afin d'éviter le syndrome de la Tour de Babel (où chacun parle sa langue et voit ses propres référents). Sinon, cela revient à naviguer sur une mer incertaine sans boussole ni GPS

Tracer pour optimiser

Plutôt que de désigner des coupables, la traçabilité a pour objectif de trouver des explications en remontant la chaine opérationnelle. Souvent, les raisons qui ont entraîné ce qui a généré un incident posent une question à laquelle il faudra bien répondre. Or, l'auteur du code ou de l'action a nécessairement des explications intéressantes à étudier.

Et si cette situation résulte d'un manque de compréhension d'un aspect du projet, il n'est certainement pas le seul concerné. D'où l'intérêt de partager aussi ces constatations avec toute l'équipe. Au-delà de la constatation, cette démarche favorise aussi les opérations de régressions suite à des modifications du code, entre autres.

Automatiser n'est pas jouer

En situation d'automatisation, les mesures relevées en temps réel et la traçabilité des actions simplifient la recherche de l'origine des erreurs. En effet, la visibilité globale de l'application devient optimale si chaque action est mesurée et divers paramètres surveillés.

Une démarche en cycles répétitifs (développement, tests, mise en production, retours des utilisateurs / développement, tests...), de mauvaises performances ou des erreurs se reproduisent immanquablement à chaque cycle. Elles peuvent même gagner progressivement en nuisance.

Et cela vaut également pour les processus automatisés, auxquels on pense parfois que l'on peut faire confiance, aveuglément. La traçabilité et les mesures favorisent la gouvernance du projet dans une optique d'amélioration continue et de qualité. Cette amélioration ne doit pas se contenter d'intégrer les remarques des utilisateurs, mais devrait aussi consister à optimiser l'infrastructure et le code. D'où la nécessité impérieuse de faire disparaître aussi les murs entre testeurs et production, dans une démarche de dialogue et de partage.

De nouveaux profils ?

Avec le DevOps, l'entreprise sera certainement amenée à étendre les rôles des informaticiens, ou à créer de nouveaux profils. Par exemple, ne serait-il pas pertinent de confier une partie du lancement ou de la supervision des tests (plus ou moins automatisés) au développeur du code ? Une spécialisation plutôt logique qui ne devrait pas déplaire à un programmeur avisé. Dans certains cas, une partie des tests ne relèverait-elle pas d'un membre de l'équipe d'exploitation ou de production ?

Dans le même sens, serait-il pertinent (ou non) de confier une partie de tests fonctionnels à un spécialiste métier formé à ces aspects ? Une sorte de « correspondant informatique » rattaché à une direction opérationnelle dont il connait bien le métier. Cet « ambassadeur » devra également faire preuve de diplomatie et être formé à la gestion de conflit, afin de les anticiper (pas forcément de les éviter). S'il faut parfois que les choses soient dites, tout tient dans la manière...

Les projets DevOps passent au contrôle continu. Sans examen final ?

Que deviendrait la belle promesse de l'approche DevOps et de son "continuous delivery", sans une gestion intégrée des retours utilisateurs (ou clients) dans un cycle de contrôle continu débouchant sur une amélioration permanente ?

Ce contrôle continu (ou Continuous Assessment) incarne l'une des règles du jeu initiales de toute démarche DevOps. En effet, un tel cycle de vie de développement doit s'articuler autour des retours utilisateurs animant les modifications et améliorations. Une application a-t-elle réellement du sens si l'utilisateur pour qui elle est conçue ne donne pas son avis ?

En intégrant les besoins et attentes de l'utilisateur et ses impressions, cette caractéristique fondatrice de la démarche intervient à chaque étape. A quoi bon reboucler sans cesse, si l'on s'aperçoit au final que cela est inutile ou problématique pour l'utilisateur ?

Continuous Assessment... Continuous quoi ?

Toutefois, le Continuous Assessment nécessite une plateforme de développement agile, flexible et élastique. Et surtout une collaboration sans faille dès l'origine du projet entre développeurs, équipes de production et testeurs (au cœur de DevOps). Ou du moins entre ces tâches, quelle que soit la personne qui s'en charge (fussent-elles automatisées).

DevOps ou OpsDev, qu'importe ?

Avec DevOps, les profils (développeurs, testeurs, ops...) évoluent et les responsabilités deviennent mouvantes. En attendant, une collaboration en bonne intelligence et une compréhension globale du projet et des rôles de chacun contribuent indéniablement au succès. La parole libre est instituée comme une règle, sans culpabilisation des personnes et de leur travail, mais dans un souci continu de qualité et d'efficacité. Chacun doit laisser sa susceptibilité mal placée au vestiaire ! D'ailleurs, mieux vaut tout se dire avant, plutôt que de l'analyser après échec. Autant de précautions, avec le recours à un référentiel intelligemment conçu, favoriseront une meilleure cohérence du projet.

On parle volontiers de DevOps qui laisserait supposer que les développeurs doivent collaborer avec les Ops. Certains évoquent aussi la notion d'OspDev, insinuant que le développeur est aussi le client des équipes d'exploitation et de production, et plus généralement de l'infrastructure. A quoi bon opposer ces deux fonctions ? Ne sont-elles pas appelées à coopérer désormais de façon encore plus étroite en prenant en compte les retours des uns des autres dans une relation d'interdépendance ?

Certes, une évolution des mentalités s'impose. En cas de soucis l'entreprise peut aussi tenter d'inverser les rôles entre Dev et Ops (et/ou testeurs) pendant une ou deux journées, afin d'inciter à la prise de conscience de l'absolue nécessité de coopérer. Vivre les contraintes de l'autre, n'est-ce pas déjà commencer à le comprendre ?

De la mesure dans toute chose.

Pour éviter d'avancer à l'aveuglette, il convient de mesurer et de superviser tous les composants de l'application. On ne comprend et l'on ne peut expliquer raisonnablement que ce que l'on peut mesurer concrètement.

Idéalement, l'équipe DevOps devrait même pouvoir simuler le fonctionnement des modifications avant de les tester, puis de les déployer. Avec ces mesures, l'entreprise dispose en outre d'une solution pour détecter les fonctions ou opérations devenues inutiles. Résultant soit d'une expression erronée d'un utilisateur, soit d'un besoin ayant évolué. Quoi qu'il en soit, cette visibilité s'avère plus qu'intéressante dans une démarche d'une amélioration continue !

Evidemment, la mesure de l'impact sur les activités de l'entreprise devient un atout indéniable pour faire ou ne pas faire. Et surtout : pour justifier tout investissement.

HPE ALM Octane : quand l'ALM devient compatible DevOps

Quel intérêt pourrait avoir une nouvelle démarche de développement purement descriptive, sans outillage adapté ? Mais, les solutions ALM reposent souvent sur des méthodes classiques. HPE ALM Octane innove et assure la continuité.

Sans les outils qui ont contribué au succès du DevOps, cette approche serait certainement venu enrichir les piles de recueils méthodologiques hantant les fonds de tiroirs (ou haut des placards) des DSI...

DevOps vise à accélérer la réalisation des projets tout en améliorant leur qualité, leur maîtrise pour concrétiser fidèlement la stratégie de l'entreprise, le tout inspiré par l'utilisateur. Comment y parvenir sans logiciel de gestion du cycle de vie des applications ou ALM (Application Lifecycle Management) ? C'est pourquoi l'ALM doit s'adapter pour répondre à cette nouvelle approche flexible et itérative, brisant avec les anciens schémas. Désormais, les méthodes de développement sont Agiles, Lean et DevOps.

Le couteau suisse DevOps ouvert à tous

Avec sa solution ALM Octane, HPE a anticipé cette situation en interagissant avec tous ces nouveaux outils, les plus utilisés par les développeurs et équipes de production. ALM Octane devient le

point de ralliement aussi bien des développeurs que des testeurs, ou des équipes d'exploitation pour concevoir le plus rapidement, la meilleure application au service des utilisateurs.

Ainsi, HPE ALM Octane utilise intimement des solutions comme Jenkins ou TeamCity pour l'intégration continue, les tests et le reporting associé. La combinaison avec GIT apporte la gestion des codes sources et la gestion des scripts de test manuels. L'automatisation de tests en amont est possible en mode Business-Driven Development (BDD) via le support de Gherkin (permettant aussi l'automatisation de tests manuels). Sans oublier la capacité d'intégrer de multiples outils d'automatisation de tests comme HPE Unified Functional Testing, HPE LeanFT, HPE StormRunner Load, ou encore Selenium.

ALM Octane est aussi optimisé pour intégrer les APIs REST utilisant le très populaire méta-langage Swagger. Autant de possibilités qui apportent plus de fluidité et de qualité aux développements agiles, dans une démarche DevOps.

La rupture dans la continuité

L'entreprise veut être réactive et pouvoir adapter ses applications en un clin d'œil. Une attente à laquelle HPE ALM Octane répond avec une approche de type "Cloud First" rendant la solution accessible de tout lieu, à tout moment et en toute sécurité. Et toutes les actions sont tracées et mesurées, soutenues par des analyses de type Big Data pour optimiser les traitements et leur efficacité.

Parce que la rupture totale est généralement impossible, HPE ALM Octane peut s'intégrer avec HPE ALM pour combler le fossé entre méthodes nouvelles et approches traditionnelles, autorisant l'utilisation éventuelle d'outils classiques dans ces nouveaux projets.

Très ouverte, la solution HPE sait aussi prendre en charge la collaboration de très grandes équipes disséminées et de gros projets, en s'appuyant sur des moteurs de règles métiers et de worflows très évolués.

Enfin, les informaticiens apprécieront l'outil ChatOps qui redéfinit la collaboration entre les silos d'une organisation informatique en simplifiant les échanges entre les développeurs, les testeurs et la production. Avec son intégration dans ALM Octane, la collaboration avec ChatOps bénéficie d'un contexte enrichi et automatisé.

Bref : un ALM de nouvelle génération pour applications de l'ère digitale.

Comment HPE met en place le DevOps pour ses propres équipes

Passer d'une logique de silos entre les équipes de développement et de production à une logique de déploiement continu est un enjeu majeur pour les DSI. HPE l'a bien compris. Retour d'expérience.

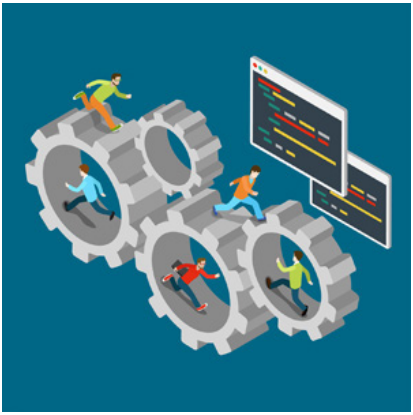
Fournisseur de solutions informatiques matérielles et logicielles pour les professionnels, HPE emploie 60 000 ingénieurs IT, dont 5 à 7 000 développeurs et membres d'équipes opérationnelles. Leur travail est d'améliorer au quotidien les logiciels utilisés par les clients de l'entreprise.

Fluidifier les relations entre développeurs et équipes de production est donc un enjeu majeur. L'objectif ? Améliorer la délivrabilité des mises à jours logicielles et être plus réactif face aux demandes des clients. C'est dans cette perspective que la méthodologie DevOps a été déployée, dans un premier temps sous forme de test avec 500 participants.

Fin 2014 un pilote de huit mois est lancé pour mieux comprendre les conséquences d'un tel projet. "Nous avons voulu avoir une approche pragmatique" explique Olivier Jacques, responsable de l'opération chez HPE IT. "Il fallait savoir si ce type de démarche était adaptée à une entreprise de… 75 ans".

Vers une logique de continuous delivery

Heureuse coïncidence, le pilote a eu lieu en même temps que la scission de HP pour former deux entités distinctes : HPE et HP Inc. “Il fallait faire la même chose avec deux fois moins de gens” explique Olivier Jacques. Donc mettre en place une méthodologie destinée à optimiser les flux de travail ne pouvait pas être une mauvaise chose ! “Avec le DevOps, nous avons voulu étendre le processus Agile, bien connu des développeurs, au périmètre de la production informatique. L’idée était de réduire la logique de silos entre le développement, les tests, la gestion du changement ou encore la sécurité”.



Pour aller de l’avant, l’équipe projet repère les structures où des initiatives de continuous delivery ont déjà été testées. Sur ce terreau fertile, de nouvelles méthodes peuvent donc être essayées. “Je ne parle plus aux équipes d’outils pour faire du code ou de builds. L’idée c’est de comprendre pour un applicatif donné quelle est la chaîne d’assemblage” évoque Olivier Jacques. “Je ne parle même plus de processus, ou de gestion du changement. Tout est géré dans la chaîne de déploiement en continu”. Dans cette logique, la chaîne d’assemblage permet à l’ensemble des acteurs de visualiser l’endroit où ils vont pouvoir s’intégrer.

Suite au pilote, l'équipe DevOps met en place un système de formation des équipes, avec 2700 inscrits à la clé. Via des Dojo DevOps, les développeurs et les équipes de production apprennent à repérer les gains d'optimisation sur les processus, et se forment à cette nouvelle méthodologie. Avant de l'appliquer au quotidien.

Des gains très perceptibles

Conséquence directe de cette refonte de la logique de production, la "réorganisation virtuelle" des structures "sans toucher à l'organigramme". Les équipes sont désormais multi-silos, et communiquent via des outils de chats (chatops) et des bots, sur lesquels sont positionnés le pipeline de la chaîne d'assemblage.

"Nous sommes passés d'une livraison tous les 6 mois à une livraison mensuelle" note Olivier Jacques quant aux bénéfices. "Sur certains applicatifs, nous avons désormais des mises à jour sans impact business, alors qu'il y avait avant 10 heures de coupure. Et nous avons réduit les tickets de 97%. Ce ne sont pas de petits changements ! Surtout, la chaîne devient fluide, les gens ont moins peur du changement".

Olivier Jacques tire de cette expérience un enseignement puissant. "Nous avons utilisé des métriques IT pour mesurer les gains de performance espérés par la mise en place du DevOps. Nous avions besoin de passer par cette étape, mais en définitive, ce sont les métriques business qui comptent